Recommending tips that support well-being at work to knowledge workers

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN ARTIFICIAL INTELLIGENCE

Thymen René Wabeke
December 17, 2014

Supervisors:
Prof. P.W.M. Desain — Donders Centre for Cognition, Radboud University
S.J. Koldijk, Msc. — Media Network Services, TNO
M. Sappelli, Msc. — Media Network Services, TNO

External examiner:
Dr. M.C. Kaptein — Donders Centre for Cognition, Radboud University

Student number:
3038491

Radboud University
Dpt. of Artificial Intelligence
Nijmegen, the Netherlands
http://www.ru.nl/ai

TNO
Dpt. of Media Network Services
Delft, the Netherlands
http://www.tno.nl

Recommending tips that support well-being at work to knowledge workers

Abstract

Knowledge workers are often exposed to a high workload. This high workload can be can be difficult to manage and may impact well-being. The present thesis examines a computer supported lifestyle coaching application (i.e. e-coach) that attempts to support well-being at work. We describe the development and evaluation of an easy-to-use recommender system that provides knowledge workers with personalized (tailored) tips that are expected to improve well-being. The evaluation of the system is split in two phases. First, the recommendation method is evaluated and optimized in an offline setting. Second, we describe a user study that investigates the usability and effectiveness of the system. The study's main objective is to examine whether tailored tips have a higher chance of being followed-up compared to randomized suggestions. Results are promising, as they suggest that knowledge workers have a positive attitude towards the implemented e-coach. On the other hand, we did not find strong evidence for our recommendation method, since tailored tips were only slightly more followed-up than tips that were not adapted the user's preferences.

Preface

For eight months, I moved from Nijmegen to Delft to conduct my internship at TNO. During this time I had pleasant and valuable experiences. I experienced what it is like to work at a company. I also met a lot of new people and enjoyed exploring Delft and its surroundings. Many people helped me in some way or another during this project. A few of them I would like to thank explicitly. First of all, my supervisors Saskia, Maya, Peter and Maurits. It was great meeting them in both the early and later stages of the project. Those meetings taught me a lot about work and discussions on a scientific level. It was great to see their different approaches that complemented each other perfectly. I also would like to thank my colleagues at TNO for their help, the daily lunches and other interesting talks we had. Additionally, I want to thank everybody who participated in my experiments or helped me find participants. Finally, I want to thank my family and friends for their support during my internship and the writing of this thesis.

Thymen René Wabeke
Nijmegen, the Netherlands
December 17, 2014

Contents

Chapter 1

Introduction

Recent research by Koppes et al. (2011) showed that almost 30% of all Dutch employees are regularly exposed to a high workload. Furthermore, 13% experiences burnout symptoms like feeling drained and experiencing a loss of motivation. These numbers emphasize the relevance and importance of stress management and improving well-being at work. Previous research in the field of stress management resulted in a number of interventions that attempt to motivate knowledge workers to enhance their coping abilities and improve their recovery from coping with high demands (e.g., Richardson & Rothstein, 2008; Schaufeli & Bakker, 2013; Wiezer et al., 2012; Ivancevich et al., 1990). Most of these interventions are initiated by employers and take an organizational approach. As an example, redesigning jobs may reduce the periods of work overload. Since well-being is highly personal, some individuals might not be motivated by these interventions to engage in more healthy behaviors (Schaufeli & Bakker, 2013). Hence, we see opportunities for a more individual approach.[1]

Research has shown that persuasive technology can play a positive and supportive role in motivating users to engage in healthy behaviors (e.g., IJsselsteijn et al., 2006; Intille, 2004). Examples of technology in the domain of stress management are so-called 'stress tests' which are widely available on the internet.[2] These tests often asses one's stress level using a questionnaire and sometimes also provide general advice on how to manage stress. According to Kool et al. (2013), most of these applications are preprogrammed or based on a standardized test. As a result, these applications may be helpful for the average knowledge worker, but individual preferences are neglected. Hence, it is questionable whether advices are always followed-up and, in turn, whether these applications optimally succeed in improving well-being at work.

Applications that support personalization to the user's needs and preferences take advantage of, for instance, user feedback, unobtrusive sensor data and intelligent algorithms in order to provide just-in-time notifications with relevant and actionable information (IJsselsteijn et al., 2006). Personalizing information such that it is intended to reach one specific person

[1] The present project is part of SWELL. The SWELL project focuses on user-centric sensing and reasoning techniques that help improve well-being at home and work. See `http://www.swell-project.net` for more information.

[2] For example, see `https://www.sterkopjewerk.nl/zelftest`.

based on an individual assessment is called tailoring (Kreuter & Skinner, 2000). Research has shown that tailoring often yields promising results (e.g., Kaptein et al., 2012; Lacroix et al., 2009). Noar et al. (2007) performed a meta-analysis in which 57 studies that examined the effect of tailoring in stimulation behavior changes were compared. Their results indicated that tailored messages are often more effective than communications that are not uniquely individualized to each person.

The present thesis examines a computer supported lifestyle coaching application (i.e. e-coach) that attempts to improve well-being at work. We describe the development and evaluation of an easy-to-use recommender system that provides knowledge workers with tailored tips. Recommender systems predict ratings for items that have not been seen by a user and, subsequently, suggest the item (or items) with the highest predicted rating to the user (Adomavicius & Tuzhilin, 2005; Ricci et al., 2010). Tips are concrete actions that can are expected to improve well-being at work. By recommending tailored tips, the e-coach tries to motivate knowledge workers to enhance their coping abilities and improve their recovery from coping with high demands.

We start the development of the e-coach by reviewing scientific literature in the field of stress management and well-being at work. Appropriate tips that could be suggested to knowledge workers are collected during this review. Several factors that researchers often use to characterize well-being interventions are also selected. Subsequently, the tips are annotated using these factors and a pilot study is conducted to investigate the tips and factors preferred by knowledge workers. The aim of this study is to collect the information needed for the implementation of our recommendation method.

Recommender systems usually operate in settings with many items and users (Ricci et al., 2010). However, the number of items and users in this project is relatively low. One of the challenges is thus applying the recommender approach to the present setting. The implemented recommendation method combines the strengths of different prediction algorithms into a single method. Users receive notifications on their smartphone when a new tip is available. Based on the user's feedback, the system is able to learn the user's preferences and generate more accurate suggestions.

An important part of this thesis focuses on evaluating the implemented recommendation method. This evaluation is split in two phases. First, the prediction algorithms that estimate ratings for unseen tips are evaluated and optimized in an offline setting. Second, a user study is conducted to examine the effectiveness and usability of the implemented system. 35 knowledge workers used the e-coach in their normal work environment for two weeks during this study. The study mainly focuses on the effect of tailoring. More specifically, it investigates whether tailored tips –generated by our recommendation method– have a higher chance of being followed-up by a user compared to tips that are not adapted the user's preferences. It is hypothesized that tailored recommendations are followed-up more often compared to randomized recommendations.

One of the main contributions of the present project lies in its scientific approach in engineering a personalized e-coach that promotes well-being at work. To our knowledge, currently available e-coaches in the field of stress management do not use recommender systems as a personalization technique. Furthermore, the applications that seem to use other personalization techniques are often commercially distributed, which makes it hard

to examine their (scientific) grounding and validity (Kool et al., 2013). Examples of such applications are the StressEraser by Western Cape Direct and the Health app by Apple.[3] Our approach is based on outcomes of existing research in the fields of stress management, machine learning and persuasive technology.

The remainder of the thesis is structured as follows. First, Chapter 2 provides more background on stress management, persuasive technology and recommender systems. In Chapter 3 we describe the procedure used to collect a set of well-being tips that could be suggested. This chapter also covers the annotation of tips and a pilot study that examines the preferences of knowledge workers. Chapter 4 explains the approach and implementation of our recommendation method. The experiments that were conducted to evaluate and optimize this method in an offline setting are described in Chapter 5. Moreover, Chapter 6 discusses a user study that examines the effectiveness and usability of the implemented recommendation method. Finally, in Chapter 7 we reflect on the findings that were revealed during the present project and suggest several directions for future research.

[3] See `http://www.stresseraser.nu/` and `https://www.apple.com/ios/whats-new/health/`. The interested reader is also referred to `http://www.digitalezorggids.nl/stress` for more examples of applications that aim to manage stress.

Chapter 2

Background

In this chapter, key themes of this thesis are introduced and explained. The concept of stress is defined in the first part (Section 2.1). Next, background is given on persuasive technology (Section 2.2). Finally, this chapter provides information about recommender systems (Section 2.3). The main objective of this section is to highlight the differences between various recommender approaches that are commonly used. Furthermore, it describes two methods for evaluating recommender systems.

2.1 The concept of stress

The term stress was first introduced by Selye (1956), who defined it as a set of physical and psychological responses to an environmental demand. Most models influenced by this definition can be summarized using the general model of occupational stress by Le Fevre et al. (2006). This model is summarized in Figure 2.1. One can observe that external demands or stressors lead to an individual perceiving stress. There are many possible stressors. For instance, task demands, a divorce, or lost keys. In addition to stressors, a perception of stress is also influenced by personal and situational characteristics. Examples of these individual characteristics are one's coping capabilities and efficacy. Due to these characteristics the same stressor can lead to different perceptions of stress among situations and individuals (Le Fevre et al., 2006).

Models on occupational stress often describe some balance between demands or stressors on the one hand, and resources, rewards and/or recovery on the other hand (see Wiezer et al., 2012, for a survey). Once there is an imbalance or misfit between the two sides, an individual is likely to perceive stress. The Effort-Reward Imbalance Model by Siegrist (1996), for instance, states that a worker's efforts and his rewards should be in balance. An imbalance might occur when the worker's rewards are smaller than his efforts (e.g., due to overcommitment), which can result in the perception of a high stress level. Perceived stress may result in behavioral, physical, and/or psychological effects. For example, experiencing pain in the shoulders or a lack of motivation. Especially if these experiences are long-standing, the worker's health is at risk (Siegrist, 1996). Two other well-known models of occupational stress stress are the Person-Environment-Fit model which highlights the

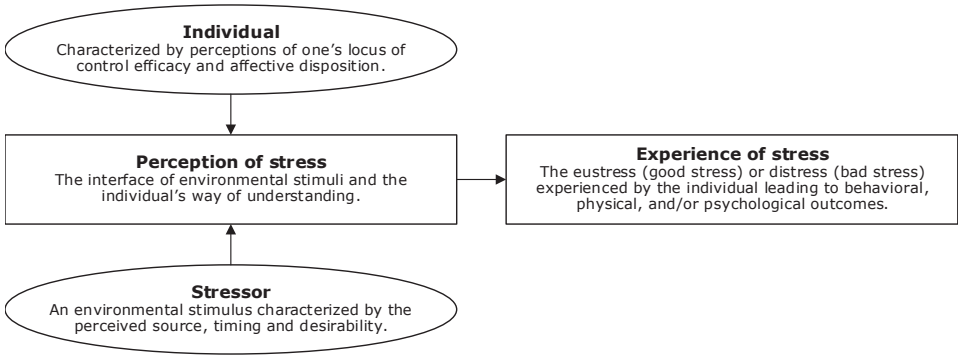


Figure 2.1: The model of occupational stress by Le Fevre et al. (2006). This model emphasizes the influence of stressors and the individual on the perception of those stressors.

balance between individual and environmental characteristics (French et al., 1981) and the Demand-Control model which is based on the fit between experienced job demands versus one's experienced control over these demands (Karasek & Theorell, 1992).

The tips that will be recommended by the e-coach create awareness about possible stressors and encourage knowledge workers to recover from coping with high demands. The e-coach thus attempts to decrease the amount of perceived stress. Chapter 3 provides more details about the tips.

2.2 Persuasive technology

IJsselsteijn et al. (2006) defines persuasive technologies as computer systems that are *intentionally designed to change a person's attitude or behavior*. Kaptein (2012) describes several examples of persuasive technologies in four application areas. For instance, researchers have investigated the effects of feedback generated by smart energy meters on the user's energy consumption (van Dam et al., 2010). Furthermore, systems like Philips DirectLife and Fitbit which monitor their users and attempt to persuade them to maintain healthy lifestyles are commercially available.[1] In this thesis, an e-coach is described that motivates knowledge workers to perform tips that support well-being at work. Our e-coach can thus be seen as a persuasive system, since by recommending tips it aims to persuade it's users to perform certain behaviors.

Research in the domain of persuasive technology resulted in many insights and several frameworks that support the design of persuasive systems. For example, Fogg (2009b) describes an eight step framework that emphasizes the importance of defining an appropriate and simple behavior to target for change. Fogg (2009b) also encourages designers to learn from prior examples and to incrementally expand a persuasive system based on small successes. Moreover, Cialdini (2001) describes six characteristics of a persuader that influence compliance to a request: reciprocity, commitment/consistency, social validation, liking, au-

[1] See `http://www.directlife.philips.com` and `http://www.fitbit.com`.

thority and scarcity. First, *reciprocity* states that people have the tendency to return a favor. Second, requests that are *consistent* with people's opinions increase compliance. Third, *social validation* means that people tend to do what others do. Request that are common have a higher compliance. Fourth, compliance is increased when persuaders are *liked*. Fifth, compliance is also increased when persuaders have a high perceived authority. Finally, whatever is *scarce* is often considered more valuable and increases compliance. The six above as well as other non-mentioned characteristics that influence compliance have been heavily researched (e.g., Cialdini, 2004; Fogg, 2002; Kaptein et al., 2012). For example, Kaptein et al. (2012) defined persuasive profiles based on these influence characteristics and showed that providing participants with persuasive messages that are adapted to one's persuasive profile lead to a decrease in snacking consumption.

Persuasive technologies may exploit unobtrusive sensor data, user feedback and intelligent algorithms to provide it's users with information that is adapted to their context/situation or personalized to their personal needs (IJsselsteijn et al., 2006; Acampora et al., 2013). Personalizing information such that it is intended to reach one specific person is called tailoring (Kreuter & Skinner, 2000). In this project, recommendations for tips are tailored, as we match tips with the user's preferences. This approach seems promising, because research has shown that tailoring often increases the effectiveness of an e-coach. Noar et al. (2007), for example, performed a meta-analysis in which 57 studies that examined the effect of tailoring in stimulation healthy behavior changes were compared. Their results indicated that tailored messages are often more effective than messages that are not uniquely individualized to each person.

2.3 Recommender systems

In this project we use a recommender system to generate tailored tips. Recommender systems –often called recommenders– are software tools that provide suggestions for items to be of use to a user (Ricci et al., 2010). In their simplest form, recommenders can be seen as machine learning algorithms that attempt to predict how a user would rate a specific item and subsequently recommend the item(s) with the highest predicted rating(s) (Adomavicius & Tuzhilin, 2005). Recommender systems are used to suggest a variety of items. For instance, Netflix, who provides an online movie rental service, utilizes it to suggest suited movies to its users (Bennett & Lanning, 2007). Another well-known example is online retailer Amazon, who uses a recommender to suggests books, CDs and other products that a user is likely to buy (Linden et al., 2003).[2] To our knowledge, currently available e-coaches in the field of well-being do not use recommender systems as a personalization technique. Given that recommender systems have been used in other application areas to, for instance, provide tailored suggestions for movies and songs, we think that these systems may also effectively recommend tailored tips. Therefore, this thesis describes the development of such a recommender system and examines whether this approach indeed is effective.

Figure 2.2 shows the interaction cycle that is used by most recommender systems. First, recommendations are generated by estimating ratings based on a user model. This model

[2]See also `http://www.netflix.com` and `http://www.amazon.com`.

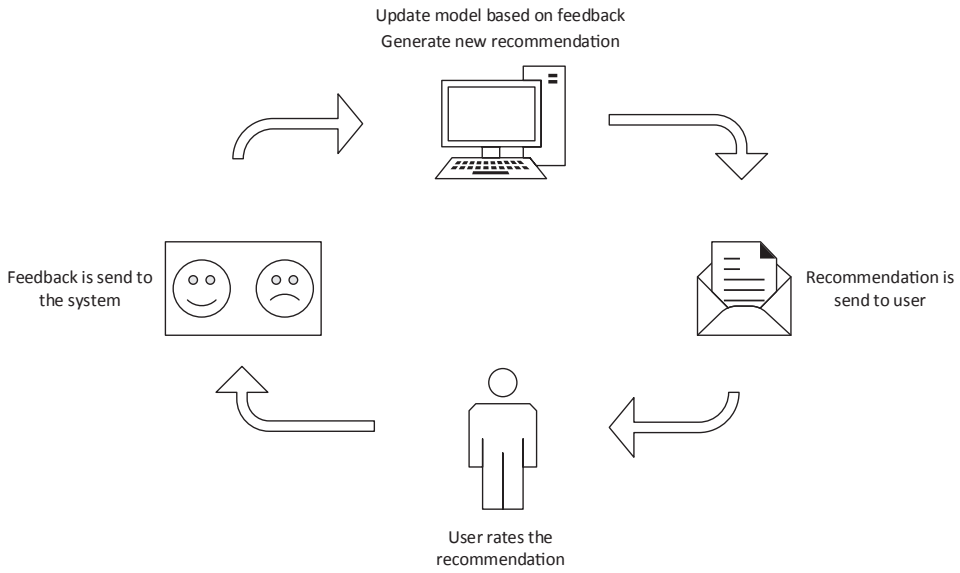


Figure 2.2: The interaction cycle of most recommender systems.

is often constructed with knowledge about the user (e.g, demographics), the user's interaction history (e.g., which items are rated/bought) and the available items (e.g., item features like the genre of a movie). The algorithms that estimate ratings are often called predictors and can follow various approaches. The most common approaches are collaborative-filtering, content-based and hybrid forms (Adomavicius & Tuzhilin, 2005; Burke, 2002). Collaborative-filtering approaches predict ratings based on items that users with similar preferences liked in the past. Content-based approaches estimate rating based on the features of items that a user liked in the past. Hybrid forms combine elements of both the collaborative and content-based approach. These approaches are detailly explained in the next subsection.

Suggestions for suited items are presented to the user in the second step of the interaction cycle. Subsequently, the user can provide feedback about the recommended item. Two types of feedback are distinguished (Jawaheer et al., 2010). Explicit feedback allows users to unequivocally express their preferences for items. Rating scales are a prime example of explicit feedback. On the other hand, implicit feedback is generated by making inferences about the user's behavior. For example, if user stops watching a movie after a few minutes the system may infer that he does not like the movie. Finally, the user's feedback is send back to the recommender system. The user profile is updated based on this new knowledge and novel suggestions can be generated.

2.3.1 Prediction approaches

This subsection explains four commonly used prediction approaches. The recommendation method that was implemented for this thesis is based on these approaches. Chapter 4

describes the implementation of this recommendation method.

Collaborative filtering approach

Prediction algorithms that take a *collaborative filtering* approach identify the user's neighbors and aggregate ratings from these neighbors in order to estimate a rating (Schafer et al., 2007; Su & Khoshgoftaar, 2009). Subsequently, a user receives recommendations for items that other users with similar preferences liked in the past. In other words, users with a rating behavior x tend to prefer item y; if user has a rating behavior similar to x, this user is likely to prefer y.

An important strength of collaborative filtering is that these algorithms exploit rating data of the whole community and generally do not depend on structured data like item features. This is especially useful when describing the characteristics that capture different preferences is hard. For example, what characteristics make one like a song? A downside is that collaborative filtering relies on the assumption that similarities in previously rated items lead directly to similarities in unseen items.

Collaborative filtering approaches also face challenges. If only a few ratings are observed for an item, there is a small chance that two neighbors rated the same item. Hence, it can be difficult to aggregate a reliable rating. Effective prediction of ratings from a small number of examples is important to overcome this challenge of *limited coverage* (Adomavicius & Tuzhilin, 2005). Furthermore, collaborative-filtering algorithms often face the *cold start problem*, which refers to a serious decline of the recommendation quality when only a small number of ratings are available (Ahn, 2008). Finally, when a user's preferences are not similar to that of any other, no reliable neighbors can be found and, hence, generating accurate predictions becomes difficult.

Content-based approach

Another widely used prediction approach is called *content-based* (Ricci et al., 2010; Lops et al., 2011). Algorithms taking this approach often generate a classifier that fits the rating behavior of a user and use this classifier to predict ratings for unseen items. Such classifiers usually depend on item features. For example, user x prefers feature y; feature y applies to item z; so item z is suited for user x.

A strength of content-based approaches is that a user's preferences for certain features are sufficient for generating suitable recommendations. In other words, these techniques do not depend on ratings given by other users. To successfully find suitable items though, content-based algorithms need to have sufficient information about items. Information about items can sometimes be parsed automatically by a computer, but often needs to be assigned manually. Issues may arise if the information about items is insufficient. For example, if two different items are represented by the same set of features, they are indistinguishable (Adomavicius & Tuzhilin, 2005).

Utility-based approach

A third approach for predicting ratings is called *utility-based* (Burke, 2002; li Huang, 2011). Algorithms taking this approach do not attempt to learn models about the user's preferences, but rather base their predictions on a computation of the utility.

The benefit of utility-based algorithms is that they do not face problems involving new users, new items, and data sparsity. The underlying reason of this advantage is that these algorithms use utility functions to estimate ratings, instead of a model that is derived from observed ratings. Therefore, utility-based algorithms are sometimes called baseline algorithms and used to provide predictions for new users (Ekstrand et al., 2011).

The biggest challenge faced by utility-based approaches is to design a proper utility function (Burke, 2002). A utility function can be designed by explicitly asking new users about their preference for certain features. For example, a system might ask new users whether they like classical music. The importance of the genre 'classical' depends on user's answer. A downside of such designs is that it increases the number of burden interactions (Burke, 2002).

Hybrid approach

The previous paragraphs discussed three commonly used prediction approaches. It was shown that each approach has its own strengths and challenges. Hybrid approaches blend two or more prediction algorithms to gain better performance with fewer of the drawbacks of any individual one (Kim et al., 2006; Adomavicius & Tuzhilin, 2005).

Hybrid approaches can use different methods for combining the individual predictions into a final prediction. One method is to switch between prediction algorithms depending on the current situation (Burke, 2002). For example, if a collaborative-based algorithm cannot generate a prediction with sufficient confidence, the system uses a utility-based prediction. When enough ratings are collected, the collaborative-based algorithm is able to generate confident predictions. At this point, the system may switch to the collaborative-based algorithm. Switching hybrids allow a recommender system to be sensitive to the strengths and weaknesses of its prediction algorithm. However, switching hybrids introduce additional complexity into the recommendation process, because the switching criteria need to be determined.

Another frequently used method to combine multiple predictions is to control the influence of prediction algorithm on the final utility using weights (Burke, 2002). For example, a collaborative-filtering algorithm is given more weight than a content-based algorithm because the former generates more accurate predictions. A benefit of weighted hybrids is that accurate prediction techniques have a larger impact on the the final prediction. This increases the chance of obtaining accurate predictions in different situations and allows the recommender system to be sensitive to the strengths of its prediction algorithms. However, weighted hybrids also make the recommedation process more complex, since weights need to be determined for each prediction algorithm.

2.3.2 Evaluating recommender systems

The evaluation of recommender system traditionally focused on optimizing prediction accuracy. The prediction accuracy can be assessed using several evaluation metrics (Herlocker et al., 2004). For instance, the root mean square error (RMSE) which measures the distance between observed and predicted ratings is often used. Metrics like precision and recall are also frequently used to evaluate the performance of a prediction algorithm in classifying whether the user would accept or reject the item.

Optimizing prediction accuracy implicitly assumes that users are always interested in the items with the highest utility (McNee et al., 2006). However, previous research suggests that high prediction accuracy does not always correlate with high user satisfaction (Pu et al., 2011). For example, users sometimes like a recommender system to suggest items they would not have thought of themselves. Therefore, researchers recently began to evaluate user satisfaction and the user's subjective opinion about suggestions (Knijnenburg et al., 2012; Pu et al., 2011). Although these evaluations can provide valuable insights, conducting a user studies that is needed for such an online evaluation is time consuming and expensive. This may thus explain why research examples of online user-centric evaluations of recommender systems are relatively sparse.

Both mentioned evaluation methods were used to examine the recommender system that was implemented during the present project. Chapter 5 describes several offline experiments that were used to evaluate the prediction accuracy of our recommendation method. Moreover, Chapter 6 discusses a user study that was used to evaluate the recommendation method in an online setting.

Chapter 3

Tips that promote well-being at work

This chapter focuses on the well-being tips that will be recommended by the e-coach. The requirements a tip must meet are explained in the first part of the chapter (Section 3.1). Here, we also briefly mention the sources of tips. The second part of this chapter focuses on the annotation of tips (Section 3.2). Next, the results of a pilot study are discussed (Section 3.3). The main objective of this study is to reveal insights on the preferences of knowledge workers regarding the well-being tips. These insights will be incorporated while implementing the recommendation method. Finally, we reflect on the pilot study (Section 3.4).

3.1 Defining appropriate tips

An important aspect of a well-designed system that recommends well-being tips are the tips themselves. The FBM model by Fogg (2009a) describes the elements of persuasive messages that are needed to effectively invoke a target behavior. This model was used as a guideline while composing the tips and states that one only performs a target behavior when one is sufficiently motivated, has the ability to perform the behavior and is triggered to perform the behavior (Fogg, 2009a). The following requirements for tips were defined based on these three dimensions.

First, a tip should contain an action/behavior that is expected to improve well-being at work. Each tip also comes with a short explanation about how the action may support well-being. It is expected that this explanation increases the motivation of knowledge workers to perform a tip. Fogg (2009a) also states that designers of persuasive technology can increase a user's ability to perform a target behavior by making the behavior easier. The simplicity of a behavior depends on six elements: time, money, physical effort, mental effort, social deviance and routine (Fogg, 2009a). We tried to ensure the simplicity of tips by only allowing actions that do not take more than three minutes, spending money, a special location, or special materials. Finally, the e-coach triggers knowledge workers to perform a tip by notifying them about a new tip three times per working day. These notifications are send at random moments between 10-11 AM, 13-14 PM and 15-16 PM. These time slots were chosen, because they ensure an even spread of the tips over the workday of most knowledge workers.

Various scientific articles, websites and magazines on well-being were reviewed to find appropriate tips (e.g., Ivancevich et al., 1990; Smith & Segal, 2014; The American Heart Association, 2014). These materials sometimes described a complete therapy. In these cases, one or more concrete actions were taken from the therapy. The set of tips was constructed such that there was variation between tips. For instance, not only tips focusing on exercises were selected, but also tips that promote scheduling. Each tip starts with the action. Subsequently, the motivation about how the action can support to well-being is mentioned. This search process resulted in a set of 54 tips. Three examples are provided below to give an impression.[1] All tips can be found in Appendix A.1.

> "Write down your last success. According to many psychologists it is important to be proud of the things you have achieved. It will give you positive energy and it will enhance your self-esteem."

> "Go stand on your toes. After three seconds you stand back normally. Repeat procedure at least 10 times. This exercise will train the muscles in your legs."

> "Keep focused and drink a glass of water. A shortage of water creates fatigue."

3.2 Annotation of tips

This section describes the annotation of tips using eight factors. We start with introducing these factors. Next, the section describes the annotation study that was conducted.

1. Stress management interventions (SMIs) are often characterized by their **type** (e.g., Richardson & Rothstein, 2008; Ivancevich et al., 1990). The following types are frequently distinguished and were thus used while annotating tips: *cognitive-behavioral*, intended to change an individual's appraisal and responses; *creative*, intended to stimulate the individual's mind in creative way; *physical exercises*, providing a physical release; *diet*, encouraging or discouraging certain food and/or drinks; *journaling*, assisting the individual to monitor his behavior; *relaxing*, bringing about a physical and/or mental state that is the physiological opposite of stress; *social* , encouraging or discouraging certain social interactions; and finally *time-management*, helping people to manage their time.

2. SMIs generally **focus** on the individual, the organization, or some combination (Giga et al., 2003). The following specific focuses were derived from these broad focuses and were used to annotate tips: one's *physical situation*, *social situation*, *recovery*, and *work situation*. The three first mentioned focuses mainly target on what Giga et al. call individual-level, whereas the latter one targets both at the individual and organization (Giga et al., 2003).

3. SMIs can also be characterized by their level (LaMontagne et al., 2007; Richardson & Rothstein, 2008). First level interventions aim to avoid stressful situations, for

[1] Tips were originally composed in Dutch. The tips mentioned in this section are translated.

instance, by creating awareness about possible stressors. Second level interventions help employees to better cope with stressful conditions and often prevent that stress is experienced. Finally, third level interventions support recovery from experienced stress. Based on these three levels, the following **goals** were distinguished: a tip may aim at *creating awareness*, at *preventing* the experience of stress, or at *recovery* from coping with high demands.

4. SMIs also differ with respect to the **required presence of colleagues**. Some SMIs effect an entire organization or department and often require peers, whereas others only apply to a single employee (Giga et al., 2003). For example, a complete organization is often invited to attend at training and education programs. On the other hand, yoga exercises may easily be conducted without the presence of colleagues. For the current purpose, we distinguished tips that are preferably performed *without the presence of peers* (e.g., 'Sing a song'), tips that do *require peers* (e.g., 'Tell a joke'), and for which it *does not matter* whether peers are present (e.g., 'Take a five minute break').

5. The **time** required to perform a target behavior is one of the elements simplicity and influences one's compliance (Fogg, 2009a). Hence, annotators defined the time required to perform a tip (in minutes).

6. The amount of brain cycles or **mental effort** needed to perform a target behavior also influences one's ability to perform a target behavior (Fogg, 2009a). The annotators thus defined the amount of mental effort needed to perform a tip on a scale from 0 to 10, where 0 denotes no effort (e.g., 'Eat a peace of fruit') and 10 maximum effort (e.g., 'Set goals for this week').

7. The amount of **physical effort** needed to perform a tip is also mentioned by Fogg (2009a) as an element of simplicity. The annotators defined the amount of physical effort needed to perform a tip on a scale from 0 to 10, where 0 denotes no effort (e.g., 'Cleanup your mailbox') and 10 maximum effort (e.g., 'Perform some push-ups').

8. The extent to which a tip **deviates from the social norm** also influences compliance towards a persuasive request (Fogg, 2009a) and was thus used to characterize tips. The deviance is displayed on a scale from 0 to 10, where 0 denotes no deviance and 10 total deviance.

3.2.1 Method

Annotators 1 man and 2 women (age 20-29) annotated the tips. All annotators were knowledge workers. Two were experts in the field of stress management.

Materials and procedure The annotators received an e-mail to participate in this study. By clicking on a link annotators were taken to an online questionnaire. This questionnaire was composed to compare different annotations. The questionnaire consisted of eight stages, each covering another factor. At the end of each stage participants were able to

	Annotator 2	Annotator 3
Annotator 1	0.78	0.58
Annotator 2		0.54

Table 3.1: The inter-annotator agreement calculated using Cohen's kappa. Values below 0.5 represent poor agreement, whereas values above 0.75 represent excellent agreement (Fleiss et al., 1981).

describe the answer options that were missing in their opinion. They could, for example, indicate whether a specific goal was missing. The sequence in which tips were displayed was randomized among the different stages. At the end of the questionnaire, annotators were asked to answer some demographical related questions. The whole procedure took about 30 minutes.

3.2.2 Results and final annotation

The annotators annotated all tips using the eight factors described above. The inter-annotator agreement was calculated using Cohen's kappa. Table 3.1 shows Cohen's kappa for each pair of annotators. The agreement between annotator 1 and 2 is excellent, whereas the agreement between annotator 1 and 3, and 2 and 3 is fair to good (Fleiss et al., 1981). The higher agreement between the first and second annotator might be explained by the fact that both are experts in the field of stress management and are thus more familiar with the answer options. The fact that the third annotator more often selected the *Other*-option also strengthens the intuition that the third annotator had a different understanding of the answer options.

For all quantitative factors, the final annotations were defined by calculating the mean of the three annotations. For all categorical factors, the final annotation was constructed using majority voting. Occasionally, all annotators annotated a tip differently. In these cases the annotation of either annotator 1 or 2 was chosen, because they had a higher overall agreement. Annotators also described the answer options that were missing in their opinion. All three annotators indicated that a focus on a worker's mental situation was missing (e.g., 'Close your eyes and imagine happiness'). Furthermore, two annotators thought that the type of tips that are about one's working conditions was missing (e.g., 'Adjust your sitting posture'). In the final annotation these novel options were picked in situations where at least two annotators had chosen the *Other*-option and the new option seemed appropriate. Appendix A.2 shows the final annotation of all tips.

3.3 Pilot study investigating preferences

The remainder of this chapter describes a pilot study that was conducted to reveal insights that can help the implementation of the recommendation method. We performed different analyses to find answers on the following questions:

- Do participants have different opinions about the tips? An important task of the recommender system will be to filter out those tips that are probably disliked by a user. It is expected that participants do no share the same opinion about all tips, which makes the development of a recommender system that filters tips based on the user's preferences relevant.
- Do the underlying characteristics of tips relate to changes in obtained ratings? It is expected that the features of tips predict the participants' ratings, which justifies the use of these features in a content-based prediction approach.
- Can participants be clustered based on their rating behavior? It is expected that clusters can be constructed.

The present section first describes the general methodology of the study. Later, the questions mentioned above are covered one by one.

3.3.1 Method

Participants 16 man and 10 women volunteered to participate in this study. All participants were knowledge workers. The average working week of participants was 35.9 hours (SD 11.0). The participants work with a computer for 30.2 hours a week (SD 12.7). The participants' ages were distributed as follows: 6 participants were 20-29 years old; 5 were 30-39 years old; 5 were 40-49 years old; 7 were 50-59 years; and 3 were older than 60.

Materials and procedure Participants were invited via e-mail to participate in this study. By clicking on a link participants were taken to an online questionnaire. Here, they were asked to rate 54 tips. Participants were told that all tips were designed to improve well-being at work. Ratings were given using a 5-point Likert item proceeded by the following question: "Imagine you are at your office and want to improve your well-being. Would you follow-up this tip?"[2] The tips were split in three pages each containing 18 tips. The sequence in which these pages were displayed to the participants was randomized. At the end of the questionnaire participants were asked to answer some demographical questions. The whole procedure took about 10 minutes.

Data preparation Some participants only used a specific range of the Likert items. Therefore, all responses were rescaled within-subject to a 0-1 scale, where 0 denotes a maximum negative rating and 1 a maximum positive rating.

3.3.2 Interpreting different opinions

The following paragraphs focus on whether participants have different opinions about the tips. Furthermore, we attempt to interpret the tips that are often (dis)liked and the tips on which participants often (dis)agree.

[2]The questionnaire was written in Dutch. The original question was "Stel, je zit op je werk en wil je welzijn verbeteren. Ga je deze tip opvolgen?"

Tip	Mean	SD
drink_water	0.79	0.24
department_walk	0.77	0.20
short_break	0.76	0.22
weekend_plans	0.74	0.17
focus_environment	0.74	0.17
chat_colleague	0.73	0.22
coffee_break	0.72	0.28
eat_fruit	0.72	0.24
set_goal_today	0.71	0.28
no_screen	0.71	0.20

(a) Tips with the highest mean rating. These tips are often liked.

Tip	Mean	SD
sing_song	0.15	0.20
pushups	0.17	0.22
hug_someone	0.19	0.28
switch_desks	0.25	0.30
make_drawing	0.25	0.22
touch_shoes	0.34	0.06
listen_music_classical	0.35	0.24
reverse_plank_stretch	0.37	0.24
side_turn_stretch	0.38	0.24
tell_joke	0.38	0.26

(b) Tips with the lowest mean rating. These tips are often disliked.

Table 3.2: Tips with the lowest and highest mean rating. Ratings are in the range from 0 to 1. Only tags of tips are displayed to safe space. The full content of tips can be found in Appendix A.1

Influence of social deviance Performing tips that are common in an office environment receive higher ratings, whereas tips that are more deviant in a common office environment are less likely to be performed. Table 3.2 contains examples for this observation. For instance, the tips drinking a glass of water and taking a short break are common in an office environment and receive high ratings (respectively 0.79 and 0.76). More deviant tips like singing a song or performing push-ups receive lower ratings (respectively 0.15 and 0.17).

Preference for tips that do not require a behavior change Tips that are probably within one's daily routine receive higher ratings compared to tips that require a behavior change. Taking a coffee break, for example, has a mean rating of 0.72, whereas making a drawing obtains a mean rating of only 0.25. The former tip is probably within ones daily routine, whereas the latter is not. More examples can be found in Table 3.2.

Preference for social tips Tips that encourage social contacts tend to receive higher ratings. In Table 3.2 one can, for instance, observe that the tip that encourages one to make a department walk has a mean rating of 0.77 and the tip that suggest to chat with a colleague obtains a mean rating of 0.73.

Disagreement about communication related tips It is interesting to see that tips related to communication tend to have relatively high standard deviation in obtained ratings. It seems that, for instance, some participants are willing to turn off their phone and notifications for a period of time, whereas others do not (standard deviation of respectively 0.36 and 0.37) .

Tip	Mean	SD
no_coffee	0.49	0.40
notifications_off	0.59	0.37
phone_off	0.47	0.36
take_stairs	0.43	0.36
listen_music_pop	0.46	0.35
listen_music_relax	0.47	0.33
adjust_brightness	0.42	0.32
watch_funny_clip	0.53	0.32
watch_ted_talk	0.48	0.30
switch_desks	0.25	0.30

(a) Tips with the highest standard deviation. Tips with a high standard deviation were rated differently among raters. One can also say that raters have different opinions about these tips.

Tip	Mean	SD
set_goals_week	0.65	0.17
focus_environment	0.74	0.17
weekend_plans	0.74	0.17
set_goal_today	0.71	0.17
sing_song	0.15	0.20
no_screen	0.71	0.20
give_compliment	0.67	0.20
focus_breath	0.70	0.20
department_walk	0.77	0.20
deep_breath_minute	0.65	0.20

(b) Tips with the lowest standard deviation. Tips with a low standard deviation are tips that were rated quite uniformly. Most raters share the same opinion about these tips.

Table 3.3: Tips with the highest and lowest standard deviation. Ratings are in the range from 0 to 1. Only tags of tips are displayed to safe space. The full content of tips can be found in Appendix A.1

Agreement about time management related tips Most time management related tips receive high ratings. Furthermore, these tips have a low variance suggesting agreement among participants. The tip that promotes setting weekly goals, for example, obtains a mean rating of 0.64 and standard deviation of 0.17. More examples can be seen in Table 3.3.

3.3.3 Generating a predictive model

The following paragraphs focus on whether the features of tips that were annotated earlier in this chapter are associated with changes in the participants' ratings.

Method Multiple regression analysis using a backward stepwise method was performed to construct a predictive model. Dummy variables were created for all categorical factors before analysis. The analysis was split in two parts. First, the multiple regression analysis was used to find the variables that seem the most relevant. Second, the analysis was repeated, but now all irrelevant variables were excluded (i.e., the ones that were statistically redundant in the first analysis).

Results The multiple regression analysis resulted in a model containing nine variables (see Table 3.4). The obtained model explained 20% of the variance in the ratings ($R^2 = 0.20$, $F(9, 1394) = 38.83$, $p < 0.01$). Although the explained variance is relatively small, the model still provides interesting insights on how different feature values are associated with changes in the participants' ratings while holding other features constant. It was found

Factor	Predictor	B	SE B
Constant		0.55	0.02
Goal	Recover vs. Prevent	-0.08	0.02*
	Recover vs. Awareness	0.10	0.03*
Type	Cognitive vs. Creative	-0.07	0.03*
	Cognitive vs. Food	0.16	0.04*
	Cognitive vs. Relaxing	0.20	0.03*
	Cognitive vs. Time-management	0.16	0.03*
	Cognitive vs. Work-conditions	0.10	0.04
Peers	Required vs. not	-0.13	0.03*
Social deviance		-0.21	0.05*

Table 3.4: The final model obtained by the multiple regression analysis using a backward stepwise method. The B column contains the beta value of each predictor, which describes to what degree each predictor affects predicted rating if the values of all other predictors. Each beta values has an associated standard error (SE B) indicating to what extent the beta varies across different samples. The standard errors are used to determine whether or not the beta value differs significantly from zero. All features marked with * are statistically significant ($p \leq 0.05$). The model converged after four steps. $R^2 = 0.20$ for Step 1; $\Delta R^2 = 0.0$ for Step 2, Step 3 and Step 4; resulting in a final model with $R^2 = 0.20$, $F(9, 1394) = 38.83$ and $p < 0.01$.

that all features related to the goal of a tip represents a significant change in the participants' ratings. Tips that aim to prevent stress obtain lower ratings compared to tips that promote recovery. However, tips that create awareness are liked more compared to recovery and prevention tips. The model also shows that tips which are preferably performed in the presence of peers are associated with higher ratings compared to tips that do not require peers. However, a significant change in ratings was not found between tips that do require peers and those that can be performed either with our without peers. The social deviance of a tip also significantly predicts changes in ratings. Lower ratings are predicted for more deviant tips. The types creative, food, relaxation and time management tips are also significantly associated with the participants' ratings. The types social, journaling and exercising do not significantly predict variations in ratings. The features that are related to the focus of a tip were already redundant in the first step of the analysis. The same goes for the required time and the required mental and physical effort of a tip.

3.3.4 Clustering participants

Cluster analysis was performed to investigate whether groups of participants who rated tips similarly could be constructed.

Method Clusters were constructed using k-means clustering. In the first step, the number of clusters in the data was defined. One common method of choosing the number of clusters

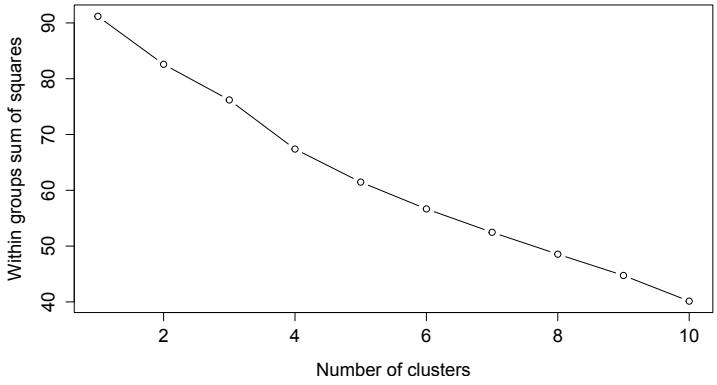


Figure 3.1: The sum of squared error (SSE) for different numbers of clusters.

is to compare the sum of squared error (SSE) for different numbers. The number of clusters where the reduction in SSE slows dramatically indicates the appropriate number of clusters. Figure 3.1 shows the SSE's that were obtained for different numbers of clusters. The SSE decreases quite linearly, which makes it hard to estimate the appropriate number of clusters and, in turn, makes it unfeasible to construct clusters in a reliable way. Therefore, it was decided to stop the cluster analysis at this point.

Results The fact that the SSE decreases linearly indicates that the data does not seem to be described well by clustering.

3.4 Discussion

In this chapter we explained the procedure used to generate and annotate the set of tips that will be used by the recommender system. Finally, a pilot study was described in which we investigated the preferences of knowledge workers.

An important task of the recommender system will be to filter out those tips that are probably disliked by a user. If it would be the case that all participants were positive about all tips, the system can just recommend any tip, which makes the development of a sophisticated recommender system less relevant. Results indicated that participants did not agree on which tips they like, which makes it relevant to filter tips based on preferences. The fact that some tips have a high variance in ratings strengthens this intuition.

Participants indicated whether they were willing to perform a tip during the pilot study. In the user study that will be described later, users will indicate whether they actually perform a tip. It might well be that though a worker has the intention to perform a tip, he will not actually perform it. Hence, it is expected that less positive ratings will be obtained during the field study and that the variation in rating behavior increases.

The pilot study also focused on the characteristics of tips. A multiple regression analysis was performed to investigate whether the annotated features are associated with variations in ratings. It was shown that different goals, types and the required presence of peers represent

a significant change in obtained ratings. Hence, these features can be used for content-based prediction approaches. Furthermore, the predictive model obtained by the regression analysis will form the basis of the utility-based prediction algorithm.

A critical note regarding the multiple regression analysis is that though the backward stepwise method helped us to easily conduct a multiple regression analysis, it also also has some drawbacks (Harrell, 2001). During the project we have gained more knowledge about how to manually conduct regression analyses.[3] In future studies we thus will be more inclined to manually construct a regression model instead of using an automated stepwise method.

The obtained ratings were rescaled within subject before the analysis that are described in this chapter were performed. Whether it is appropriate to use such standardizing methods has been subject of debate (e.g., Fischer & Milfont, 2010; Jamieson et al., 2004). One of the issues with standardizing Likert-scales is that the differences between levels cannot be presumed equal (Jamieson et al., 2004). For example, it is questionable whether the difference between *strongly disliking* and *disliking* is equal to the difference between *disliking* and being *neutral* about a tip. Furthermore, some information about the participants' attitude towards tips was lost by rescaling ratings to their full-response range. For instance, imagine a participant whose lowest rating was *neutral*. After rescaling, the participant's lowest rating (i.e. *neutral*) equals the lowest possible rating (i.e. *strongly dislike*). Hence, it can be expected that the average rating of this participant is quite neutral now. Moreover, the information about the participant's positive attitude is lost after rescaling. Given this discussion about standardizing Likert-scales, we will be more cautious with rescaling data in future studies.

The next chapter describes the implementation of the recommendation method, including all prediction algorithms.

[3]The book about multilevel models by Gelman and Hill (2006) was especially helpful.

Chapter 4

Implementation of the recommendation method

This chapter explains the recommendation method that will be examined later in this thesis. First, we describe the implementation of three prediction algorithms that are used to estimate ratings. This includes an explanation of the collaborative-based, content-based and utility-based predictor (Section 4.1, 4.2, and 4.3). Subsequently, we show that each predictor has its own strengths and explain the hybrid prediction strategy which combines these strengths (Section 4.4). Next, this chapter focuses on the recommendation pipeline (Section 4.5). This pipeline integrates all steps that are performed to generate recommendations and thus summarizes the complete recommendation method. Finally, some notes about the implementation of the recommendation method are discussed (Section 4.6).

4.1 Collaborative-based predictor

This section explains the implementation of the collaborative-based prediction algorithm. The collaborative-based predictor estimates a rating $r_p(u, i)$ of item i for user u based on observed ratings $r_o(u_k, i)$ of item i by users $u_k \in U$ that are 'similar' to u. First, Section 4.1.1 defines the concept of similar users and explains the process of constructing neighborhoods of similar users. Second, the procedure of predicting utilities by aggregating ratings from nearest-neighbors is explained in Section 4.1.2.

4.1.1 Generating neighborhoods of similar users

The collaborative-based predictor assumes that each 'active' user has a neighborhood of 'other' users (or neighbors) that are similar to the active user. In this project, the correlation in observed ratings is used to estimate the similarity between two users. The *Pearson Correlation (PC)* was chosen, because this measure has proven to be successful in other recommender systems. Furthermore, PC has the advantage that the effects of mean and variance in ratings made by a user are removed (in contrast to, for example, *Cosine Similarity*). PC is defined such that users who often rated an item similarly, have a high correlation (1 denotes total positive correlation). Users who often gave opposite ratings seem to have

different preferences and thus have a negative correlation (-1 denotes total negative correlation). If the rating behavior of two users shows both similarities and differences, no correlation is found (0 denotes no correlation).

A correlation is more reliable when it is based on more data. Hence, the algorithm uses *significance weighting*. This is a commonly used method to penalize the correlation between users when it is based on a relatively small set of co-rated items (Herlocker et al., 2002). Definition 4.1 shows how the final similarity between users is estimated based on the Pearson Correlation and *significance weighting*.

Definition 4.1. *The similarity between users u and v is calculated using a significance weighted Pearson Correlation:*

$$sim(u,v) = PC(u,v) * \frac{\min(||I_{uv}||, \kappa)}{\kappa}$$

The observed correlation is weighted/penalized when $||I_{uv}||$, the number of items that u and v have both rated, is below parameter κ.

4.1.2 Estimating predicted ratings using neighborhoods

The collaborative-based predictor estimates utilities by aggregating ratings that are given by neighbors of the active user. The neighborhood of an active user contains a given number of other users who have the highest similarity with the active user. There is, however, one restriction. If a candidate neighbor has not rated the item of which the rating is being predicted, he will not be added to the neighborhood —even if the correlation is very high. The underlying reason for this restriction is the following. If a user has not rated the item, adding the user to the neighborhood makes no sense, because no rating can be aggregated from this user. The neighborhood of an active user thus contains the η other users, that have the highest similarity with the active user and have rated the item that is currently being predicted. A more formal description of a neighborhood is given in Definition 4.2.

Definition 4.2. *Let $S_u = \{s_1, \ldots, s_{n-1}\}$ be the set of similarities between user u and all other users $u_n \in U$. The neighborhood $N_{ui} = \{n_1, \ldots, s_\eta\}$ of u when predicting item i contains the top η users with the highest similarity who have rated i.*

Once the neighborhood is estimated, a prediction can be made. A prediction is calculated by aggregating the observed ratings given by a user's neighbors. A rating is multiplied by the similarity, such that more similar neighbors have a higher influence on the prediction. Finally, the absolute sum of similarities is used to normalize the prediction to the range from -1 to 1, where -1 denotes total negative utility and 1 total positive utility. The process of calculating a prediction is formalized in Definition 4.3. Note that there is no threshold for the similarity between users. It is thus expected that the predictor's accuracy is poor when the most similar neighbor of a user in fact is quite dissimilar.

Definition 4.3. *The predicted rating $r_p(u, i_k)$ of item $i_k \in I$ for user u is calculated using the following formula:*

$$r_p(u,i) = \frac{\sum_{v \in N_{ui}} r_{vi} \times sim(u,v)}{\sum_{v \in N_{ui}} |sim(u,v)|}$$

Factor	Associated features
Goal	Recovery, prevention, and awareness.
Focus	Recover, tasks, social situation, and mental situation.
Type	Creative, food, relaxing, time-management, work conditions, cognitive-behavioral, journaling, social, and exercise.
Peers required	Yes, no, and does not matter.

Table 4.1: The features that are used to construct item vectors. All features relate to a factor. All features are binary. This means that the value of a feature is 1 if the feature applies to an item and 0 otherwise. Exactly one feature within each dimension applies to an item (e.g., an item is assigned to one type).

$sim(u, v)$ denotes similarity between u and its neighbor v and $|sim(u, v)|$ represents the absolute value of this similarity. r_{vi} denotes the observed rating by v for item i

4.2 Content-based predictor

This section explains the implementation of the content-based prediction algorithm. The content-based predictor estimates a utility $r_p(u, i)$ of item i for user u based on u's preferences. These preferences are derived from observed ratings $r_o(u, i_k)$ to items $i_k \in K$ by u. First, Section 4.2.1 explains the representation of items using feature vectors. Later, Section 4.2.2 describes the calculation of user profiles. Finally, Section 4.2.3 explains the procedure of estimating utilities by calculating the similarity between item vectors and user profiles.

4.2.1 Items represented as feature vectors

The content-based predictor requires that items are described by structured data which captures the item's distinctive characteristics. The content-based predictor uses the goal, focus, type and the requirement of peers to describe tips. These four factors were split in multiple 'dummy features'. For example, the factor *goal* was split in the features *recovery*, *prevention* and *awareness*. By splitting factors into multiple features exactly one feature applies to an item within each factor (e.g, the feature *recovery* applies to an item, all other goal related features do not apply). Table 4.1 provides an overview of all factors that were used and their associated features.

Since each item $i_k \in I$ is defined by a set of features, an item i_k can be represented as a feature vector. Each element of this vector is related to a single feature. For example, the first element describes whether the *goal* of an item is *recovery*, whereas the second element describes whether the *goal* is *prevention*. If a feature applies to an item, the corresponding value in the feature vector is set to 1. If the feature does apply, it's value is set to 0. The item representation as described above is formalized in Definition 4.4, 4.5 and 4.6.

Definition 4.4. *Let $F = \{f_1, \ldots, f_n\}$ be the set of features and $D = \{d_1, \ldots, d_n\}$ be the set of factors. Each feature $f_i \in F$ is associated with a factor $d_i \in D$.*

Definition 4.5. *Let $I = \{i_1, \ldots, i_n\}$ be the set of available items. An item $i_k \in I$ can be written as a vector of feature values $\vec{I_k} = (w_0, \ldots, w_n)$, where w_i represents the item's value of feature $f_i \in F$. This value is restricted to 0 and 1, where 0 represents a feature that is absent from the item and 1 represents a feature that applies to item.*

Definition 4.6. *Given the feature vector $\vec{I_k}$ of item $i_k \in I$. For each dimension $d_i \in D$ there is exactly one associated feature $f_j \in F$ of which the corresponding weight w_i equals 1. The weights of all other features associated with dimension d_i have a weight of 0. Hence, $||\vec{I_k}||$, the length of $\vec{I_k}$, equals $||D||$, the number of dimensions.*

4.2.2 Calculating user profiles

The content-based predictor analyzes previously rated items to learn a user profile. A user profile describes the user's preferences for all features and can thus be seen as a vector of preferences. If a user dislikes a specific feature, it's corresponding preference is set to 0. When a user favors a feature, the corresponding preference increases up to a maximum of 1.

Definition 4.7. *Let $P = \{p_1, \ldots, p_n\}$ be the set of user profiles. A user profile $\vec{P_u}$ is written as vector $\vec{P_u} = (f_0, \ldots, f_n)$, where f_i represents the user's preference for feature i. The preferences range from 0 to 1, where 0 represents a feature that is not preferred by the user and 1 represents a fully preferred feature.*

By now, the representations of items and user profiles are discussed. The following paragraphs explain the process of learning a user profile. This process is mainly based on Rocchio's algorithm for relevance feedback (e.g., Salton & Buckley, 1997). When calculating profiles using Rocchio's algorithm, all positively rated item vectors are directly added to the profile vector and all negatively rated item vectors are directly subtracted from the profile vector. If a user rates an item positively, the profile vector thus moves towards the positively rated item vector. If an item is rated negatively the profile vector moves away form the item vector. Neutral ratings are ignored, because it is not possible to estimate whether profile should be shaped towards or away from the item vector in these situation. The algorithm used for calculating profiles is formalized in Definition 4.8.

Definition 4.8. *Let I_u^+ be the set of items rated positively by user u, and I_u^- the set items rated negatively by u. When an item $\vec{I_k}$ has been rated by u, it will be added to either I_u^+ or I_u^- depending on the observed rating. Subsequently, the (updated) user profile $\vec{P_u}$ will be calculated using the following formula:*

$$\vec{P_u} = \beta \sum_{\vec{I_j} \in I_u^+} \vec{I_j} - (1 - \beta) \sum_{\vec{I_k} \in I_u^-} \vec{I_k}$$

Parameter β controls the relative importance of positive and negative ratings in shaping the profile.

If a user did not yet rate any items or only rated items neutrally, the profile vector cannot be shaped. In the beginning, all preferences are thus set to 0. This initial profile is formalized in Definition 4.9.

Definition 4.9. *If the set of positively rated items I_u^+ and the set of negatively rated items I_u^- of user u are both empty, all preferences in the user's profile vector $\vec{P_u}$ are set to 0.*

4.2.3 Estimating predicted ratings

Once a profile vector is learned, the content-based predictor is able to predict ratings. Predictions are calculated by measuring the similarity between items vectors and the user's profile vector. If an item vector and profile vector are close to each other, the item fits the preferences of a user well. Hence, a higher utility is calculated.

The similarity measure that is used to compare the user profile and item vector is shown in Definition 4.10. The numerator actually computes the similarity between the two vectors, whereas the denominator is used to normalize the obtained similarity such that all similarities range from 0 to 1.

Definition 4.10. *Given profile vector $\vec{P_u}$ of user u and feature vector $\vec{I_k}$ of item $i_k \in I$. The similarity between $\vec{P_u}$ and $\vec{I_k}$ is defined as:*

$$sim(\vec{P_u}, \vec{I_k}) = \frac{\vec{P_u} \cdot \vec{I_k}}{||D||}$$

Denominator $||D||$ denotes the number of dimensions and is used to normalize the similarity. $sim(\vec{P_u}, \vec{I_k})$ ranges from 0 to 1, where 0 represents maximum dissimilarity and 1 represents maximum similarity.

As shown in Definition 4.10, the (unnormalized) similarity is calculated using the dot product and has the following properties. First, features that do not apply to an item do not influence the similarity, because their values are 0. Second, when a feature does apply to an item, the similarity is directly influenced by the user's preference for that feature. Preferred features highly increase the similarity, whereas less preferred features result in a lower increase of the similarity.

Finally, all predicted utilities are transposed to the range from -1 to 1, where -1 denotes total negative utility and 1 total positive utility. Definition 4.11 formalizes the procedure used to calculate a prediction.

Definition 4.11. *The predicted rating $r_p(u, i_k)$ of item $i_k \in I$ for user u is calculated using the following formula:*

$$r_p(u, i_k) = sim(\vec{P_u}, \vec{I_k}) \times 2 - 1;$$

Factor	Feature	Coefficient
Constant		0.55
Goal	Prevention	-0.08
	Awareness	0.10
Type	Creative	-0.07
	Food	0.16
	Relaxation	0.20
	Time-management	0.16
	Work-conditions	0.10
Peers	Not required	-0.13
Social deviance		-0.21

Table 4.2: The coefficients that are used in the utility function of the utility-based predictor.

4.3 Utility-based predictor

This section explains the implementation of the utility-based prediction algorithm. The utility-based predictor estimates a utility $r_p(u, i)$ of item i for user u using a utility function $u(i)$. The utility function is an equation that predicts the rating of i as a linear function of i its features. Definition 4.12 shows the formalization of the utility function.

Definition 4.12. *The utility of item i is defined as:*

$$u(i) = c + \sum_{f \in F} b_f \times v_{if}$$

F denotes a vector of features, b_f represents the coefficient of feature f, and v_{if} denotes whether the feature f applies to item i. v_{if} is 1 if f applies to i and 0 otherwise. Finally, c denotes a constant value which is the output of the function when none of the features apply (or when all coefficients are 0).

As shown in Definition 4.12, the utility function uses a constant value and several coefficients. These values were derived from the predictive model that was obtained by the multiple regression analysis described in Section 3.3. Table 4.2 contains these values.

The original predictive model was trained to predict ratings in the range from 0 to 1. Hence, all predicted utilities are transposed to the range from -1 to 1, where -1 denotes total negative utility and 1 total positive utility. Note that in contrast to the other two predictors, the utility-based predictor is static. The coefficients are trained offline and do not change when the user has rated an item.

4.4 Hybrid prediction strategy

At this point, the three prediction algorithms have been described. It is expected that each predictor obtains it's best accuracy for different types of users. Section 4.4.1 describes

an ideal situation for each predictor. These different situations provide motivation for a hybrid prediction strategy that combines the strengths of the individual predictions. The implementation of this hybrid prediction strategy is explained in Section 4.4.2.

4.4.1 Motivation

Collaborative-based predictor The accuracy of the collaborative-based predictor depends on the quality of the neighborhood. If a user has many neighbors with a high similarity, it is expected that the predictor estimates accurate ratings. However, the predictor cannot estimate accurate ratings when the user's rating behavior is not similar to that of any other user. Furthermore, the predictor needs to have a sufficient amount of observed ratings in order to estimate reliable similarities. Hence, it is expected that the accuracy of the predictor increases when more ratings become available.

Content-based predictor The accuracy of the content-based predictor depends on the quality of the user profile. In turn, the quality of the profile depends on the item features and the user's rating consistency on these features. If the features do not well describe the items, or more specifically, do no well capture the users' preferences for items, the predictor is unable to discriminate liked and disliked items. High accuracies are expected for users who rate items consistently.

Utility-based predictor The utility-based predictor is based on the predictive model obtained by a multiple regression analysis. Hence, the accuracy of this predictor depends on the generalization of this model. It expected that the predictor produces relatively accurate predictions. However, in contrast to the other two predictors, the utility-based predictor does exploit observed ratings to improve its model. It is thus expected that the other two predictors are able to estimate more accurate ratings in a later stage.

4.4.2 Implementation

The previous paragraphs hypothesized that each predictor obtains it's best accuracy for different types of users and different amounts of observed ratings. To ensure high accuracies in all situations, a hybrid prediction strategy was implemented. The implementation of this strategy is partly based on AdaBoost (Freund et al., 1999), which is a frequently used method to combine multiple 'weak' predictions into a single 'strong' prediction.

The hybrid prediction strategy uses weights to control the influence of each predictor on the final prediction. The predictors' weights are estimated per user, based on the error rate of the predictor for the current user. The error rate denotes the proportion of observed ratings that are misclassified by the predictor. Definition 4.13 formalizes the procedure of determining weights.

Definition 4.13. *The weight a_{tu} of predictor t for user u calculated using the following formula:*

$$a_{tu} = \frac{1}{2} \times \ln \frac{1 - \epsilon_{tu}}{\epsilon_{tu}}$$

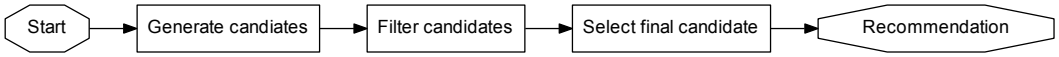


Figure 4.1: The recommendation pipeline. First, predicted ratings are calculated for each available item which results in a list of candidate recommendations (Figure 4.2 detailly displays this step). Second, the list of candidates is filtered. Finally, a candidate is selected and recommended to the user.

ϵ_{tu} denotes the error rate of predictor t for user u.

Once a weight is determined for each predictor, the individual predictions are combined using a weighted sum. This weighted sum ensures that more accurate predictions have a higher influence on the final prediction. The weighted sum is formalized in Definition 4.14.

Definition 4.14. *Given a set of predictors T, the final prediction of item i for user u is calculated using the weighted sum. The influence of utility $r_t(u,i)$ estimated by $t \in T$ on the final prediction depends the predictor's weight a_{tu}.*

$$R(u,i) = \sum_{t \in T} a_{tu} \times r_t(u,i)$$

At this point, the strategy used to combine multiple predictions into a hybrid prediction has been described. The next section explains the role of the hybrid prediction strategy in generating recommendations.

4.5 Recommendation pipeline

The aim of the recommender system is to suggest suitable tips. The prediction algorithms that have been described in the previous sections are important elements of our recommendation method, since they estimate the preferences of users. However, predicted utilities by themselves are not yet recommendations. The present section explains the recommendation pipeline. This pipeline contains all steps that are performed to generate a recommendation (see Figure 4.1).

4.5.1 Generating candidates

The first step in generating a recommendation is calculating predictions for all items. At the end of this step, the number of candidate recommendations equals the number of available items and each candidate has a predicted utility ranging from -1 to 1. Predictions are calculated using the following procedure (see Figure 4.2). First, the system checks whether the user has already rated the current item. If this is the case, the predicted utility equals the observed rating and no utility will be estimated. If the user has not yet rated the item, the system estimates a utility using the hybrid prediction strategy.

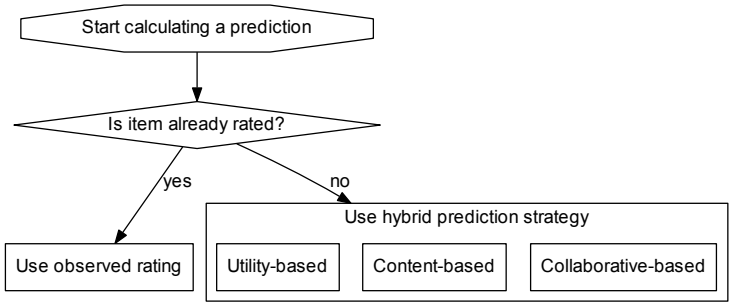


Figure 4.2: The procedure used to estimate the utility of an item. This procedure specifies the first step of the recommendation pipeline (Figure 4.1). Once a utility is estimated, filters can be applied to alter the prediction.

4.5.2 Filtering the candidates

In the second step, multiple filters are applied to alter the list of candidate recommendations. Two types of alterations are distinguished. First, the value of a predicted rating may change. Such filters can, for instance, be used to promote active learning. Second, a filter be used to remove certain candidates from the list of candidate recommendations. This prevents that the candidate will be recommended to the user. The filters that were implemented are explained below. The order in which filters are described reflects the sequence in which they are applied to the candidates.

1. *RecentlyRecommendedFilter*: It is expected that users do not want to perform the same tip from time to time (even if they like the tip). Therefore, candidates that have been recommended in the past five working days are filtered. This filters ensures that it takes at least a week before the same tip can be suggested.

2. *LowCoveredTypeFilter*: Serenedipitous items are items that are both attractive and surprising to the user. Serendipity is important for recommender system, as it helps to reveal unexpressed preferences (Ge et al., 2010; McNee et al., 2006). On the one hand, providing users with unexpected and fortuitous suggestions, may increase their satisfaction about the system. On the other hand, the user's feedback about serenedipitous items contributes to the construction of a reliable user model and may prevent that a prediction algorithm reaches a local optimum.

   The *LowCoveredTypeFilter* was implemented to enhance serendipity. This filter slightly increases the predicted utility of a candidate when the user did not yet receive a recommendation for a tip of the current type. It was decided to increase the utility of candidates for unseen types with a value of 0.2. By using this relatively small increase, we aim to increase the chance that serenedipitous items are suggested and try to prevent that unfortuitous tips are given too much credit.

3. *LowCoverageFilter*: The collaborative-based predictor requires a sufficient amount of ratings per item to estimate reliable predictions. When too little ratings are available the predictor is expected to face the cold start problem which refers to a serious decline of the recommendation quality (Ahn, 2008; Adomavicius & Tuzhilin, 2005).

   The *LowCoverageFilter* filter was implemented to enhance item coverage. This filter slightly increases the predicted utility of an candidate when the current item is rated by less than three users. In this way the filter attempts to help the collaborative-based predictor quickly overcome the cold-star problem. A disadvantage of the filter is that it also increases the chance that items which are probably disliked are suggested to a user. However, since the predicted utility of low covered items are only increased by 0.1, the filter tries to prevent that unfortuitous tips are given too much credit.

4.5.3 Selecting a candidate

After this second step, a candidate is selected from the list of filtered candidates. The implemented recommendation method uses a top-N recommendation approach. This means that the candidate with the highest predicted utility is supposed to be the most appealing to the user. Hence, this item will be selected. In the event that two or more candidates share the highest predicted utility, one of these candidates will be selected randomly. Finally, the selected candidate is recommended to the user.

4.6 Discussion

This chapter explained the recommendation method which generates tailored suggestions for tips. The following paragraphs mention two critical notes regarding our implementation.

The content-based predictor constructs user profiles based on the user's preference for different goals, types and the presence of peers. We think that learning the user's preferences for these manually annotated features is legitimate, especially because a multiple regression analysis showed that three of these selected features represent relatively high changes in ratings (see Section 3.3 for more details). However, future studies may want to use data-driven methods to find novel features that capture the preferences of knowledge workers and add these features to the user profiles.

The utility-based predictor is based on a predictive model obtained by a multiple regression analysis. The ratings used to build this model were based on participants' willingness to perform a tip, whereas the utility-based predictor utilizes the model to predict whether participants actually perform a tip. In other words, the setting in which the ratings used to construct the model were obtained slightly differs from the setting in which the utility-based predictor is applied. The model used by the utility-based predictor is thus out-of-sample which may negatively influence the predictor's accuracy.

The remainder of this thesis focuses on evaluating the recommendation method. The next chapter describes several experiments that were conducted to evaluate an optimize the method in an offline setting. Later, Chapter 6 covers a user study that examines the effectiveness of the recommendation method in an online setting.

Chapter 5

Optimization of the recommendation method: offline experiments

This chapter describes several offline experiments that were used to evaluate and optimize the recommendation method that was explained in the previous chapter. We start with explaining the methodology of the experiments (Section 5.1). Next, this chapter describes analyses that focus on finding those parameter settings that yield promising predicting accuracies (Section 5.2). By tuning the recommendation method in an offline setting, we aim to increase the chance that suitable recommendations are generated during the user study. Subsequently, this chapter describes a comparison between the prediction algorithms and the hybrid prediction strategy (Section 5.3). It is expected that the hybrid prediction strategy outperforms all other prediction algorithms in terms of accuracy, since it is designed to combine the strengths of these individual algorithms. Finally, this chapter reflects on the offline experiments that were conducted (Section 5.4).

5.1 Method

This section describes the methodology of the offline experiments that were conducted. It provides information about the dataset, evaluation metrics and baseline that were used.

5.1.1 Dataset and pre-processing

The present subsection explains the characteristics of the data that was used to train and test the prediction algorithms. Both the train and test data were derived from the ratings that were obtained during the pilot study (see Section 3.3). Participants rated 54 tips/items using a 5-point Likert scale in this study. Given that 26 people participated in the study, the obtained dataset contains 26 ratings for all 54 items. This sums up to a total of 1404 ratings. More formally, the dataset contains an observed rating r_{ui} for each possible combination of participant (u) and item (i). A combination of a participant and item is also referred to as an user-item pair (u, i).

Positive, neutral and negative ratings were distinguished while training the prediction algorithms, because participants of the user study will be able to give these ratings to recom-

	# ratings per user
Positive ratings	24.23 ($SD = 6.98$)
Neutral ratings	14.65 ($SD = 8.56$)
Negative ratings	15.12 ($SD = 6.11$)

Table 5.1: The average number of positive, neutral and negative ratings per user in the dataset. Averages were computed over 26 participants. The source of the data is the pilot study that was described in Chapter 3.

Size of the training set	# ratings per user	# ratings in total
10%	5	130
20%	11	286
40%	21	546
60%	32	832
80%	43	1118
100%	54	1404

Table 5.2: The six possible sizes of the training set. The predictors were trained using different amounts of training data to better simulate an online recommender situation. For each possible size, the number of ratings per user and in total is displayed. Note that the content-based predictor was trained per user on (at most) 54 ratings, whereas the collaborative-based predictor was trained on ratings of all users (maximum of 1404 ratings).

mended tips. For that purpose, all ratings in the original dataset were transposed to positive, neutral and negative ratings. Ratings with a Likert score above 3 were seen as positive ($r_{ui} = 1$); ratings below 3 were seen as negative ($r_{ui} = -1$); and all ratings with a value of 3 were seen as neutral ($r_{ui} = 0$). Table 5.1 shows the distribution of ratings in the dataset.

In situations where a recommender system is used online, users have often only rated a few items. To make the offline experiments closer to a real online situation, predictors were trained using subsets containing respectively 10%, 20%, 40%, 60%, 80% and 100% of the ratings given by a user. Ratings were randomly picked and added to the subsets. Table 5.2 shows the total number of ratings and the number of ratings per user for different sizes of the training set.

5.1.2 Validation and evaluation metrics

All experiments were performed using leave-one-out cross-validation to ensure a reliable investigation. This means that while training the model, the item of which the predicted utility is being calculated is left out of the training set. Furthermore, each experiment consisted of 3 complete runs of a predictor. During each run, a rating was predicted for each possible user-item pair.

At the end of each experiment, the predictor's accuracy was estimated using four evaluation metrics. Three of these metrics focus on the accuracy of a predictor in classifying user-item pairs. For that purpose, a positive and negative class was distinguished, since participants of the user study will either follow or reject a recommended tip. Ratings with a Likert score of at least 3 were added to class of positives; and ratings below 3 were classified as negative.

The following paragraphs explain the four evaluation metrics in detail. Note that the results described in this chapter are averages over all 26 participants. It will be explicitly stated when the result of a single user is presented.

Root-mean-square error (RMSE) The RMSE measures the distance between predicted ratings (r_{ui}) and the values actually observed ($\hat{r}_{ui}$). The RMSE reaches its best value at 0 and worst score at 1. The RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{\tau} \sum_{(u,i) \in \tau} (\hat{r}_{ui} - r_{ui})^2}$$

Recall The recall measures how good a predictor is at detecting positives. It is defined as $tp/(tp + fn)$, where tp represents the number of items that were correctly classified as being positive (true positive) and fn denotes the number of items that were classified as being negative, but in fact were positive (false negative). The recall reaches its best value at 1 and worst score at 0. A predictor can cheat and maximize this measure by predicting only positive ratings.

Precision This measures indicates how many of the positively classified items were relevant. Precision is defined as $tp/(tp + fp)$ where tp denotes the number of true positives and fp represents the number of items classified as being positive, but in fact were negative (false positive). The precision reaches its best value at 1 and worst score at 0. A predictor can cheat and maximize this measure by predicting only one positive rating on the prediction it's most confident in. Although having both a high recall or high precision would be preferable, an effort to improve the performance of either the recall or precision often causes the performance of the other to drop. This effect is often referred to as the precision-recall trade-off (Buckland & Gey, 1994).

Specificity The specificity measures how good a predictor is at detecting negatives. It is defined as $tn/(tn + fp)$, where tn represents the number of items that were correctly classified as being negative (true negative) and fp represents the number of false positives. The specificity reaches its best value at 1 and worst score at 0. A predictor can cheat and maximize this measure by predicting only negative ratings.

5.1.3 Baseline

A baseline score was estimated for each evaluation metric, which provides a basis for comparison. Several methods can be used for generating a RMSE baseline (Ekstrand et al.,

2011). The simplest baseline is to predict the average rating over all ratings in the system. However, in this project a lazy predictor was implemented to obtain a more enhanced RMSE baseline. This lazy predictor sets the prediction of each item to the mean of all observed ratings by the current user. Subsequently, the RMSE was calculated using the default procedure (i.e., calculating the error between predicted and observed rating). This yielded a RMSE baseline of 0.41.

The recall, precision and specificity relate to classifier performance. A lazy classifier which predicts the majority class is often used to obtain a baseline for these metrics. The following baseline scores were obtained using a lazy classifier which always predicts the positive class (i.e., the majority class in the present dataset):

Recall = 1.00 All predictions are classified as positive. Hence, all positives are detected, which results in an optimal recall.

Precision = 0.72 Since all items are classified as positive, the percentage of correctly classified positives equals the percentage of positives in the dataset.

Specificity = 0.00 All predictions are classified as positive. Hence, none of the negatives are detected.

5.2 Optimizing the predictors

This section describes several exploratory data analyses that were performed to select a promising parameter setting for each prediction algorithm. By tuning the prediction algorithms in an offline setting, we aim to increase the chance that suitable recommendations are generated during the user study. The present section is split in two parts. First, optimal parameters for the collaborative-based predictor are defined (Section 5.2.1). Next, the section describes the optimization of the content-based predictor (Section 5.2.2). No analysis of the utility-based predictor was conducted, because this predictor has no parameters that can be tuned.

5.2.1 Collaborative-based predictor

An exploratory data analysis was performed to optimize the content-based predictor (Section 4.1 explains the predictor in detail). The collaborative-based predictor itself has two parameters that can be varied. Namely, the neighborhood size and the minimum number of items two users should have both rated without having to penalize the correlation between them. Moreover, the amount of data that is used for training the predictor's model can be varied. The predictor's accuracy was examined for many possible combinations of these three parameters. Table 5.3 describes all parameter settings that were tested. The following paragraphs discuss the influence of different parameter settings on the accuracy of the collaborative-based predictor. These results were obtained when the predictor was trained

Parameter	Tested values	Description
subsetSize	10%, 20%, 40%, 60%, 80%, 100%	The size of the training set. The prediction of a user-item pair (u, i) is calculated using the observed ratings given to item i by neighbors of user u.
numNeighbours	1, 2, 3, 4 ,5, 6, 7, 8, 9, 10	The size of the users' neighborhoods.
bothRatedTreshhold	1, 2, 3, 4 ,5, 6, 7, 8, 9, 10	The number of items two users should both have rated. The correlation between two users is penalized if the number of co-rated rated is below the limit.

Table 5.3: The collaborative-based predictor was tested for each combination of these parameter values. This resulted in 60 x 10 x 10 = 600 combinations that were tested.

using 40% of the available ratings per user. This size was chosen because it reasonably simulates a real-world situation in which users have often only rated a few items.[1]

Influence of different parameter settings

Figure 5.1 shows the RMSE's that were obtained when training the collaborative-based predictor using 40% of the available ratings per users. A colored dot is plotted for each parameter setting that was tested. The color of the dot reflects the RMSE that was obtained in that setting. One can observe that higher *bothRatedTreshhold* values yield more bluish dots, which represent better RMSE's. In other words, penalizing the correlation if two users have only co-rated a small portion of items seems to result in better accuracies. This shows that the more data points that we have to compare the preferences of two users, the more we can trust that the computed correlation is representative of the true correlation between the two users (Herlocker et al., 2002).

In contrast to the parameter *bothRatedTreshhold*, the parameter *numNeighbours* only has a small influence on the predictor's accuracy. This can be observed in Figure 5.1 due to relatively small color differences for different values of *numNeighbours* when keeping the other parameter constant.

Optimal parameter setting

In conclusion, better results were obtained when penalizing the correlation of two users that have only co-rated a small portion of items. Therefore, a value of 8 is chosen as optimal

[1]Results that were obtained when the predictor was trained using different amounts of training data are shown in Appendix B. The results in this appendix indicate that the predictor is unable to produce accurate ratings in any setting when only a few ratings are available. It also shows that the differences between settings become smaller when almost all ratings are available for training. Moreover, Appendix B shows the results for other evaluation metrics than the RMSE. In this section we merely focus on the influence of different parameter settings on the RMSE, because similar results were obtained for all other evaluation metrics.

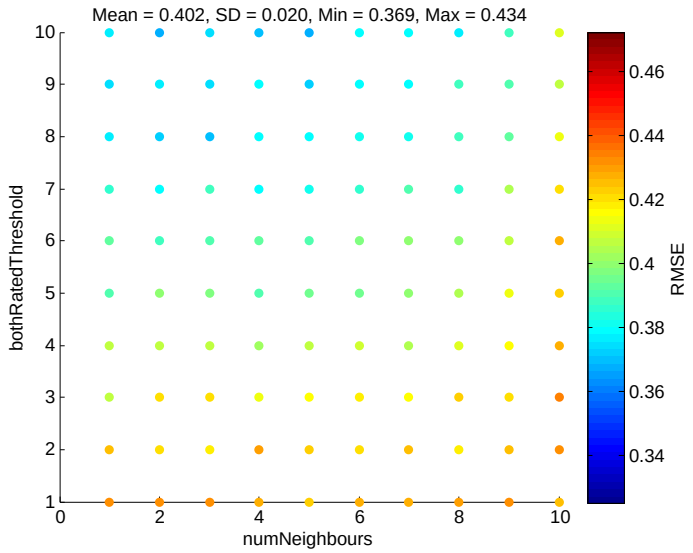


Figure 5.1: RMSE's that were obtained when training the collaborative-based predictor on 40% of the available ratings per users. The x-axis shows the possible values of the parameter *numNeighbours*, whereas the y-axis contains all possible values of parameter *bothRatedTreshhold*. For each combination of the two parameters a colored dot is plotted. The color of the dot reflects the score that was obtained for that combination. Note that lower RMSE's represent better accuracies.

Parameter	Tested values	Description
subsetSize	10%, 20%, 40%, 60%, 80%, 100%	The size of the training set. The predictor's model only depends on rating given by the user himself and is not influenced by ratings of others.
beta	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0	The relative importance of positive and negative ratings in shaping the profile vector of a user.

Table 5.4: The content-based predictor was tested for each combination of these parameter values. This resulted in 60 x 10 = 60 combinations that were tested.

value of parameter *bothRatedTreshhold*. Given the relatively small effect of parameter *numNeighbours*, the optimal neighborhood size is set to 3. Table 5.5 shows the predictor's score on all evaluation metrics when using this optimal parameter setting.

5.2.2 Content-based predictor

An exploratory data analysis was performed to optimize the content-based predictor (Section 4.2 explains the predictor in detail). The content-based predictor itself has one parameter that can be varied. Namely, beta which controls relative importance of positive

and negative ratings in shaping the profile vector of a user. Moreover, the amount of data that is used for training the predictor's model can be varied. The predictor's accuracy was examined for many possible combinations of these two parameters. Table 5.4 describes all parameter settings that were tested. The following paragraphs discuss the influence of different parameter settings on the accuracy of the content-based predictor.

Influence of parameter beta

Figure 5.2 depicts the results that were obtained while optimizing the content-based predictor. Each graph focuses on a single evaluation metric. The x-axis represents the amount of ratings per user that was used to train the predictor's model (in percentages). Each plotted line represents a parameter setting. The obtained scores are plotted on the y-axis. The results suggest that higher *beta* values are desired when one wants the predictor to detect positives well. If the goal is to detect negatives, however, one should chose a low *beta* value. This observation can be observed via a high recall for higher *beta* values (Figure 5.2b) and a high specificity for lower *beta* values (Figure 5.2d).

Although parameter *beta* thus seems strongly related to the ability of detecting either positives or negatives, choosing extreme *beta* values seems to make the predictor lazy. Figure 5.2 shows that the recall approaches 1, whereas the specificity approaches 0 when beta is set to 1. This means that the predictor detects almost all positives, but is unable to detect negatively rated items. This observation suggests that the predictor simply classifies all items as positive. On the other hand, extremely low beta values results in a predictor that simply classifies most items as negative. The recall approaches 0.2 and the specificity approaches 0.95 if beta is set to 0.

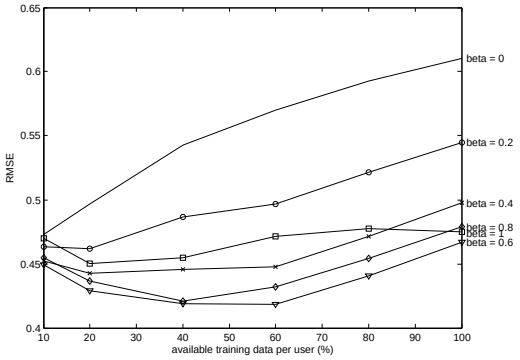
Optimal parameter setting

In conclusion, using extreme *beta* values yielded one-sided results. Using a relatively high value resulted in the best accuracy, a promising recall and a proper precision. Therefore, the value 0.8 is picked as optimal value of parameter *beta*. This value is also often used in other studies (Manning et al., 2008). Table 5.5 shows the predictor's score on all evaluation metrics when using this optimal parameter setting.

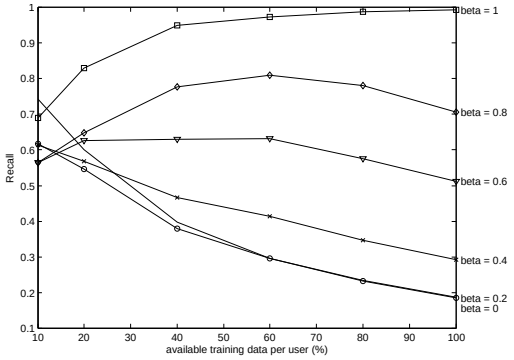
5.3 Comparing the prediction approaches

At this point, optimal parameter settings have been selected for the collaborative-based and content-based predictor. No optimization study was conducted for the utility-based predictor, because this predictor has no parameters that can be tuned.

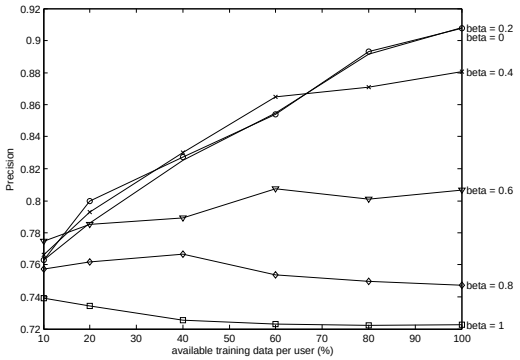
The three prediction algorithms and the hybrid prediction strategy are compared with each other in this section. We start with describing a comparison between the three predictors in terms of prediction accuracy (Section 5.3.1). Later, this sections focuses on evaluating the hybrid prediction strategy (Section 5.3.2).



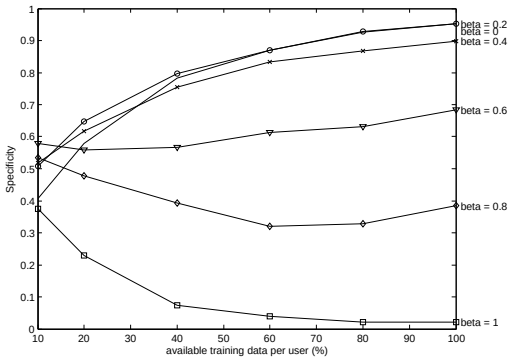
(a) Better RMSE's were found for higher *beta* values. This suggests that more accurate predictions are made when constructing a user profile based on positive ratings. The baseline RMSE is 0.41. Hence, the accuracy of the predictor is worse than the baseline in all settings.



(b) The recall differs enormously among parameter settings. If *beta* is set to 1, the recall approaches 1 which indicates that the predictor detects all positives. The recall is much lower, when lower *beta* values are used. The baseline recall is 1.00.



(c) Better precision is obtained when *beta* is low. This means that the amount of positively classified items that were relevant slightly increases if a user profile is more based on negative ratings. The baseline precision is 0.72.



(d) The specificity increases when *beta* is low, which indicates that the algorithm better detects negatives in these settings. The baseline specificity is 0.00.

Figure 5.2: The results obtained while optimizing the content-based predictor. Each graph focuses on a single evaluation metric. The x-axis represents the amount of ratings per user that was used to train the predictor's model (in percentages). Each plotted line represents a parameter setting. The obtained scores are plotted on the y-axis.

Evaluation metric	Collaborative-based	Content-based	Utility-based	Baseline
RMSE	0.42	0.42	0.38	0.41
Recall	0.84	0.78	0.77	1.00
Precision	0.81	0.77	0.83	0.72
Specificity	0.51	0.39	0.38	0.00

Table 5.5: The performance of the three predictors when using their optimal parameter setting. The baseline scores are also displayed to have a basis for comparison. Predictors were trained on 40% of the ratings.

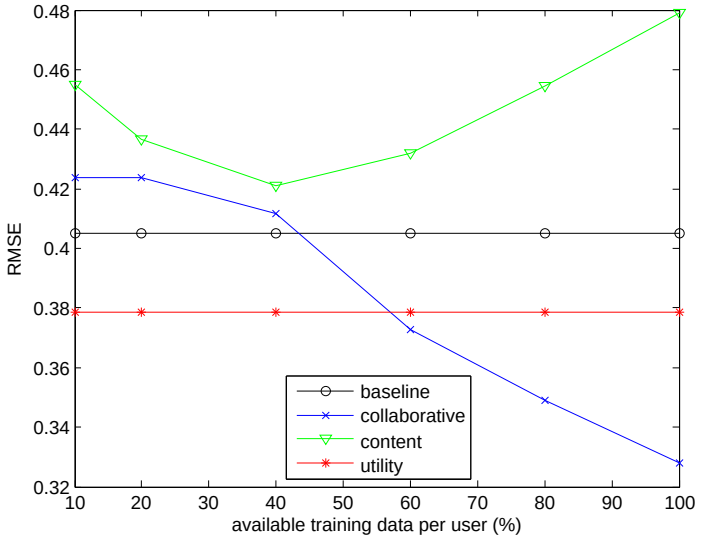


Figure 5.3: A comparison between the RMSE baseline and the RMSE's that were obtained by the collaborative-based, content-based, and utility-based predictor. The x-axis represents the amount of ratings per user that were available for training. The RMSE's can be found on the y-axis. Note that lower RMSE's represent better accuracy.

5.3.1 Comparing the accuracy of the three predictors

This subsection describes a comparison between the collaborative-based, content-based and utility-based predictor in terms of prediction accuracy. For that purpose, the three prediction algorithms estimated a rating for each user-item pair using their optimal parameter setting. This procedure was repeated six times. The difference between these runs was the amount of available training data. The RMSE was calculated for each predictor using the procedure explained in Section 5.1.1.

Figure 5.3 shows the results of this comparison between the three predictors. The figure depicts four lines. The black line represents the baseline RMSE, whereas the three colored lines represent the RMSE that was obtained by one of the predictors. The results that were previously described in this chapter when optimizing the collaborative-based and content-

based predictor are also reflected in this figure. For instance, it can be observed that the collaborative-based predictor faces the cold-start problem, because the accuracy of this predictor is poor when little training data is available and improves substantially when more ratings become available. Furthermore, the figure shows that the accuracy of the content-based predictor initially improves (visible due to a lower RMSE). If more than 40% of the ratings per user are available for training, the accuracy decreases. In this perspective, the two predictors seem to complement each other. By giving more weight to the content-based predictor if little data is available and increasing the influence of the collaborative-based predictor on the final prediction when more ratings are observed, the overall accuracy might always be guaranteed.

One can also observe in Figure 5.3 that adding more training data does not influence the accuracy of the utility-based predictor. This result can be explained by the fact that this predictor is static and does not exploit ratings to adjust its model. Moreover, the utility-based predictor was in fact already trained on the complete dataset during the regression analysis described in Section 3.3. Although its accuracy is relatively poor, this predictor does not depend on the user's ratings and can thus be useful for providing predictions for new users.

The previous paragraphs showed that each predictor can play a valuable role in the prediction process. This strengthens the decision of generating predictions using a hybrid prediction strategy, which combines the three prediction algorithms. An evaluation of this hybrid prediction strategy is described in the following subsection.

5.3.2 Evaluating the hybrid prediction strategy

This subsection describes a comparison between the three predictors and the hybrid prediction strategy in terms of classifier performance. The hybrid prediction strategy combines the utilities that are estimated by the three optimized predictors using weights. These weights are determined per-user based a predictor's error rate and control the influence of a predictor on the hybrid prediction (see Section 4.4 for a detailed explanation). It is expected that the hybrid prediction strategy outperforms all prediction algorithms, as it is designed to combine the strengths of these algorithms.

For the purpose of the comparison, all prediction approaches first estimated for each user-item pair whether the user would follow or reject the item. The are four prediction approaches, namely the three predictors and the hybrid prediction strategy. Subsequently, the number of successfully classified items were calculated per user for each prediction approach. Finally, the best and second best approach was identified for each user. The best prediction approach is defined as the approach that classifies the largest number of items correctly. This procure was repeated six times. The difference between these runs was the amount of available training data.

Table 5.6 shows the results of this comparison between the four prediction approaches. This table displays how frequently the hybrid prediction strategy was the best or second best approach for different sizes of the training set. One can observe that the hybrid prediction strategy is the best approach in almost all cases. Results also show that there is a minority of users for which the hybrid prediction strategy is not the best prediction approach. However,

Available training data	# best	# 2nd best	# neither best or 2nd best
10%	12	14	-
20%	16	10	-
40%	20	4	2
60%	15	10	1
80%	15	10	1
100%	18	7	1

Table 5.6: The number of cases for which the hybrid prediction strategy was the best or second best prediction approach in terms of classifying the largest number of items correctly.

in most of these cases the hybrid prediction strategy is still the second best approach. These promising results strengthen the decision of using a hybrid prediction strategy.

5.4 Discussion

In this chapter we described several experiments that were conducted to evaluate and optimize the recommendation method. Results indicate that the hybrid prediction strategy nearly always outperforms the individual predictors when it comes to predicting whether a user would perform or reject a tip. Results also show that especially the collaborative-based predictor yielded good predicting accuracies. The accuracy of the utility-based predictor was mediocre, whereas the content-based predictor scores at best as good as the baseline on most evaluation metrics. The following paragraph provides three possible explanations for this unexpected result.

First of all, it could be that the set of item features does not capture the users' preferences well enough. If this is indeed the case, the content-based predictor was unable to discriminate the liked and disliked items. Second, given that a profile vector is shaped by moving towards or away from features based on observed ratings, the performance of the content-based predictor also depends on the consistency of a user's rating behavior. It could be that the content-based predictor was unable to make accurate predictions, because users were inconsistent and often rated items sharing the same feature differently. A different explanation could be that the amount of training data was too small. The content-based predictor was trained per user on (at most) 54 ratings, whereas the collaborative-based predictor was trained on ratings of all users (maximum of 1404 ratings). It might be that the accuracy of the predictor increases when more training data becomes available. However, in a real online recommender situation users have often only rated a few items. A predictor should thus be able to produce accurate ratings very quickly.

All prediction algorithms were evaluated in this chapter. However, we did not evaluate the implemented filters. These filters are explained in Section 4.5 and, for instance, used to prevent that recently recommended tips are suggested to a user. The impact of these filters on the recommendation method was not evaluated due to time constraints.

The next chapter describes a user study that was conducted to evaluate the recommendation method in an online setting.

Chapter 6

Evaluation of the recommendation method: a user study

The recommendation method described in the previous chapters was evaluated in a user study. For that purpose, an Android app was implemented that provided recommendations. The study has two main focuses. First, it investigates whether tailored recommendations for tips have a higher chance of being followed-up compared to recommendations that are not adapted to the user's preferences. Second, we are interested in how knowledge workers experience an e-coach that automatically suggests tips that can improve well-being at work. This chapter is structured as follows. First, the methodology of study is explained (Section 6.1). Later, we describe the analyses that were performed to answer the research questions and their results (Section 6.2). Finally, the obtained results are discussed (Section 6.3).

6.1 Method

This section describes the methodology of the user study that was conducted. It provides information about the participants, the materials that were used and the procedure of the experiment.

6.1.1 Participants

Participants were approached through the personal network of the researcher. 16 men and 19 women ranging in age from 24 to 63 years volunteered to participate in the study ($M = 38.9$ years, $SD = 11.3$ years). All participants were knowledge workers. The average working week of the participants was 37.9 hours ($SD = 5.8$). On average, the participants worked using a computer for 29.0 hours a week ($SD = 6.0$). There were 16 participants randomly assigned to the test condition. The remainder was assigned to the baseline condition (19 participants).

(a) The home screen was displayed when participants opened the app. It allowed them to manually request a tip and to lookup previously recommended tips.

(b) This screen displayed a single suggestion. It was shown when participants clicked on a notification or selected a tip from the list of previous recommendations. Participants rated tips using this screen. If a participant indicated that he did not perform a tip, a dialog appeared asking the participant to motivate his decision.

(c) This screen displayed the list of previous recommendations. The icon to the left of a tip indicates whether the tip was followed. The text beneath a tip shows the time of rating. Finally, the text to the right depicts when the tip was suggested.

Figure 6.1: The NiceWork app provided participants with automatic notifications for well-being tips. On each notification the participant's smartphone vibrated. The app allowed participants to view and rate recommended tips.

6.1.2 Materials

The NiceWork app was implemented to provide the participants with recommendations (see Figure 6.1). The NiceWork app for Android had three main objectives. First, the app notified participants about new recommendations. Second, it allowed participants to indicate whether they performed a recommended tip. Finally, the app enabled participants to get an overview of previously suggested tips. All tips were said to improve well-being at work and are explained in Chapter 3. Recommendations were generated using the recommendation method described in Chapter 4. A detailed explanation of the NiceWork app, including its implementation, can be found in Appendix D.

Furthermore, two questionnaires were designed. The first questionnaire was used to asses the working days, stage of change and workload of the participants. The 'stage of change' shows how far an individual proceeded in making a sustainable behavior change

Followed up	Observed rating	Motivation
Yes	Positive	
No	Neutral	The timing of the tip was not right.
No	Neutral	I was not at work when I received the tip.
No	Negative	I do not want to receive this tip again.
No	Negative	I already work using the suggested method.
No	Negative	Performing the tip takes too much time.
No	Negative	I do not understand the tip.

Table 6.1: The six answer options and their associated ratings. If participants rejected a tip, they were able to pick the most suitable motivation from a list of six possible motivations.

(Prochaska & DiClemente, 2005). This questionnaire was integrated with the NiceWork app and filled out when the app was opened for the first time. At the end of the experiment, participants filled out a second questionnaire on a website. This questionnaire contained questions about the participants' experience with the e-coach, their working situation, as well as general questions about age and gender. Both questionnaires can be found in Appendix C.

6.1.3 Procedure

The experiment consisted of a preparatory, testing and closing stage. In the preparatory stage, participants received an e-mail in which they were asked to install the NiceWork app using the Google Play Store. Furthermore, participants were asked to answer some questions about their working days, stage of change and workload. Installing the app and answering these questions took about five minutes. When a participant fulfilled both preparatory steps, he was ready to proceed to the testing stage.

The testing stage took two weeks. All participants started and ended with this stage at the same time. During the testing stage, participants automatically received three recommendations for a tip per working day. Recommendation were send at random moments between 10-11 AM, 13-14 PM and 15-16 PM. These time slots were chosen, because they ensured an even spread of the suggestions over the workday of most knowledge workers. On each recommendation, the smartphone vibrated to notify the participant about the new tip. Participants were asked to indicate whether they performed the suggested tip. If a participant rejected a tip, he was asked to motivate his decision. Participants were able chose the most relevant motivation out of a list of six possible motivations (see Table 6.1).

The method that was used to generate recommendations depended on the participant's experimental condition. Participants in the test condition received recommendations using the described recommendation method (see Chapter 4). This means that the recommendations were made by matching tips with the profile of the participant. Observed ratings were exploited to improve the participant's profile. Participants in the baseline condition, however, received randomized recommendations. This means that tips were not matched to the

participant's profile and that the system did not learn from the user's ratings. Participants were randomly assigned to either the test or baseline condition.

After two weeks of testing, participants entered the closing stage. Participants were asked to fill out a questionnaire on a website. This questionnaire contained questions about the participants' experience with the system, their working situation, as well as general questions about age and gender. Answering these questions took about 10-15 minutes.

Note that the procedure described above was not pre-tested.

6.2 Results

The analyses that were performed to answer the research questions and their results are described in this section. First, Section 6.2.1 focuses on whether tailoring increases the chance that recommended tips are followed-up. Second, the analyses and results that relate to how knowledge-workers experience an e-coach that automatically recommends tips are described in Section 6.2.2.

6.2.1 Recommendation method

In this subsection, the analyses and results that relate to the recommendation method are presented. We start with describing a random-effects model. This model was generated to investigate which variables influence the chance of following-up a suggested tip. Second, this subsection focuses on the diversity of recommended tips. Here we describe an analysis of the percentage of uniquely recommended tips. Finally, the motivations for not performing a tip are discussed.

Predictive model A logistic random-intercept model was generated to investigate which variables predict the chance of following-up a recommended tip. This chance is also referred to as $P(followUp = 1|x)$. The independent variable $testCondition$ was added to the model to investigate whether the chance of performing a tip differed between the baseline and test condition. Moreover, the independent variable day and interaction variable $day \times testCondition$ were added. These variables were used to assess whether the chance of performing a tip changes during the course of the experiment and whether this change differs between the two conditions. Finally, by including a random intercept we assume that there is a random heterogeneity in the participants' attitude towards performing tips that persists throughout the entire duration of the study.

The fixed effects that were obtained by the model are shown in Table 6.2. The estimate column displays the estimated influence of a variable on the the log odds of $P(followUp = 1|x)$. Each estimate has an associated standard deviation (SD) indicating to what extent the estimate varies across different samples. The standard deviation is used to determine whether an estimated influence differs significantly from zero. In addition to this table, Figure 6.2 plots the chance of performing a recommended tip against the day of the experiment. This figure contains probabilities instead of log odds and may thus be easier to interpret.

We can conclude that the chance of following-up a recommended tip given the obtained model is $invlogit^{-1}(0.16) = 0.55$ for the baseline and $invlogit^{-1}(0.16 + 0.60) = 0.67$

Variable	Estimate	SD	Z	P
$Intercept$	0.16	0.20	0.77	0.44
$testCondition$	0.60	0.31	1.92	0.05
day	0.03	0.03	1.00	0.32
$day \times testCondition$	-0.08	0.05	-1.56	0.12

Table 6.2: The fixed effects that were obtained by a logistic random-intercept model that was generated to investigate which variables predict $P(followUp = 1|x)$. The estimate column displays how the model predicts the influence of a variable on the the log odds of $P(followUp = 1|x)$. The standard deviation (SD) indicates to what extent the estimate varies across different samples and is used to determine the significance of each variable (column Z and P).

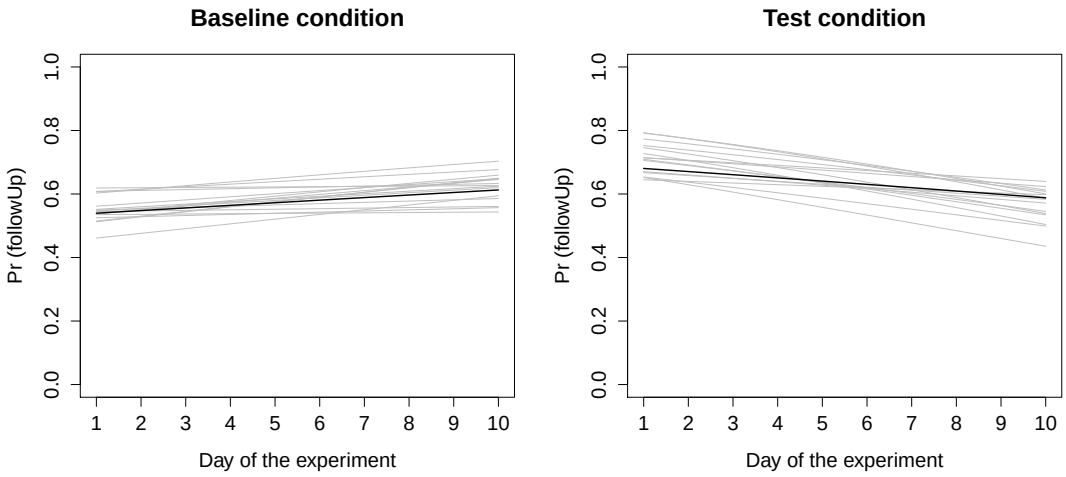


Figure 6.2: The probability of following-up a recommended tip against the day of the experiment for both the baseline and test condition. These plots were constructed using the fixed effects that were estimated by a random-intercept model. On each plot, the solid line shows the fitted regression model and the light lines depict 15 simulations which represent uncertainty about the model. One can observe that the estimated chance of performing a tip is higher in the test condition than in the baseline condition at the start of the experiment. The model predicts that the chance of performing a tip changes during the course of the experiment in both conditions. However, the substantial variability in grey lines indicate these changes are quite uncertain. These plots do not display the random-intercept, which implies that participants have a baseline chance of performing a tip within the range of 0.30 and 0.76 at a 95% confidence interval.

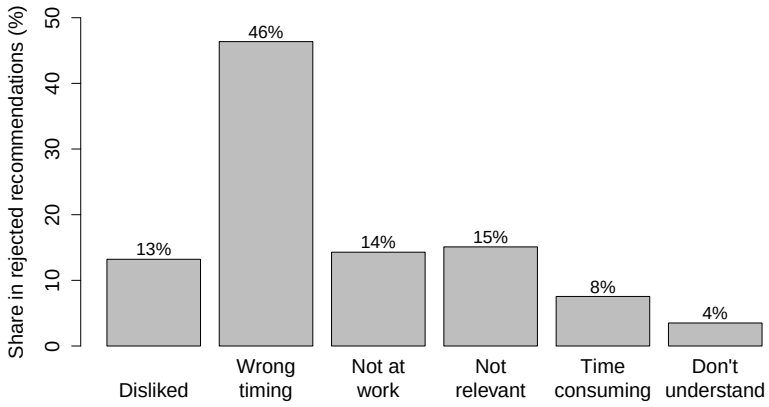


Figure 6.3: Motivations for not following-up a recommended tip. Each bar represents the share of a motivation in the total set of rejected recommendations. One can observe that most tips were rejected, because the moment of recommendation was not right.

for the test condition at the beginning of the experiment. This difference between the two conditions is in line with our hypothesis, but is just not significant, $p = 0.05$. In contrast to our expectations, the log odds of $P(followUp = 1|x)$ decrease approximately linearly with 0.05 per day in the test condition, $0.03 - 0.08$. Hence, the model estimates that the chance of performing a tip is only 0.58 for the test condition at the last day of the experiment, $invlogit^{-1}(0.16 + 0.60 + 0.03 * 10 - 0.08 * 10)$. The log odds of $P(followUp = 1|x)$, however, increase approximately linearly with 0.03 per day in the baseline group. This means that model estimates the chance of performing a tip is 0.62 for the baseline condition at the last day of the experiment, $invlogit^{-1}(0.16+0.60+0.03*10)$. We have to be careful with these latter two results though, because the associated estimates vary substantially across samples and, hence, these results are far from significant. Finally, the estimated variance of the random intercept is 0.27. This implies that there is substantial variability in the attitude towards performing tips among participants, since approximately 95% of the participants have a baseline chance of performing a tip within the range of 0.30 and 0.76.

Diversity The *unique tips rate* of a participant shows the diversity in recommended tips. This measure is defined as the number of uniquely recommended tips divided by the total number of recommendations. A rate of 1 denotes that all recommendations covered a unique tip, whereas a rate that approaches 0 indicates that a single tip was recommended every time. On average, the diversity of tips is lower in the test condition ($M = 0.67, SD = 0.14$) than in the baseline condition ($M = 0.77, SD = 0.11$). An independent-samples t-test showed that this difference in the unique tips rate was significant, $t(30.71) = 2.40, p < 0.05$; furthermore, it did represent a large-sized effect $r = .72$.

Motivations for not following up a tip Participants motivated their decision when they did not perform a tip. Figure 6.3 shows how frequently each motivation was chosen. It

turned out that a tip was often not performed, because the timing of the recommendation was inappropriate; either because the moment of recommendation was not good or because the participant was not at work.

6.2.2 User experience

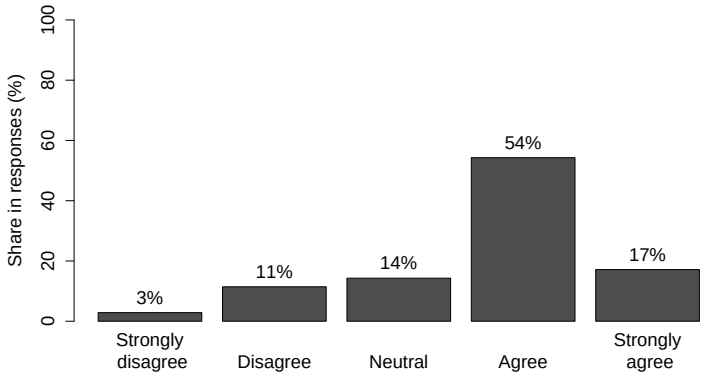
The present subsection starts with describing how participants evaluated the usability of the implemented e-coach. Second, the quality of the tips is discussed. Third, it is evaluated whether participants liked to automatically receive notifications for recommendations. Finally, it is described how participants evaluated the amount of tips that were suggested daily.

Usability Participants answered three 5-point Likert items that assessed whether the user interface of the e-coach was clear, attractive and intuitive. The *usability* scale was defined as the average of these items and had a high reliability, Cronbach's $\alpha = .82$. The mean usability score was 4.10 ($SD = 0.79$), which indicates that on average the participants were satisfied with the usability of the e-coach. The baseline ($M = 4.32$, $SD = 0.41$) and test condition ($M = 3.85$, $SD = 1.05$) did not differ significantly, $t(18.82) = 1.66, p > 0.05$.

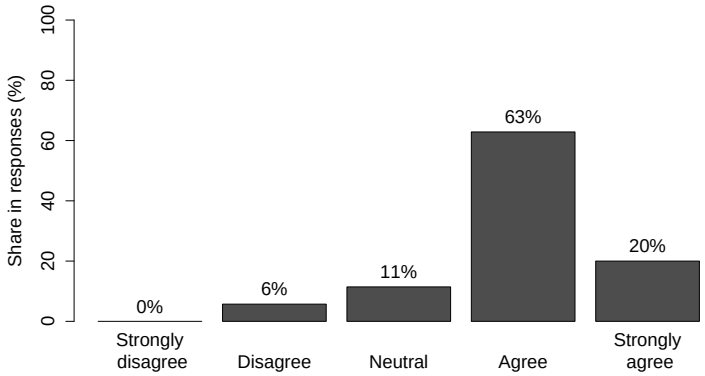
Quality of tips Participants answered four 5-point Likert items that assessed whether the recommended tips were interesting, useful, diverse and similar. The *quality of tips* scale was defined as the average of these items and had a high reliability, Cronbach's $\alpha = .76$. The mean quality of tips score was 3.40 ($SD = 0.70$), which indicates that on average the participants were content with the quality of the tips. The baseline ($M = 3.50$, $SD = 0.68$) and test condition ($M = 3.28$, $SD = 0.73$) did not differ significantly, $t(31.15) = 0.91, p > 0.05$.

Automatic notifications A 5-point Likert item assessed whether participants agreed with the proposition that receiving automatic notifications for recommended tips is pleasant. This item received a median score of 3.97 ($SD = 0.75$), which indicates that on average participants liked to receive notifications. Figure 6.4a shows how frequently each answer option was chosen. The baseline ($M = 4.05$, $SD = 0.71$) and test condition ($M = 3.88$, $SD = 0.81$) did not differ significantly, $t(30.13) = 0.69, p > 0.05$.

Amount of recommendations A 5-point Likert item assessed whether participants agreed with the proposition that the right amount of tips was recommended everyday. This item received a mean score of 3.71 ($SD = 0.99$), which indicates that on average the participants agreed that three tips per working day is the right amount of recommendations. Figure 6.4b shows how frequently each answer option was chosen. The baseline ($M = 3.58$, $SD = 0.96$) and test condition ($M = 3.88$, $SD = 1.02$) did not differ significantly, $t(30.13) = 0.69, p > 0.05$.



(a) Automatic notifications appreciated.



(b) Right amount of recommendations.

Figure 6.4: Results of two Likert-items related to the usability of the e-coach.

6.3 Discussion

The results show that the chance of performing a recommended tip is higher in the test condition than in the baseline condition. However, this difference is not that large and only marginally significant. There are several possible explanations for this relatively small effect and lack of significance.

- First, the estimated variance of the random intercept shows that there is substantial variability in the attitude toward performing tips among participants. The effect of the recommendation method becomes relatively small and unreliable due to this individual-level variability. The reliability could be improved by increasing the number of participants. However, assuming that conducting a larger experiment would result in a model with similar estimates, this approach would yield a significant, but still relatively small effect.

- Second, the motivations for rejecting a recommendation may also explain the relatively small effect of the recommendation method. The results show that 60% of the

tips were rejected, because the moment of recommendation was somehow inappropriate. Our recommendation method did not support time awareness and, hence, was not able to filter this majority of rejected tips. Therefore, time awareness seems like a relevant direction that should be explored in future studies when one wants to increase the chance that tips are followed-up.

- Finally, it could be that personal preferences did not play such a big role, since the overall attitude towards tips was relatively high. Given this high attitude, tailoring the content of tips filters only a few irrelevant tips, which in turn makes the effect of a tailored recommendation method smaller. The fact that participants were approached through the personal network of the researcher –and thus probably wanted to please– may explain the high attitude towards tip.

It was also expected that the chance of performing tailored tips increases during the course of the experiment, because our recommendation method exploits observed ratings to generate more suitable recommendations. We did not find evidence of any such effect. The following paragraphs provide three possible explanations.

- In the baseline condition complete randomness of recommended tips is suggested, but since the tips that were previously disliked are filtered upon future recommendations, the random condition does use some knowledge of the participant. It might be that this knowledge is already sufficient and that exploiting ratings to better match tips with the participant's preferences does not result in more suitable recommendations.

- Furthermore, the freedom in tip selection is greatly reduced by matching tips. For example, imagine a participant who disliked one or two exercise tips. It is unlikely that the tailored recommendation method will generate a novel exercise tip for this participant. This decreases the number of tips that can be suggested, which might be an issue especially because our set of tips is relatively small. Results seem to confirm this reasoning, because participants in the test condition more often received a suggested tip twice. It could be that participants did not want to perform the same tip twice within two weeks. Hence, the freedom limitation in selecting tips possibly explains why the chance of performing a tip does not increase in the test condition.

- Finally, our recommendation method utilizes insights that were obtained during a pilot study to generate suitable, but non-tailored suggestions at the start of the experiment. It may be that the advantage of our recommendation method lies in this prior knowledge and that exploiting ratings to better estimate the participant's preferences does not result in more suitable recommendations. Future studies may compare a tailored recommendation method with a baseline method which also utilizes prior knowledge to answer this question.

The present study also evaluated how participants experienced the implemented e-coach. It turned out that on average the participants like to receive automatic recommendations. The relatively high chances of performing a tip in both conditions also highlights the potential of an e-coach that automatically suggest well-being tips to knowledge workers. We have to

keep in mind though that participants voluntary signed up for the study. This means that some curiosity and a positive attitude can be expected.

Our overall conclusion is that recommender systems are a novel and interesting area in the domain of well-being at work and that these systems can be used to tailor the content of suggestions. The study also revealed several interesting research directions that should be explored in future studies, because they can provide relevant insights on the factors that predict whether a tip is performed.

Chapter 7

Conclusions and discussion

The goal of the research described in this thesis was to investigate an e-coach that suggests tips that can help knowledge workers improve their well-being at work. For that purpose, a recommender system was designed which generated tailored suggestions. Our main hypothesis was that tailored tips would be followed-up more often compared to randomized suggestion. We did not find strong evidence for this hypothesis. Although this result was unexpected, various novel insights were revealed. Our findings also disclosed several relevant research directions that should be explored to design more effective e-coaches in the future.

Our results, as well as their implications (including future work) are discussed in this chapter. We start with summarizing the recommendation method and reflecting on the experiments that were used to evaluate and optimize this method in an offline setting (Section 7.1). Later, the results of a user study are discussed (Section 7.2). Here, we also provide three research directions that should be explored to design more effective e-coaches. Finally, this chapter describes our main conclusions (Section 7.3).

7.1 Recommendation method

In this thesis we took a recommender approach to provide knowledge workers with suitable tips. A first step was collecting the tips that could be suggested. Various scientific articles, websites and magazines on well-being at work were reviewed to collect and annotate appropriate tips. This resulted in a set of 54 annotated tips. The tips –and especially their annotation– are an important contribution to science, since other researchers can use them in their studies. For example, researchers may investigate to what extent the tips actually help managing stress. They may also add new tips which eventually could lead to a large database of interventions that help knowledge workers improve their well-being at work.

Subsequently, a pilot study was conducted to investigate the tips and factors preferred by knowledge workers. It turned out that the participants did not agree on which tips they liked. This finding shows that tips can be filtered and, hence, provides motivation for a recommender approach.

The next step was to develop the recommender method that finds suitable tips. Our recommendation method estimates the user's utilities for unseen tips using a hybrid prediction strategy. This hybrid prediction strategy combines the strengths of the following three prediction algorithms. First, the collaborative-based predictor estimates the user's rating for a tip by aggregating observed ratings from other users with a similar rating history. Tips that people with similar preferences liked receive high utilities. Second, the content-based predictor estimates ratings based on the user's preferences. Tips that are similar to the ones a user liked in the past receive high utilities. Finally, the utility-based predictor utilizes a predictive model that was obtained by a multiple regression analysis during the pilot study to estimate ratings.

Several experiments were conducted to evaluate and optimize the predictors in an offline setting. Results show that especially the collaborative-based predictor yielded good predicting accuracies. This suggests that estimating the preferences for tips by aggregating ratings from peers with a similar rating behavior is a promising prediction approach. Even though many effort has been put in finding relevant features and annotating all tips using these features, the content-based predictor turned out to be not very accurate. It could be that this predictor faces the problem of *limited content analysis* (Adomavicius & Tuzhilin, 2005). This would imply that that the selected features do not well capture the user's preferences and, hence, that the predictor frequently was unable to distinguish differently rated tips based on features. Future studies may thus want to use data-driven methods like principal component analysis to find other features that better capture the preferences of knowledge workers.

The three predictors were also compared with the hybrid prediction strategy. By combining the strengths of the three individual predictors, the hybrid prediction strategy aims to guarantee proper accuracies for each type of user. The obtained results are promising, as they show that the hybrid prediction strategy nearly always outperforms the three individual predictors when it comes to predicting whether a user would perform or reject a tip. Although the hybrid prediction strategy thus enhances classifier performance, it also introduces more complexity to the recommendation method. This increased complexity makes it harder to explain which characteristics of the underlying predictors are important.

7.2 Future of e-coaches

A user study was conducted to evaluate the recommendation method in an online setting. Our first hypothesis was that knowledge workers have a positive attitude towards the e-coach. This hypothesis was confirmed in our user study. The number of followed-up tips was high and most participants agreed that it is pleasant to receive automatic notifications for recommendations. The study also showed that three recommendations per day seems a right amount of suggestions. Moreover, indicating whether a tip was followed-up and asking for a short motivation when a tip was rejected turned out to be a well-design method for providing feedback.

Our second hypothesis was that tailored recommendations are followed-up more often compared to randomized suggestions. We did not find strong evidence for this hypothesis.

Results show that our recommendation method, which provides tailored suggestions, did not substantially increase the number of tips that were performed compared to a method that provided randomized suggestions. We described several possible explanations for this unexpected result. For instance, the small effect of the recommendation method could be caused by the substantial variability that was found on an individual-level. The following paragraphs describe three directions for future research. Based on our findings, we think that exploring these directions may result in more effective e-coaches.

The first direction relates to contextualization. In this project we focused on tailoring the content of tips based on the user's preferences. Results show that only a tips were not followed-up, because the tip was disliked. Instead, tips were mostly rejected, because the moment of recommendation was somehow inappropriate. This finding suggests that future e-coaches may increase their effectiveness by recommending tips at appropriate times. Different researchers have emphasized the importance of providing persuasive messages just-in-time (Fogg, 2009a; Intille, 2004). A simple way of defining appropriate times seems to establish a link between the recommender system and the user's agenda. Such a link enables the system to prevent that suggestions are send when a user is, for example, on the go or in a meeting. We also see opportunities for adopting context-aware algorithms (Adomavicius & Tuzhilin, 2011; Hawkins et al., 2008). Such algorithms are designed to automatically infer what a person is doing and can be used to accurately define the appropriate time for a recommendation.

Second, each knowledge worker probably has his own strategy for coping with high demands and periods of stress. Future projects may therefore want to ask knowledge workers themselves about their coping strategies and collect their tips for improving well-being at work. It is expected that the number of tips that can possibly be suggested increases substantially by using such a crowd sourcing approach. Furthermore, the diversity in crowd sourced tips is probably quite large. For example, some tips may be of lower quality or only be preferred by certain knowledge workers. Our recommendation method can also be used to find suitable crowed sourced tips.

Finally, this thesis focused on whether suggestions were followed-up. Whether performing tips actually contributes to well-being at work still remains an open question. Since well-being at work is such a relevant and important problem for both employers and employees, it seems important to validate e-coaches that aim to improve well-being at work (Kool et al., 2013). Furthermore, e-coaches are intended to invoke a behavior change. The trans-theoretical model states that health behavior changes involve progress through six stages of stage (Prochaska & DiClemente, 2005). A change is called stable once an individual has changed to the healthy behavior more than 6 months ago and still maintains this new behavior. It is not yet known whether the implemented e-coach invokes such longstanding behavior changes. Hence, future studies may want to investigate whether the e-coach is also as effective in the long run.

7.3 Conclusion

We see opportunities for an e-coach which suggests personalized tips that promote well-

being at work, since well-being at work is such a relevant issue for both employers and employees. Our results are promising, as they suggest that knowledge workers have a positive attitude towards the implemented e-coach. Although results show that tailoring the content of tips only slightly increased the number of performed tips, we still believe that personalizing tip suggestions is a good approach for supporting well-being at work. This thesis revealed research directions that may result in more effective future e-coaches. Adopting context-aware algorithms, which ensure that tips are suggested just-in-time, seems the most important personalization method that should be explored in future research.

References

Acampora, G., Cook, D. J., Rashidi, P., & Vasilakos, A. V. (2013). A survey on ambient intelligence in healthcare.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, *17*(6), 734–749.

Adomavicius, G., & Tuzhilin, A. (2011). Context-aware recommender systems. In *Recommender systems handbook* (pp. 217–253). Springer.

Ahn, H. J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, *178*(1), 37 - 51.

Bennett, J., & Lanning, S. (2007). The netflix prize. In *Proceedings of kdd cup and workshop* (Vol. 2007, p. 35).

Buckland, M. K., & Gey, F. C. (1994). The relationship between recall and precision. *JASIS*, *45*(1), 12–19.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, *12*(4), 331–370.

Cialdini, R. (2001). Influence: Science and practice. *Boston: Allyn & Bacon*.

Cialdini, R. (2004). The science of persuasion. *Scientific American*, 7684.

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

Fischer, R., & Milfont, T. L. (2010). Standardization in psychological research. *International Journal of Psychological Research*, *3*(1), 88–96.

Fleiss, J. L., Levin, B., & Paik, M. C. (1981). The measurement of interrater agreement. *Statistical methods for rates and proportions*, *2*, 212–236.

Fogg, B. (2002). Persuasive technology: using computers to change what we think and do. *Ubiquity*, *2002*(December), 5.

Fogg, B. (2009a). A behavior model for persuasive design. In *Proceedings of the 4th international conference on persuasive technology* (p. 40).

Fogg, B. (2009b). Creating persuasive technologies: an eight-step design process. In *Persuasive* (p. 44).

French, J., Rogers, W., & Cobb, S. (1981). A model of person-environment fit. *Society, stress and disease*, *4*, 39-44.

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, *14*(771-780), 1612.

Ge, M., Delgado-Battenfeld, C., & Jannach, D. (2010). Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth acm conference on recommender systems* (pp. 257–260).

Gelman, A., & Hill, J. (2006). *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press.

Giga, S. I., Cooper, C. L., & Faragher, B. (2003). The development of a framework for a comprehensive approach to stress management interventions at work. *International Journal of Stress Management*, *10*(4), 280.

Harrell, F. E. (2001). *Regression modeling strategies: with applications to linear models, logistic regression, and survival analysis*. Springer.

Hawkins, R. P., Kreuter, M., Resnicow, K., Fishbein, M., & Dijkstra, A. (2008). Understanding tailoring in communicating about health. *Health education research*, *23*(3), 454–466.

Herlocker, J. L., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information retrieval*, *5*(4), 287–310.

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

IJsselsteijn, W., de Kort, Y., Midden, C., Eggen, B., & van den Hoven, E. (2006). Persuasive technology for human well-being: setting the scene. In *Persuasive technology* (pp. 1–5). Springer.

Intille, S. S. (2004). A new research challenge: persuasive technology to motivate healthy aging. *Information Technology in Biomedicine, IEEE Transactions on*, *8*(3), 235–237.

Ivancevich, J. M., Matteson, M. T., Freedman, S. M., & Phillips, J. S. (1990). Worksite stress management interventions. *American Psychologist*, *45*(2), 252.

Jamieson, S., et al. (2004). Likert scales: how to (ab) use them. *Medical education*, *38*(12), 1217–1218.

Jawaheer, G., Szomszor, M., & Kostkova, P. (2010). Comparison of implicit and explicit feedback from an online music recommendation service. In *proceedings of the 1st international workshop on information heterogeneity and fusion in recommender systems* (pp. 47–51).

Kaptein, M. C. (2012). *Personalized persuasion in ambient intelligence*. Unpublished doctoral dissertation, Eindhoven University of Technology.

Kaptein, M. C., De Ruyter, B., Markopoulos, P., & Aarts, E. (2012). Adaptive persuasive systems: A study of tailored persuasive text messages to reduce snacking. *ACM Trans. Interact. Intell. Syst.*, *2*(2).

Karasek, R., & Theorell, T. (1992). *Healthy work: stress, productivity, and the reconstruction of working life*. Basic books.

Kim, B. M., Li, Q., Park, C. S., Kim, S. G., & Kim, J. Y. (2006). A new approach for combining content-based and collaborative filters. *Journal of Intelligent Information*

Systems, *27*(1), 79–91.

Knijnenburg, B. P., Willemsen, M. C., Gantner, Z., Soncu, H., & Newell, C. (2012). Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, *22*(4-5), 441–504.

Kool, L., Timmer, J., & van Est, Q. C. (2013). *Keuzes voor de e-coach: maatschappelijke vragen bij de automatisering van de coachingspraktijk.* `http://www.rathenau.nl/publicaties/publicatie/keuzes-voor-de-e-coach.html`.

Koppes, L. L. J., De Vroome, E. M. M., Mars, G. M. J., Janssen, B. J. M., van Zwieten, M. H. J., & van den Bossche, S. N. J. (2011). *Nationale enquête arbeidsomstandigheden 2012. methodologie en globale resultaten* (Tech. Rep.). TNO/SZW/CBS.

Kreuter, M. W., & Skinner, C. S. (2000). Tailoring: what's in a name? *Health education research*, *15*(1), 1–4.

Lacroix, J., Saini, P., & Goris, A. (2009). Understanding user cognitions to guide the tailoring of persuasive technology-based physical activity interventions. In *Proceedings of the 4th international conference on persuasive technology* (p. 9).

LaMontagne, A. D., Keegel, T., Louie, A. M., Ostry, A., & Landsbergis, P. A. (2007). A systematic review of the job-stress intervention evaluation literature, 1990–2005. *International Journal of Occupational and Environmental Health*, *13*(3), 268–280.

Le Fevre, M., Kolt, G. S., & Matheny, J. (2006). Eustress, distress and their interpretation in primary and secondary occupational stress management interventions: which way first? *Journal of Managerial Psychology*, *21*(6), 547–565.

li Huang, S. (2011). Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. *Electronic Commerce Research and Applications*, *10*(4), 398 - 407.

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

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp. 73–105). Springer.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1). Cambridge university press Cambridge.

McNee, S. M., Riedl, J., & Konstan, J. A. (2006). Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *Chi'06 extended abstracts on human factors in computing systems* (pp. 1097–1101).

Noar, S. M., Benac, C. N., & Harris, M. S. (2007). Does tailoring matter? meta-analytic review of tailored print health behavior change interventions. *Psychological bulletin*, *133*(4), 673.

Prochaska, J. O., & DiClemente, C. C. (2005). The transtheoretical approach. *Handbook of psychotherapy integration*, *2*, 147–171.

Pu, P., Chen, L., & Hu, R. (2011). A user-centric evaluation framework for recommender systems. In *Proceedings of the fifth acm conference on recommender systems* (pp. 157–164).

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2010). *Recommender systems handbook*. Springer.

Richardson, K. M., & Rothstein, H. R. (2008). Effects of occupational stress management intervention programs: a meta-analysis. *Journal of occupational health psychology*, *13*(1), 69.

Salton, G., & Buckley, C. (1997). Improving retrieval performance by relevance feedback. *Readings in information retrieval*, *24*(5).

Schafer, J. B., Frankowski, D., Herlocker, J. L., & Sen, S. (2007). Collaborative filtering recommender systems. In *The adaptive web* (pp. 291–324). Springer.

Schaufeli, W. B., & Bakker, A. B. (2013). *De psychologie van arbeid en gezondheid*. Bohn Stafleu van Loghum.

Selye, H. (1956). The stress of life.

Siegrist, J. (1996). Adverse health effects of high-effort/low-reward conditions. *Journal of occupational health psychology*, *1*(1), 27.

Smith, M., & Segal, R. (2014). *Stress management: How to reduce, prevent, and cope with stress.* Retrieved from `http://www.helpguide.org/articles/stress/stress-management.htm`

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*, 4.

The American Heart Association. (2014). *Four ways to deal with stress.* Retrieved from `http://www.heart.org/HEARTORG/GettingHealthy/StressManagement/FourWaystoDealWithStress/Four-Ways-to-Deal-with-Stress_UCM_307996_Article.jsp`

van Dam, S., Bakker, C., & Van Hal, J. (2010). Home energy monitors: impact over the medium-term. *Building Research & Information*, *38*(5), 458–469.

Wiezer, N., Schelvis, R., Zwieten, M., Kraan, K., Klauw, M., van der Houtman, I., ... Bakhuys Roozeboom, M. (2012, 12). *Tno rapport werkdruk.* `http://publications.tno.nl/publication/100619/BjfzVu/wiezer-2012-werkdruk.pdf`.

List of Figures

List of Tables

Appendix A

Materials related to tips

The present appendix contains extra materials that are related to the well-being tips that were suggested to knowledge workers. These materials are all introduced in Chapter 3. First, the tips themselves are listed (Section A.1). Next, this appendix contains a table displaying the annotation of tips (Section A.2).

A.1 List of tips

Table A.1: The 54 tips that could be recommended to knowledge workers. Various scientific articles, websites and magazines on well-being at work were reviewed to collect these tips. Tags are often mentioned in this thesis instead of a complete tip for clarity reasons. Each tag refers to a single tip.

Identifying tag	Content of tip
reflect_wellbeing	Bedenk welke dingen belangrijk zijn voor je welzijn op het werk. Als je weet wat belangrijk voor je is, weet je ook wat je kan verbeteren.
last_success	Benoem je laatste succes. Volgens veel arbeidspsychologen is het belangrijk om trotst te zijn op wat je hebt bereikt. Hierdoor krijg je positieve energie en ontwikkel je meer zelfvertrouwen.
set_goals_week	Bepaal drie belangrijke doelen die je deze week wilt halen en schrijf de doelen op.
reflect_positive_tasks	Denk na over de taken die je vandaag hebt uitgevoerd en energie gaven. Het is belangrijk dat je energie krijgt van je werk.
reflect_negative_tasks	Denk na over de taken die je vandaag hebt uitgevoerd en veel energie kostten. Het is belangrijk dat je energie krijgt van je werk.
identify_completed	Werk aan een positieve instelling en benoem drie dingen die je vandaag hebt bereikt. Hans Selye, een van de pioniers op het gebied van stress, zei ooit: “Adopting the right attitude can convert a negative stress into a positive one”.
set_goal_today	Bedenk een doel dat je vandaag wilt bereiken. Meerdere theorien op het gebied van planning wijzen er op dat het belangrijk is om doelen te stellen die niet te ver in de toekomst liggen. Op deze manier voorkom je dat je het doel uitstelt.
not_check_mail	Sluit je mailprogramma voor een uur af. Nieuwe mails leiden je af van de taak waar je mee bezig bent. Voorkom door deze actie dat je wordt afgeleid.
no_screen	Geef je ogen rust door een minuut niet op een scherm te kijken. Onze ogen zijn niet gemaakt om uren in het licht te kijken. Experimenten hebben aangetoond dat je ogen minder snel vermoeid raken door korte tijd niet naar een scherm te kijken.
notifications_off	Schakel de automatische notificaties voor nieuwe mail uit. Eric Schmidt, nota bene de baas van Google, gaf deze tip op een conferentie. Door de vele prikkels van apparaten is het soms lastig je te focussen en te ontsnappen.
phone_off	Gun jezelf wat rust door je telefoon een uur uit te zetten. Steeds meer mensen hebben last van een ‘information overload’. Door het gebruik van een smartphone neemt de hoeveelheid informatie die je moet verwerken sterk toe. Onderzoek toont aan dat dit leidt tot extra spanning en vermoeidheid.
hug_someone	Stress ontstaat door een grote hoeveelheid cortisol. Onderzoek toont aan dat de hoeveelheid cortisol flink daalt door lichamelijk contact. Verminder je stress door iemand te omhelzen.
chat_colleague	Maak een praatje met een collega. De Duitse onderzoeker Uvnas-Moberg heeft veel onderzoek gedaan naar oxytocine. Hij ontdekte dat het hormoon wordt afgegeven tijdens sociale interacties. Het maakt je rustig en draagt zo bij aan je welzijn op het werk.
tell_joke	Bij Cognitieve Therapie wordt vaak gebruik gemaakt van humor omdat het je ontspant en je creativiteit stimuleert. Aaron Beck, de grondlegger van deze therapie, raadt daarom aan om af en toe een mop of grappig verhaal te vertellen.

Continued on next page

Table A.1 – *Continued from previous page*

Identifying tag	Content
drink_water	Blijf alert en drink een glas water. Een tekort aan water zorgt voor vermoeidheid.
drink_thee	Drink een kop thee. Thee werkt ontspannend, maar wist je ook dat het je stress vermindert? University College London toonde aan dat je stressniveau met 20% kan dalen door het drinken van een kop thee.
no_coffee	Drink vandaag geen koffie meer. Koffie geeft een korte energieboost, maar op lange termijn verhoogt cafene de kans op stress.
eat_fruit	Verbeter je welzijn op het werk en eet een stuk fruit. De universiteit van Wageningen heeft onderzoek gedaan naar fruit op het werk. Wat bleek: 80% van de mensen die fruit eet, voelt zich gezonder en hun productiviteit stijgt met 10%.
cleanup_mailbox	Plan een moment om je mailbox op te ruimen. Onderzoek door IBM Research toonde aan dat een opgeruimde mailbox zorgt voor meer structuur en minder stress.
clean_desk	Ruim je bureau op. Een gestructureerde werkomgeving kan je helpen beter te focussen.
adjust_brightness	Stel de helderheid van je scherm opnieuw in. Opticiens behandelen steeds meer mensen met klachten over vermoeide ogen, ontstaan door het werken met heldere schermen.
adjust_posture	Controleer of je goed zit en stel je bureaustoel, indien nodig, opnieuw in. Volgens ergonomen is een goede lichaamshouding van groot belang om gezond te werken.
focus_breath	Let op je ademhaling en zorg er voor dat je langzaam en diep ademt. Veel onderzoek wijst er op dat je welzijn voor een groot deel afhangt van de juiste ademhaling.
deep_breath_10	Sluit je ogen en neem 10 diepe ademhalingen. Bekende yoga trainingen focussen op een rustige ademhaling, omdat het werkt als een goede ontspanner.
deep_breath_minute	Probeer een halve minuut vanuit je onderbuik te ademen. Een juiste ademhaling geeft je meer energie.
reverse_plank_stretch	Ga met je rug naar het bureau staan. Pak met je handen de rand ven het bureau vast. Span vervolgens je rug en leun 10 seconden met je handen op het bureau.
back_chair_stretch	Ga op een stoel zitten en span je rug. Plaats beide armen achter de rugleuning en pak de ene hand vast met de ander. Span je armen nu voor een paar seconden. Door deze simpele rek oefening worden je arm spieren actief.
calf_raises_stretch	Ga op je tenen staan. Na 3 seconden ga je weer normaal staan. Herhaal dit minimaal 10 keer. Deze oefening houdt je benen actief.
side_turn_stretch	Ga staan met je voeten een stukje uit elkaar. Steek je rechter arm de lucht in een leun naar links. Probeer 10 seconden te hangen zodat je je buikspieren voelt. Vervolgens spiegel je de oefening door je linker arm in de lucht te steken.
back_stretch	Span je rug en doe je handen achter het hoofd. Duw nu voor een paar seconden je ellebogen naar achter. Door deze oefening neemt opgebouwde stress in je schouders en armen af.
touch_shoes	Ga staan en probeer met je schoenen aan te raken. Het menselijk lichaam is eigenlijk niet gebouwd om te zitten. Ergonomen raden daarom aan om je lichaam zo af en toe te 'resetten'. Dit is een goede manier om dat doel te bereiken.
shoulder_stretch	Steek je armen voor je uit en maak een kruis met beide onderarmen. Vervolgens probeer je met de ene arm de andere arm weg te duwen. Na tien seconden maak je het kruis andersom en herhaal je de oefening nog eens. Op deze manier voorkom je een muisarm.
five_minute_break	Ga vijf minuten naar buiten voor een korte pauze. Korte pauzes zorgen ervoor dat je aan het einde van de dag meer energie hebt.

Continued on next page

Table A.1 – *Continued from previous page*

Identifying tag	Content
department_walk	Loop een rondje over de afdeling. Op deze manier vang je drie vliegen in klap: je neemt een korte pauze, beweegt en ontspant door sociale interactie.
coffee_break	Neem een drinkpauze. Het regelmatig nemen van korte pauzes houd je fit en geconcentreerd.
short_break	Neem een korte pauze om bij te tanken. Onderzoek door Stanford University toont aan dat je veel productiever werkt door regelmatig een korte pauze te nemen.
restroom_break	Ga een paar minuten naar een plek waar je alleen bent. Een rustmoment zorgt ervoor dat je de dag beter doorkomt. Trek jezelf bijvoorbeeld even terug op de wc.
watch_funny_clip	Bekijk een grappig filmpje. Lachen is gezond. Het is niet alleen een spreekwoord, maar ook wetenschappelijk aangetoond.
watch_ted_talk	Bekijk een TED talk of een andere inspirerende video. De Amerikaanse psycholoog Todd Thrash heeft aangetoond dat inspiratie zorgt voor nieuwe inzichten en een beter welzijn.
listen_music_relax	Luister naar een relax muziek nummer. Door het luisteren naar rustige muziek word je lichaam ook rustig.
listen_music_classical	Luister naar een stuk klassieke muziek. Diverse onderzoeken hebben aangetoond dat klassieke muziek je concentratie verhoogt.
listen_music_pop	Luister naar een vrolijk pop nummer. Onderzoek door de universiteit van Tilburg toonde aan dat vrolijke muziek zorgt voor een positievere stemming.
make_drawing	Maak een tekening van iets dat je mooi vindt. Het creatief bezig zijn werkt ontspannend.
switch_desks	Ruil een uur met het bureau van je collega. Een andere omgeving werkt ontspannend en inspirerend.
sing_song	Zin een stuk van een lied. De British Association of Music Therapy raadt aan regelmatig te zingen, omdat hierdoor de psychologische druk verlaagd.
pushups	Doe een paar push-ups. Het is algemeen bekend dat sporten goed is goed voor je fysische gezondheid en conditie. Wist je dat beweging je mentale welzijn ook verbetert?
take_stairs	Loop met de trap naar de bovenste verdieping en terug. Als je beweegt wordt de neurotransmitter endorfine geproduceerd. Deze neurotransmitter zorgt ervoor dat je je vrolijk voelt.
grateful_aspects	Bedenk drie dingen waar je dankbaar om bent. Het is goed om zo nu en dan stil te staan bij de dingen die goed gaan en je gelukkig maken.
weekend_plans	Bedenk iets leuks dat je dit weekend wilt doen. Doordeweeks hard werken is goed, maar vergeet niet leuke en ontspannende dingen te doen.
focus_environment	Kijk uit het raam en neem de omgeving in je op. Hoe goed ken je je omgeving eigenlijk?
motivational_quote	Schrijf een motiverende uitspraak op. Als de druk je te hoog wordt, kan deze uitspraak je energie geven.
imagine_happiness	Sluit daarom je ogen en denk aan iets prettigs. Door aan leuke dingen te denken verbetert je stemming ook.
give_compliment	Geef een compliment aan een collega. Japans onderzoek toont aan dat mensen een taak beter uitvoeren als ze een compliment krijgen. Daarbij zijn deze mensen ook meer tevreden met resultaat.
provide_feedback	Geef feedback aan een collega. Het is fijn om te weten waar je goed in bent, maar het is ook fijn om te weten wat beter kan.

A.2 Annotation of tips

Table A.2: The annotation of tips. Tips were annotated by conducting an annotation study. Tags are used to identify tips. All tips are listed in Appendix A.1

Identifying tag	Goal	Focus	Type	Peers	Duration	Mental	Physical	Deviance
reflect_wellbeing	Awareness	Mental	Journalig	Maybe	3	70%	0%	10%
last_success	Awareness	Work	Journalig	Maybe	2	60%	0%	0%
set_goals_week	Awareness	Work	Time mngmt	Maybe	5	80%	0%	10%
reflect_positive_tasks	Awareness	Work	Journalig	Maybe	3	80%	0%	0%
reflect_negative_tasks	Awareness	Work	Journalig	Maybe	4	80%	0%	10%
identify_completed	Awareness	Work	Journalig	Maybe	3	60%	0%	10%
set_goal_today	Awareness	Work	Time mngmt	Maybe	2	80%	0%	0%
not_check_mail	Prevent	Social	Time mngmt	Maybe		40%	0%	40%
no_screen	Recover	Physical	Relax	Maybe	1	10%	0%	10%
notifications_off	Prevent	Social	Time mngmt	Maybe	3	50%	0%	20%
phone_off	Prevent	Work	Time mngmt	Maybe	1	20%	10%	60%
hug_someone	Recover	Social	Social	Yes	2	40%	50%	10%
chat_colleague	Recover	Social	Social	Yes	5	50%	10%	0%
tell_joke	Recover	Social	Social	Yes	3	40%	30%	30%
drink_water	Recover	Physical	Food	Maybe	1	0%	20%	0%
drink_thee	Recover	Physical	Food	Maybe	3	10%	10%	0%
no_coffee	Prevent	Physical	Food	Maybe		40%	0%	0%
eat_fruit	Prevent	Physical	Food	Maybe	3	0%	30%	0%
cleanup_mailbox	Prevent	Work	Time mngmt	Maybe	2	50%	0%	0%
clean_desk	Prevent	Work	Work cond	Maybe	5	50%	60%	10%
adjust_brightness	Prevent	Work	Work cond	Maybe	2	10%	10%	0%
adjust_posture	Prevent	Physical	Work cond	Maybe	3	20%	30%	10%
focus_breath	Prevent	Physical	Relax	Maybe	2	20%	50%	10%
deep_breath_10	Prevent	Physical	Relax	Maybe	1	10%	50%	0%
deep_breath_minute	Prevent	Physical	Relax	Maybe	1	10%	30%	0%
reverse_plank_stretch	Recover	Physical	Exercise	Not	2	10%	70%	90%
back_chair_stretch	Recover	Physical	Exercise	Maybe	2	10%	70%	40%
calf_raises_stretch	Recover	Physical	Exercise	Maybe	2	0%	50%	50%
side_turn_stretch	Recover	Physical	Exercise	Maybe	2	10%	80%	70%
back_stretch	Recover	Physical	Exercise	Maybe	2	10%	40%	30%
touch_shoes	Recover	Physical	Exercise	Maybe	2	0%	50%	40%
shoulder_stretch	Recover	Physical	Exercise	Maybe	2	20%	60%	50%
five_minute_break	Recover	Recovery	Relax	Maybe	5	10%	60%	40%
department_walk	Recover	Social	Social	Yes	4	30%	40%	20%
coffee_break	Recover	Recovery	Relax	Maybe	4	10%	30%	0%
short_break	Recover	Recovery	Relax	Maybe	5	30%	10%	30%
restroom_break	Recover	Recovery	Relax	Not	4	10%	20%	20%
watch_funny_clip	Recover	Recovery	Relax	Not	4	10%	0%	40%
watch_ted_talk	Recover	Recovery	Creative	Maybe	6	40%	0%	40%
listen_music_relax	Recover	Recovery	Creative	Maybe	4	20%	0%	20%
listen_music_classical	Recover	Recovery	Creative	Maybe	4	10%	0%	40%
listen_music_pop	Recover	Recovery	Creative	Maybe	4	10%	0%	40%
make_drawing	Recover	Recovery	Creative	Maybe	3	40%	30%	40%
switch_desks	Prevent	Work	Social	Yes	5	10%	40%	70%

Continued on next page

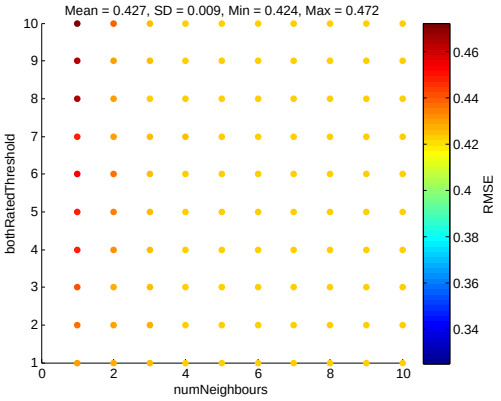
Table A.2 – *Continued from previous page*

Identifying tag	Goal	Focus	Type	Peers	Duration	Mental	Physical	Deviance
sing_song	Recover	Recovery	Creative	Not	2	30%	20%	80%
pushups	Recover	Physical	Exercise	Not	3	20%	100%	70%
take_stairs	Recover	Physical	Exercise	Maybe	5	10%	80%	40%
grateful_aspects	Awareness	Mental	Cognitive	Maybe	3	70%	0%	0%
weekend_plans	Recover	Mental	Time mngmt	Maybe	3	60%	0%	20%
focus_environment	Recover	Recovery	Relax	Maybe	2	20%	30%	20%
motivational_quote	Prevent	Mental	Cognitive	Maybe	3	80%	0%	10%
imagine_happiness	Recover	Mental	Relax	Not	1	50%	0%	40%
give_compliment	Awareness	Social	Social	Yes	1	50%	0%	30%
provide_feedback	Awareness	Social	Social	Yes	3	70%	0%	20%

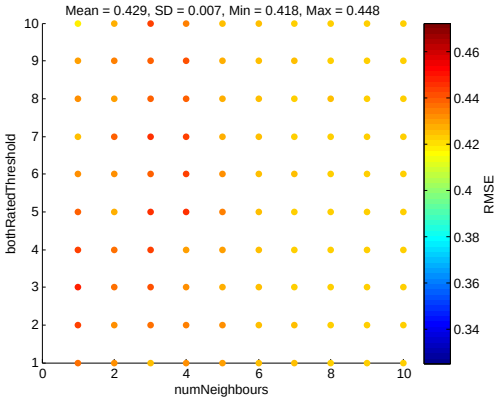
Appendix B

Materials materials related to the offline experiments

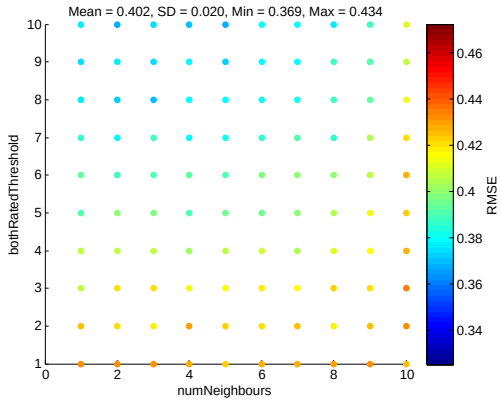
The present appendix contains extra materials that are related to evaluation and optimization of the recommendation method in an offline setting. These materials are all introduced in Chapter 5. Figure B.1 shows the influence of different amounts of training data on the accuracy of the collaborative-based predictor. Furthermore, the influence of different parameter settings on the performance of the collaborative-based predictor is shown in Figure B.2. These results were obtained when the predictor was trained on 40% of the available ratings per user.



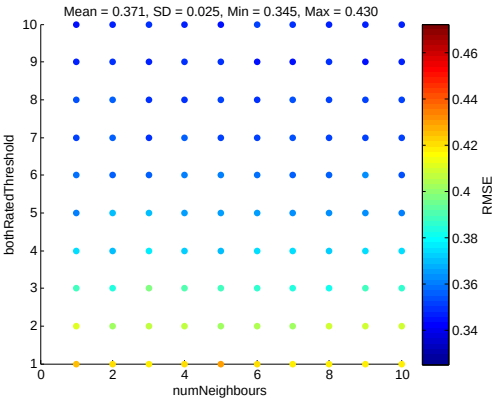
(a) Trained using 10% of the ratings per user.



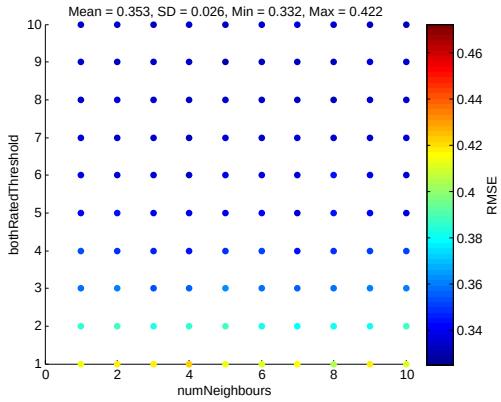
(b) Trained using 20% of the ratings per user.



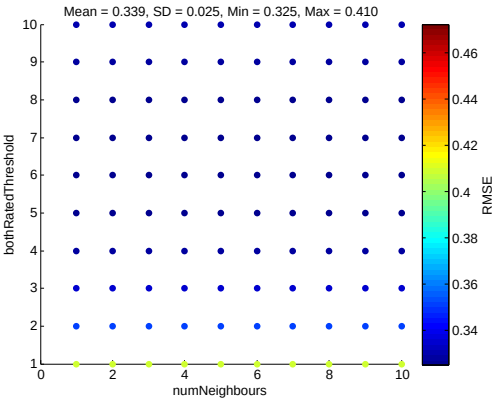
(c) Trained using 40% of the ratings per user.



(d) Trained using 60% of the ratings per user.

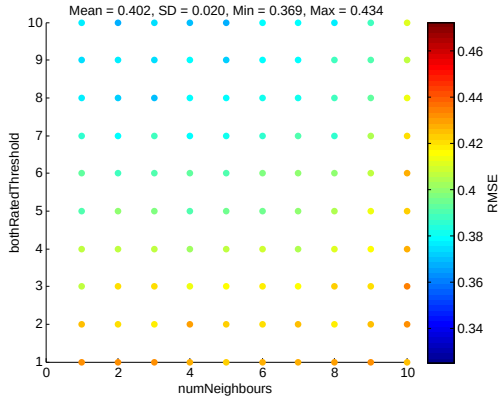


(e) Trained using 80% of the ratings per user.

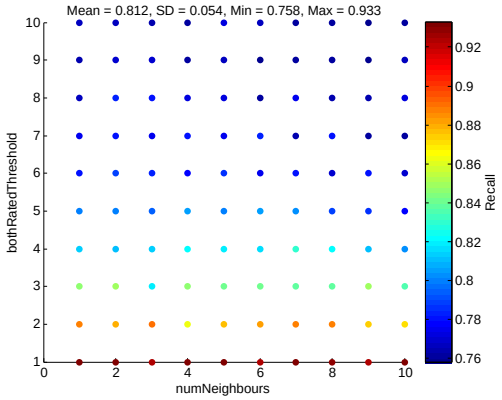


(f) Trained using 100% of the ratings per user.

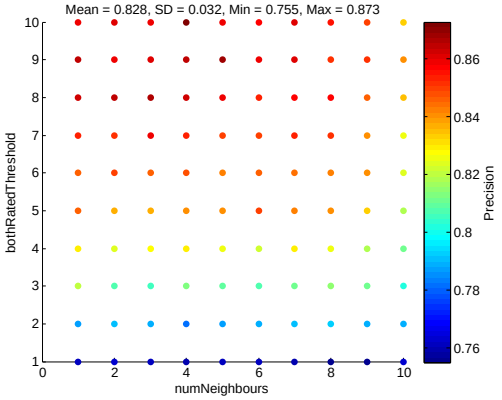
Figure B.1: The accuracy of the collaborative-based predictor for different amounts of training data. Each graph focused on a different amount of available training data. The x-axes show the possible values of parameter *numNeighbours*, whereas the y-axes contain all possible values of parameter *bothRatedThreshold*. For each combination of the two parameters a colored dot is plotted in the graph. The color of the dot reflects the RMSE obtained for that combination. Recall that the baseline RMSE is 0.405 and that lower RMSE's represent lower prediction errors. The results of this investigation are discussed in Section 5.2.1.



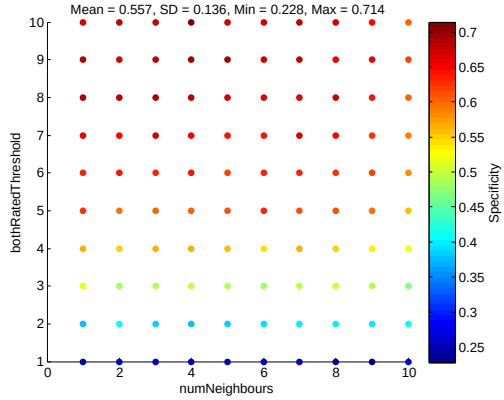
(a) Lower RMSE's are obtained when *bothRatedThreshold* increases. Furthermore, the effect of the neighborhood size is relatively small. It seems that the accuracy is slightly better if the neighborhood is not too large.



(b) The predictor's recall is high if *bothRatedThreshold* is set to 1. When *bothRatedThreshold* increases, lower scores are obtained. There seems to be no influence of the neighborhood size.



(c) The precision gradually increases when higher *bothRatedThreshold* values are used. Furthermore, slightly better scores are obtained if the neighborhood size is set around 4.



(d) The specificity is very high if *bothRatedThreshold* is set to 10. When *bothRatedThreshold* decreases, lower scores are obtained. Different neighborhood sizes do not yield notable changes.

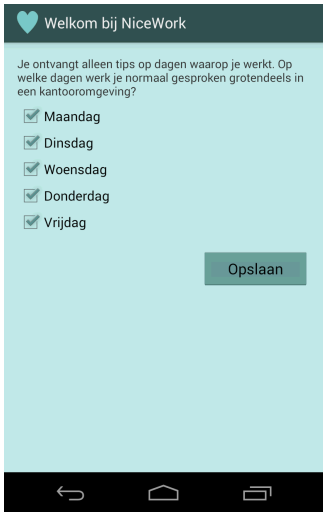
Figure B.2: The performance of the collaborative-based predictor on all evaluation metrics when 40% of the training data was available. Each graph focuses on a single evaluation metric. The x-axes show the possible values of parameter *numNeighbours*, whereas the y-axes contain all possible values of parameter *bothRatedThreshold*. For each combination of the two parameters a colored dot is plotted in the graph. The color of the dot reflects the score that was obtained for that combination. The results of this investigation are discussed in Section 5.2.1.

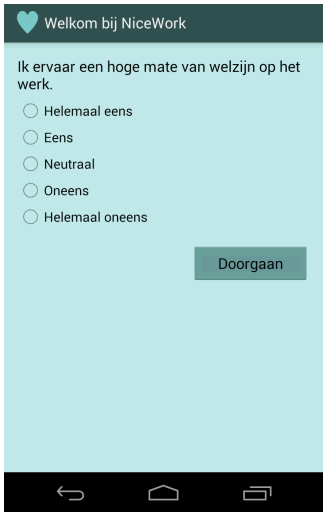
Appendix C

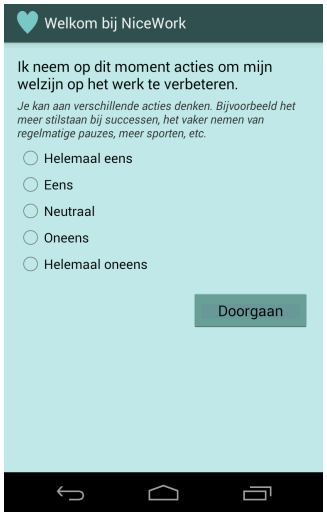
Materials related to the user study

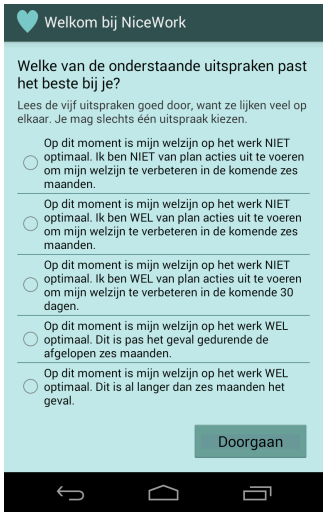
The present appendix contains extra materials that are related to the well-being tips that were suggested to knowledge workers. These materials are all introduced in Chapter 6.

C.1 Pre-questionnaire









C.2 Post-questionnaire

De eerste twee weken van september heb je deelgenomen aan mijn afstudeeronderzoek. Gedurende twee weken heeft de NiceWork app je tips aanbevolen waarmee je je welzijn op het werk kon verbeteren. Hartelijk dank voor het testen van de app!

Het laatste onderdeel van je deelname bestaat uit het beantwoorden van een aantal vragen over je ervaringen met de app. Het beantwoorden van deze vragen neemt ongeveer 10 tot 15 minuten in beslag. Op vragen die beginnen met * is een antwoord vereist.

Om te kunnen beginnen met het beantwoorden van de vragen dien je eerst je proefpersoonnummer en sleutel op te geven. Deze gegevens heb je voor aanvang van het experiment per e-mail ontvangen. Je kan ze ook terug vinden in de NiceWork app onder het 'Instellingen'-scherm.

Alvast bedankt voor het beantwoorden van de vragen! Voor meer informatie kan je contact opnemen met Thymen Wabeke (t.wabeke@student.ru.nl).

***Proefpersoonnummer:**

***Sleutel:**

Algemene gebruiksvriendelijkheid

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
De app is duidelijk.	○	○	○	○	○
De app is mooi.	○	○	○	○	○
De app is intuïtief.	○	○	○	○	○
Na de eerste tip had ik direct door hoe ik de app moest bedienen.	○	○	○	○	○

***Ontving je automatisch drie tips op de door jou opgegeven werkdagen?**

- ○ Ja, ik ontving automatisch drie tips op iedere werkdag.
- ○ Nee, op één werkdag ontving ik geen of minder tips.
- ○ Nee, op meerdere werkdagen ontving ik geen of minder tips.
- ○ Nee, ik ontving helemaal geen automatische tips.

Indien je aangaf dat je een tip niet opvolgde kon je een reden selecteren waarom je de tip niet opvolgde. Onderstaande vragen gaan over die redenen.

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
De redenen om een tip niet op te volgen waren duidelijk omschreven.	○	○	○	○	○
Er stond altijd een geschikte reden in de lijst die ik kon kiezen als ik een tip niet opvolgde.	○	○	○	○	○

Het kan zijn dat je tijdens het experiment het gevoel hebt gehad dat de reden waarom je een tip niet opvolgde niet in de lijst met mogelijke antwoorden stond.

Welke motivaties/redenen om een tip niet op te volgen miste je? Indien je altijd een passend antwoord hebt kunnen geven kun je deze vraag overslaan.

- ☐ Ik ben het niet met de tip eens.
- ☐ Ik kan deze tip niet uitvoeren op mijn werk.
- ☐ Ik mag deze tip niet uitvoeren op mijn werk.
- ☐ De tip kost me te veel moeite.
- ☐ Ik heb al veel vergelijkbare tips ontvangen.

Anders (geef nadere toelichting)

Kwailiteit van de aanbevolen tips

Tijdens het experiment ontving je tips waarmee je je welzijn op het werk kon verbeteren. Onderstaande stellingen gaan over de kwaliteit van de tips.

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
De hoeveelheid automatisch aanbevolen tips was goed.	○	○	○	○	○
De momenten waarop tips werden gegeven waren geschikt.	○	○	○	○	○
Het is prettig om automatisch tips te ontvangen.	○	○	○	○	○
De tips die werden aanbevolen waren interessant.	○	○	○	○	○
De tips die werden aanbevolen waren nuttig.	○	○	○	○	○
De tips die werden aanbevolen waren divers.	○	○	○	○	○
Ik wil graag meer tips ontvangen.	○	○	○	○	○
Ik wil graag minder tips ontvangen.	○	○	○	○	○
Ik vond het storend dat de app mij tips aanbeval.	○	○	○	○	○
Ik ontving vaak tips die veel op elkaar leken.	○	○	○	○	○

***Hoe vaak ontving je in de eerste week van het experiment tips die je waarschijnlijk nooit zou opvolgen?**

- Nooit.
- Enkele keren per week.
- Eén keer per dag.
- Twee keer per dag.
- Drie keer per dag.
- Meer dan drie keer per dag.

***Hoe vaak ontving je in de tweede week van het experiment tips die je waarschijnlijk nooit zou opvolgen?**

- Nooit.
- Enkele keren per week.
- Eén keer per dag.
- Twee keer per dag.
- Drie keer per dag.
- Meer dan drie keer per dag.

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
Naarmate het experiment vorderde ontving ik minder tips die ik niet leuk vond.	○	○	○	○	○
De tips die ik ontving sloten steeds beter aan bij mijn voorkeuren naarmate het experiment vorderde.	○	○	○	○	○
Ik begrijp waarom de tips mij werden aanbevolen.	○	○	○	○	○
De app weet wat voor een tips ik wil ontvangen.	○	○	○	○	○

Eindoordeel

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
Over het algemeen ben ik tevreden over de app.	○	○	○	○	○
De app is een nuttig hulpmiddel om mijn welzijn op het werk te verbeteren.	○	○	○	○	○
Ik zou de app nog eens willen gebruiken.	○	○	○	○	○

Welke nieuwe functies zou je graag terug zien in een toekomstige versie van de app? Wat zou je verbeteren?

Werksituatie

Op deze laatste pagina staat een aantal vragen over je werksituatie.

***Geef voor onderstaande stellingen aan in hoeverre je het met de stelling eens bent.**

	Helemaal oneens		Neutraal		Helemaal eens
Ik ervaar een hoge mate van welzijn op het werk.	○	○	○	○	○
Ik neem op dit moment acties om mijn welzijn op het werk te verbeteren.	○	○	○	○	○

***Hoeveel minuten ben je bereid om per dag te besteden aan het verbeteren van je welzijn op het werk?**

***Hoeveel uur werk je gemiddeld per week?**

***Hoeveel uur werk je gemiddeld per week achter de computer?**

***Hoeveel minuten beweeg je gemiddeld per dag? (zowel intensief als niet intensief)**

***Wat is je geslacht?**

○ Man

○ Vrouw

***Wat is je leeftijd?**

Bedankt voor het invullen van de vragenlijst. Heb je opmerkingen/suggesties?

Appendix D

System guide

The present appendix provides an overview of the technical and practical aspects of the recommender system and the NiceWork app for Android. All the necessary files in order to run the system can be found at: `https://github.com/thymen/NiceWork`[1]

The system, as shown in Figure D.1 consists of two core components, namely: the recommender system and an Android app. The recommender system generates tailored suggestions for tips by implementing the recommendation method described in Chapter 4. These suggestions are provided to users (i.e. knowledge workers in our setting) using the Android app. Both components are connected to the internet and can communicate with each other via a RESTful endpoint that is part of the recommender system. In the following sections each individual component is discussed in more detail.

D.1 Recommender system

The recommender system was implemented in Java and builds upon the PREF framework that was developed by TNO. The project of the recommmender system is called *nicework-pref*. This project runs on a Tomcat webserver and requires a PostgreSQL database to store all relevant information (see Table D.1 for detailed requirements).

D.1.1 Relevant layers

The *nicework-pref* project has several Maven dependencies (all listed in *pom.xml*). The following paragraphs describe the most important dependencies. Each mentioned dependency represents an individual layer of PREF framework and, hence, a layer of the developed recommeder system.

1. The ***pref-common*** dependency contains materials that are shared by different layers. For example, it defines different models (e.g., a user model) and project related exceptions.

[1] This is a private repository. One can gain request by sending an email to `t.wabeke@student.ru.nl`.

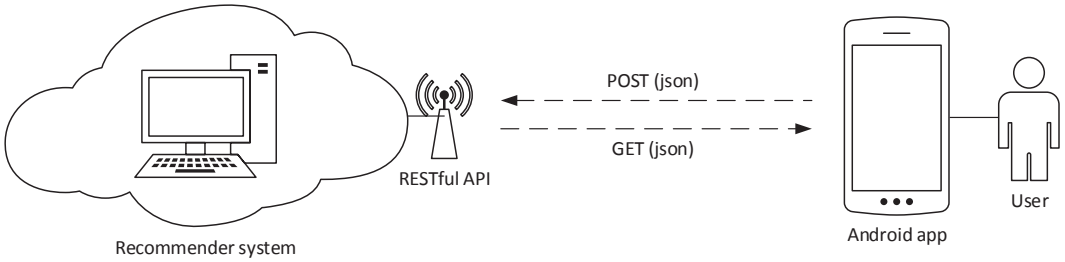


Figure D.1: System overview.

Product	Version	Description
PostgreSQL	9.1.13	Information about items, users and ratings are stored in a PostgreSQL database.
Java Runtime Environment (JRE)	JRE 6 update 45	The recommender system was implemented in Java and thus needs a Java runtime to compile.
Tomcat 7	7.0.30	Tomcat is an open source web server and servlet container that is needed to communicate with the recommender system.

Table D.1: Middleware requirements of the recommender system.

2. The ***pref-dao*** dependency defines the data access object (DAO) layer which is responsible for accessing and storing information in the database. Classes within the project need to instantiate a *DAOFactory* in order to communicate with the database. By using this factory classes can easily access and modify items, users, ratings and recommendations.

3. The ***pref-recommender*** layer is responsible for generating recommendation (lists). *RecommenderManager* and *RecommendationStrategy* are two important interfaces in this layer. *RecommenderManager*'s typically handle recommendation specific calls (generation, getting and rating). The purpose of *RecommendationStrategy*'s is to create a tree of *PredictionStrategy* nodes. A *PredictionStrategy* node can implement an individual prediction algorithm (e.g., the collaborative-based predictor) or a composite that can hold an unlimited number of child prediction algorithm's (e.g., the hybrid prediction strategy). In addition to *PredictionStrategy* nodes, *RecommendationStrategy* also supports clipping and filtering functionality.

4. The ***pref-service*** layer defines the functionally of the API.

5. The ***pref-rest*** layer implements the API that allows communication with the recommender system. A RESTful endpoint was implemented for this project.

URI	Functionality
/user/{user-id}	Add or remove a user.
/user/{user-id}/characteristic/{tag}/{value}	Add or remove the value of a feature/tag for a user.
/user/{user-id}/recommendations	Retrieve the recommendations for a user.
/item/{item-id}	Add or remove an item.
/item/{item-id}/characteristic/{tag}/{value}	Add and remove the value of a feature/tag for an item.
/item/{item-id}/rating/{user-id}/{utility}	Add or remove a rating by a user for an item.
/tag/{name}/{description}	Add or remove a feature/tag.
/perform-batch	Perform batch generating of recommendations.

Table D.2: Important calls supported by the RESTful API of the recommender system. The result of each call is either a HTTP response or JSON object.

D.1.2 Main contribution to the layers

The PREF framework allowed us to relatively easy develop a recommender system. The following paragraphs describe three main contributions that were made to the framework during this project. First, we implemented our own recommendation method. This means that various algorithms were added that extended the third layer which is responsible for generating recommendations.

Second, the PREF framework did not support evaluation of *PredictionStrategy*'s (i.e. prediction algorithms). Therefore, the *Evaluator* class was implemented to run the offline experiments described in Chapter 5. The *FilterRatings* wrapper that temporary filters certain ratings in the database was also implemented for evaluation purposes. For example, this wrapper adds support for leave-on-out cross validation. Furthermore, all *PredictionStrategy*'s implemented the *NiceWorkPredictor* interface which allowed prediction algorithm to run in a evaluation mode.

Finally, we implemented a RESTfull endpoint. This endpoint allows communication with the recommender system using an web-based API. The *NiceWorkEndpoint* class defines all API calls that can be requested. Each call is associated with a resource class. A resource class processes the request and ascertains that the proper service(s) is/are started. The result of a request is either a HTTP response code or a JSON object. The calls that are supported by the RESTful API are shown in Table D.2.

D.2 Android app

The NiceWork app was implemented to provide users with recommendations for tips. The app also enables users to rate recommendations and to look-up previous suggestions.

The app was implemented using *Eclipse ADT* and utilizes the *Android SDK* by Google.[2] This SDK provided most of the API libraries and developer tools that were necessary to build the app. Moreover, the app requires the *appcompat v7* support library by Google and the *PreferenceFragment compatibility layer* by J. Harrison to function properly on older devices.[3] The NiceWork app was targeted for Android SDK version 19 (devices running Android 4.4), but requires at least version 9 (devices running at least Android 2.3).

D.2.1 Relevant packages

As the code of the app often comes with Javadoc descriptions and Android support is widely available on the internet, we slightly limit our explanation of the app in this appendix. Nevertheless, the remainder of the appendix introduces the packages that are relevant when one wants to extend or reproduce parts of the app.

- The ***model*** package defines the models that are used throughout the app. For example, a user model is defined in this package.

- The ***database*** package defines the database that is used to store tips, recommendations and ratings for recommendations. Several data access objects (DAOs) were implemented to easily access and modify information in the database. These DAOs can be accessed by instantiating a singleton in *DatabaseHelper*.

- The ***android*** package contains classes that define the activities/views of the app. The classes in this package are also responsible for processing events that are instantiated by the user (e.g., button clicks). Note that consuming events like transferring data are not directly executed in these view related classes. Instead, such events often trigger an asynchronous task or background task which do not block the user interface. The following activities were implemented:

  - The *MainActivity* is started when the user starts the app. If a user has not yet logged in, this activity immediately stops and *LoginActivity* is started. If the experiment has not started yet, *BeforeExperimentFragment* is shown within the main activity. If the experiment has finished, *AfterExperimentFragment* is shown. Finally, *HomeFragment* is shown when the experiment is running.
  - The *LoginAcitivity* activity is started when the user needs to login. The activity also makes sure that some resources are downloaded (e.g., the start and end date of the experiment and the list of tips).
  - The *AllRecommendationsActivity* shows the list of previously suggested tips. *AllRecommendationsFragment* and *AllRecommendationsAdapter* are associated with this activity.

[2] Both available at `https://developer.android.com/sdk`.

[3] Available at respectively `http://developer.android.com/tools/support-library` and `https://github.com/justinharrison/android-support-v4-preferencefragment`.

  - The *SingleRecommendationActivity* activity shows a single suggested tip and allows the user to rate the tip. *SingleRecommendationFragment* is associated with this activity.
  - The *SettingsActivity* allows a user to change settings. *SettingsFragment* is associated with this activity.

- The ***receivers*** package defines several broadcast receivers. A receiver is an Android component which allows one to register for system or application events. For instance, two receivers were implemented to respectively schedule and process recurring events like retrieving new recommendations. Another receiver was implemented to start the service that creates notifications for tips.

- The ***services*** package defines (background) services. A service is a component which runs in the background without direct interaction with the user. For instance, two services were implemented to respectively schedule and fire the notification for a tip. A service that executes automatic synchronization was also implemented.

- The ***tasks*** package contains several asynchronous tasks. Most of these tasks are relatively simple but time-consuming operations like transferring data that cannot be executed in the thread of the user interface.

In addition to these packages, the Android app also utilizes various resources. Examples of such resources are string resources that provide text strings for the app and layout resources that define the architecture for the user interface in an activity. All resources can be found in the */res* folder.