

# Context relevance assessment and exploitation in mobile recommender systems

Linas Baltrunas · Bernd Ludwig · Stefan Peer ·  
Francesco Ricci

Received: 17 November 2010 / Accepted: 20 April 2011 / Published online: 21 June 2011  
© Springer-Verlag London Limited 2011

**Abstract** In order to generate relevant recommendations, a context-aware recommender system (CARS) not only makes use of user preferences, but also exploits information about the specific contextual situation in which the recommended item will be consumed. For instance, when recommending a holiday destination, a CARS could take into account whether the trip will happen in summer or winter. It is unclear, however, which contextual factors are important and to which degree they influence user ratings. A large amount of data and complex context-aware predictive models must be exploited to understand these relationships. In this paper, we take a new approach for assessing and modeling the relationship between contextual factors and item ratings. Rather than using the traditional approach to data collection, where recommendations are rated with respect to real situations as participants go about their lives as normal, we simulate contextual situations to more easily capture data regarding how the context influences user ratings. To this end, we have designed a methodology whereby users are asked to judge whether a contextual factor (e.g., season) influences the rating given a certain contextual condition (e.g., season is summer). Based on the analyses of these data, we built a context-aware mobile recommender system that utilizes the

contextual factors shown to be important. In a subsequent user evaluation, this system was preferred to a similar variant that did not exploit contextual information.

**keywords** Context · Collaborative filtering · Recommender system · Mobile applications · User study

## 1 Introduction

Recommender systems (RSs) are software tools and techniques providing suggestions for items predicted to be useful for a given user [3, 32]. Often, system-generated recommendations can be more compelling and useful if a description of the contextual situation of the user when is supposed to consume the recommended item is known. For instance, in a travel recommender, the season of the travel or the distance to a place of interest (POI) are important contextual factors to consider before suggesting that a tourist should visit that POI. For this reason, context-aware recommender systems (CARSs) are gaining ever more attention [4] in several application areas including music, places of interest, multimedia, and restaurants (see Sect. 7 for some important examples). Various approaches have been used to incorporate contextual information into recommender systems and have offered improved performance in terms of mean absolute error [10], recall [2], and prediction accuracy [29].

To adapt the recommendations to the user's current contextual situation requires an understanding of the relationship between user preferences and contextual conditions. In many recommender systems, especially those based on collaborative filtering, the user preferences are expressed via item ratings. Therefore, to model the relationship between ratings from context, it has been proposed

---

L. Baltrunas · B. Ludwig · S. Peer · F. Ricci (✉)  
Free University of Bolzano, Piazza Domenicani, 3,  
39100 Bozen-Bolzano, Italy  
e-mail: fricci@unibz.it

L. Baltrunas  
e-mail: lbaltrunas@unibz.it

B. Ludwig  
e-mail: bernd.ludwig@unibz.it

S. Peer  
e-mail: stefan.peer@stud-inf.unibz.it

that explicit user-item ratings should be captured under several different contextual conditions [3, 4]. For instance, the user must rate the suggestion to visit a Museum when she has children with her (contextual factor: *composition of the group of visitors*, contextual condition: *visit with children*), when she does not have children with her, when it is raining/not raining etc.. Such data are expensive to obtain in naturalistic experiments (where the user is given suggestions as they go about their lives as normal) because it requires the same recommendations to be rated in multiple contextual situations. The user must provide item ratings after those items have been *experienced* in several different contextual conditions. Therefore, it is important to minimize the risk to set up an expensive process for acquiring such ratings and then discover a posteriori that the selected contextual conditions were in actual fact irrelevant or not important to investigate.

### 1.1 Context-aware recommendations are difficult to compute

Hence, a major issue for the successful design of CARSs is the discovery of the contextual factors that are worth considering when generating recommendations. This is not an easy problem to solve. It requires formulating informed conjectures about the influence of certain factors before collecting data in naturalistic environments. It is a kind of active learning problem, where the relevance of the data to acquire must be estimated to minimize the cost of the real data acquisition phase [33]. This estimation of the relevance of contextual factors is the first problem for building CARSs.

After a meaningful set of contextual conditions is identified, a predictive model must be built that can predict how the evaluation of an item changes as a function of the contextual factors [6]. This model is then used to select items given a target context. This step again requires the collection and utilization of explicit ratings for items under several distinct contextual conditions. In collaborative filtering (CF), this means, as we mentioned above, the collection of ratings in context. The acquisition of a representative sample of in-context item ratings is the second problem for CARSs.

Finally, after the model is built, a complete context-aware recommender system can be generated. By this, we mean a complete human–computer interaction layer can be designed and implemented on top of the core predictive model. The user should be able to query the system for recommendations, specifying preferences and contextual conditions, and should receive useful suggestions that will be actually acted on [26]. In a travel guide application, for instance, the user may request recommendations adapted to the precise day of the trip, a specific starting location, and

the fact that his family want to travel by bike. Computing good recommendations efficiently on the basis of a given predictive model is the third problem for CARSs.

Moreover, the system must adapt the recommendations to this request and provide an effective visualization of the recommendations, including useful item descriptions and explanations for the recommendations [14, 27, 36]. Visualization and related issues for the user interface therefore are the fourth problem for CARSs.

### 1.2 Toward a methodology for developing CARS

In this paper, we propose systematic solutions for the four problems introduced above. The main contribution therefore is a methodology for supporting the development cycle for CARS. This methodology comprises four steps: determining which contexts are interesting to study; acquiring user ratings in specific contexts of interest; predicting ratings given a specific context; context-aware recommendation visualization and updating. Each of these steps is supported by a specific system or algorithm, which we briefly illustrate in this paper.

First, in order to quantitatively estimate the dependency of the user preferences from an initial candidate set of contextual factors, we developed a web tool for acquiring context relevance subjective judgements. To achieve this, the user is requested to evaluate if a particular contextual condition, e.g., “it is raining today”, may have a positive or negative influence on the user’s rating of a particular item. We analyzed the resulting data statistically to establish which contextual factors are most likely to influence the user decisions for different types of item.

Second, we developed a further web application that generates example contexts and recommendations for these contexts, which the participants were asked to rate. The generation process was informed by the results of the previous step. The more relevant a contextual factor is according to the results of the first data collection, the more often contextual conditions specifying this factor are generated for rating requests in the second system. For instance, because the first results show that the distance to a castle is a very relevant contextual factor for that POI type, in the second application the users are very likely to be requested to rate a recommendation to visit *Mareccio Castle* assuming that they are either close to or far from it. The ratings collected in this evaluation, as with the previous step, were provided by the users not while experiencing the items in these contextual conditions but by imagining the situation and providing a judgement. The advantage of this approach is that ratings can be recorded for multiple relevant contextual conditions, without any constraint related to what the user has actually experienced in the past. This is not possible using the approach typically used

to evaluate context-aware recommender systems based on collaborative filtering.

Third, a predictive model is built, the goal of which is to predict the user's ratings for items under other contextual situations where these ratings are not known. Recommendations, as usual, are then selected among the items with the largest predicted rating for the context situation of the target user. Here, a contextual situation is a combination of several individual contextual conditions, e.g., the user is asking a recommendation in a rainy day (condition one) when he is traveling with children (condition two). This model computes the influence of each contextual condition on the prediction and therefore can also be used to isolate the contextual conditions that have a larger effect either in increasing or decreasing the predicted ratings for items.

Finally, the predictive model is exploited in a mobile application, called ReRex—a travel planning application aimed at recommending points of interests (POIs) to mobile users. It offers two functionalities. Firstly, context-dependent and personalized recommendations of touristic POI. Secondly, assistance in the preparation of a complete itinerary and the modification of the itinerary according to circumstances and eventualities that occur during the itinerary. In particular, the user can request recommendations for a particular context. The application presents the recommendations generated by the predictive model and justifies the recommendations with a direct and simple explanation of the main reason why an item is recommended for that particular contextual situation. So, referring to the example mentioned previously, the system may justify the recommendation to a particular POI (e.g., Oetzi Museum) because the user is traveling with children. Furthermore, the mobile application can asynchronously notify the user that a contextual condition (e.g., weather) is likely to change and therefore revises the recommendations for the user.

ReRex mobile application was tested in a live user experiment to validate the full process defined by the methodology proposed above. The methodology described in this paper is pretty general and can be applied to recommender systems in various domains. A more detailed description of the web interface and the methods to acquire the relevance of context can be found in Baltrunas [11]. In fact, we are currently also testing the approach on an in-car context-aware music recommendation scenario.

### 1.3 ReRex: the new methodology in practice

In conclusion, in this paper, we focus on the discussion of our methodology for building context-aware recommender systems (CARS) and the presentation of ReRex, our prototype for putting the methodology into practice.

In Sect. 2, we explain our approach for acquiring user data needed for estimating relevant contextual factors. We claim that the contextual factors that are worth considering in a CARS must be assessed before the in-context evaluations are collected. In fact, we discover that the relevance of a contextual factor can be user and item specific. We show how the relevance can be acquired by asking users to evaluate if a contextual condition for that factor has a positive or negative impact on his decision to select the item.

In Sect. 3, we present the methodology that we have applied to acquire in-context item ratings, i.e., assuming that a certain relevant contextual condition holds. We illustrate how useful in-context evaluations can be acquired by asking to users to imagine that a contextual condition holds and then to rate an item. Even if these ratings may differ from those that would be collected after the user has really experienced the item in the context [9, 28], they nevertheless help to model the expected appropriateness of an item [32].

In Sect. 4, our predictive model for context-aware recommendations, which extends matrix factorization, is presented and its accuracy is evaluated off-line with the acquired rating data. We show that using the proposed model both personalization and context awareness can substantially improve the model accuracy.

In Sect. 5, we introduce our context-aware mobile recommender system prototype ReRex. ReRex offers places of interest recommendations, but also identifies the contextual condition with the greatest (positive) impact on the predicted rating of a recommended item. This condition is used as an argument to explain what makes the item better suited in the current user situation.

ReRex is evaluated in Sect. 6. It is shown that ReRex received a better evaluation compared with a similar, but not context-aware system. We observe that in this evaluation recommendations were generated for the target users without knowing any of their ratings for items. In fact, ReRex used a context aware but not personalized recommendation model (a variant that is supported by our rating prediction technology). Hence, we show that effective recommendations can be computed knowing only the user's current contextual situation. The paper ends with a discussion of the related work, the conclusions we have drawn from this study, and lists some open issues that call for some future work.

## 2 Acquiring context relevance

The first step of the methodology described in the Introduction pertains to the computation of a quantitative measure of the influence of some potentially relevant

contextual factors on the user-item selection decisions. To attain this objective, we collected data how users *change* their inclination to visit a POI while they imagine that a certain contextual condition, i.e., a specific value for a contextual factor, holds. For that purpose, we designed an online web application.

First, a large set of contextual factors and conditions (values for the factors), as found in the literature on consumer behavior in tourism was selected [35]. The selected contextual factors and conditions are listed in Table 1. We observe that an alternative approach for identifying potentially relevant contextual conditions could be a qualitative study, such as a diary study. Then, a relatively small list of types of POIs in Bolzano (Italy) and other nearby towns were identified. POIs were aggregated into categories in order to avoid sparseness of the collected data; that is, we assume that the influence of a contextual factor is uniform for all the POIs in a category. We defined ten categories: castle; nature wonder; cycling and mountain biking; theater event; folk festival, arts and crafts event; church or monastery; museum; spa and pampering; music event; walking path.

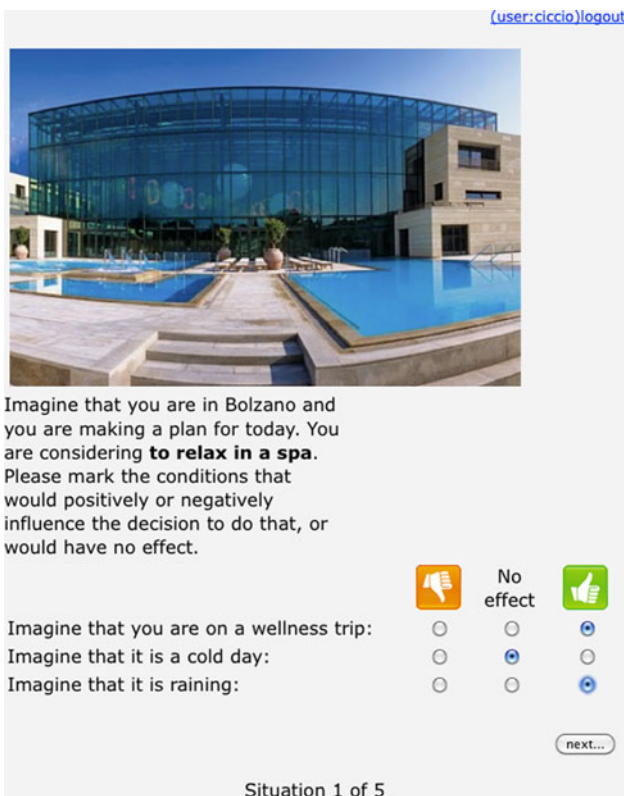
Then, we have developed a simple web application (see Fig. 1) for acquiring the relevance of the selected contextual factors for the POIs categories. In this web application, the users are requested to evaluate the influence of the selected contextual conditions on their decisions to visit the POIs belonging to a randomly selected item category. As

an example of the questions posed to the user consider the situation depicted in Fig. 1. Here, we first request to the user to imagine a typical visit situation: “Imagine that you are in Bolzano making a plan for today. You are considering to relax in a spa”. Then, we asked to select the influence of three randomly selected contextual conditions on that visit. The influence can take one among three possible values: positive, negative, or neutral. As an example of a contextual condition: “Imagine that it is a cold day”. So the user was requested to evaluate if this condition increases, decreases, or has no influence on his decision. Every user was requested to interact with at least five of these pages (as in Fig. 1).

Thirty-three participants (mostly from our computer science faculty) took part in this web survey. Overall, they gave 1,524 responses to one of the questions shown in Fig. 1. For the specification of the context, the factors presented in Table 1 were applied in a randomized way; for each question, a POI category was drawn at random along with a value for a context factor. This sampling has been implemented such that a uniform distribution over the possible categories and the possible contextual conditions is achieved. A different sampling could also be applied if a prior distribution is known. In this way, one can try to obtain more information on the conditions that are more likely to occur or that are more likely to influence the user’s decision.

**Table 1** Context factors used in the web survey

Context factor	Conditions	Context factor	Conditions	Context factor	Conditions	Context factor	Conditions
Budget	Budget traveler	Crowdedness	Not crowded	Companion	With girl/boyfriend	Season	Spring
	High spender		Crowded		With family		Summer
	Price for quality	Empty	With children		Autumn		
Time of the day	Morning time	Travel goal	Health care	Weather	Alone	Transport	Winter
	Afternoon		Cultural experience		With friends		Public transport
	Night time		Scenic/landscape		Snowing		No means of transp.
Day of the week	Weekend	Education		Clear sky		Bicycle	
	Working day		Hedonistic/fun		Sunny		Car
Distance to POI	Near by	Social event		Rainy	Temperature	Warm	
	Far away		Religion				Cloudy
Knowledge About area	New to city	Activity/sport		Mood	Happy	Time available	Hot
	Citizen of the city		Visiting friends				Active
	Returning visitor	Business		Sad			More than a day
			One day				



**Fig. 1** Web-based survey tool to acquire context relevance

### 2.1 Analysis of context relevance

The web survey delivered samples for the probability distributions  $P(I, C_i, T)$ ; where  $C_1, \dots, C_N$  are the context factors that (may or may not) influence the user decisions,  $T$  is a POI category and  $I$  (Influence) is the response variable, taking one of the three values: positive, negative, or neutral. The probabilities  $P(I|C_i, T)$  model the influence of the context factors on the user’s decision. The knowledge of  $P(I|C_i, T)$ , as we will discuss in the next section, can drive the acquisition of context-dependent ratings for the context factors that have a large influence on the user evaluation for the items in a given category  $T$ . Hence, it is interesting to measure, given a category  $T$ , the impact of context  $C_i$  on  $I$ , and then which  $C_i$  best explains  $I$ .

The spread of a categorical variable  $X = \{x_1, \dots, x_n\}$  can be measured with its entropy [23]. If  $P(X = x_i) = \pi_i$ , the entropy of  $X$  is defined as follows:

$$H(X) = - \sum_{1 \leq i \leq n} \pi_i \cdot \log \pi_i$$

This measure of the spread can be used to estimate the association of two variables, e.g.,  $I$  and  $C_i$ , i.e., how much the *Inclination of the user to visit an item*, is influenced by the context of the current situation, e.g., the *Current weather condition*. Informally, this influence is strong if the

knowledge about the weather reduces the spread of  $I$ , and it is weak if the spread of  $I$  remains unchanged even if one knows the weather. Therefore, the difference between the spread of  $I$  and the expected spread of  $(I|C_i)$  is a measure for the association of the two variables. As the spread of  $(I|C_i)$  should not be larger than that of  $I$  alone we can normalize the difference to the interval  $[0,1]$  by:

$$U = \frac{H(I|T) - H(I|C_i, T)}{H(I|T)}$$

$U$  is 1 if the spread of  $(I|C_i, T)$  is zero. This occurs if for each value of  $C_i$  the value  $I$  is certain. Conversely,  $U$  is zero, if  $C_i$  does not have any influence of  $I$ , in which case the spread of  $(I|C_i, T)$  is not different from that of  $(I|T)$ .

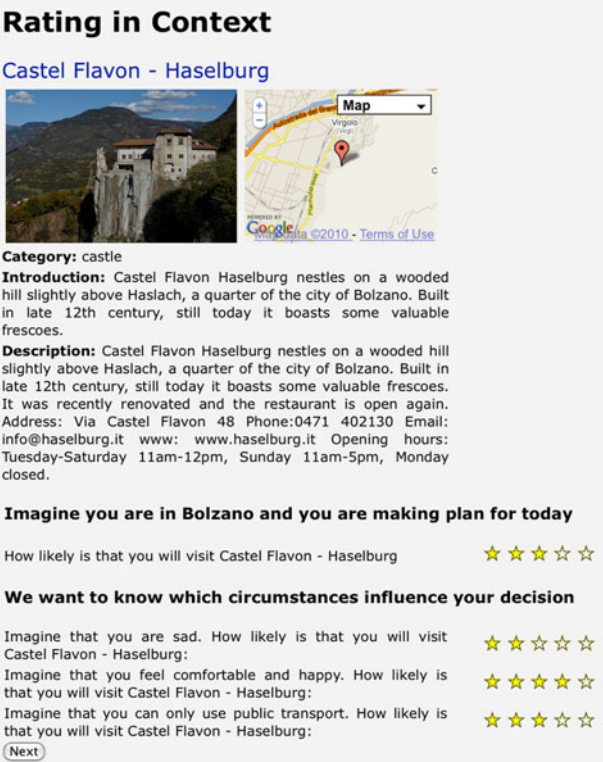
Hence, in order to understand which context factors help most to decrease the uncertainty about  $I$ , we have computed  $U$  for all factors and POI categories. Ordering the factors in descending value of  $U$ , one gets the results reported in the Appendix 1 of this paper. That table indicates that there are some factors that indeed are relevant for all the categories, among them: *distance to POI*, *time available*, *crowdedness*, and *knowledge of the surroundings*. Others often appear to be less relevant: *transport*, *travel goal*, *day of the week*. Finally, some factors appear to have a different relevance depending on the category.

### 3 Acquisition of ratings in context

In the second phase of our study, as we wanted to measure how a POI’s rating is actually modified by the contextual conditions. For that purpose, we considered a set of POI in Bolzano, namely 20 POIs, and asked the users to rate them under different contextual conditions. The same POIs have been also used in the mobile recommender system, ReRex, that is described in Sect. 5. In fact, in order to be able to measure the *variation* of the rating for a POI when a contextual condition holds or not, in each interview, we asked the user to rate a POI in two situations: without considering any contextual condition and assuming a particular contextual condition to be true. Figure 2 shows a screenshot of the web tool that we implemented for this task. In an interview, a single POI is considered and four ratings are requested: first, in general, how likely is the user to visit the POI and then the same evaluation but assuming that three alternative contextual conditions are given. In these interviews, each contextual factor is drawn from the full set of factors proportionally to its relevance  $U$ , as computed in the previous phase. We collected 322 of such interviews from 20 different users, that live in Bolzano city; hence, a total of 1,272 ratings were acquired for 20 POIs (one-fourth of them without a context specification

### Rating in Context

#### Castel Flavon - Haselburg



**Category:** castle

**Introduction:** Castel Flavon Haselburg nestles on a wooded hill slightly above Haslach, a quarter of the city of Bolzano. Built in late 12th century, still today it boasts some valuable frescoes.

**Description:** Castel Flavon Haselburg nestles on a wooded hill slightly above Haslach, a quarter of the city of Bolzano. Built in late 12th century, still today it boasts some valuable frescoes. It was recently renovated and the restaurant is open again. Address: Via Castel Flavon 48 Phone:0471 402130 Email: info@haselburg.it www: www.haselburg.it Opening hours: Tuesday-Saturday 11am-12pm, Sunday 11am-5pm, Monday closed.

**Imagine you are in Bolzano and you are making plan for today**

How likely is that you will visit Castel Flavon - Haselburg ★★★★★

**We want to know which circumstances influence your decision**

Imagine that you are sad. How likely is that you will visit Castel Flavon - Haselburg: ★★★★★

Imagine that you feel comfortable and happy. How likely is that you will visit Castel Flavon - Haselburg: ★★★★★

Imagine that you can only use public transport. How likely is that you will visit Castel Flavon - Haselburg: ★★★★★

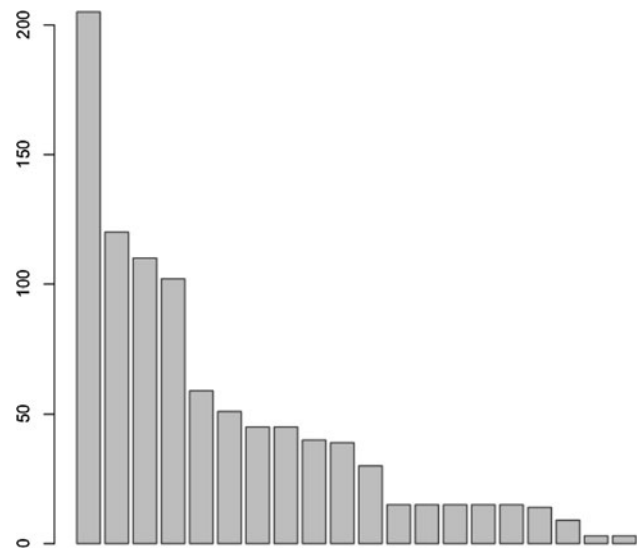
Next

**Fig. 2** Web-based survey tool to acquire ratings in context

and three-fourths with a contextual condition specified). We observe that these ratings have been used to build the rating prediction model (described in Sect. 4) and to generate the recommendations in the ReRex mobile application (described in Sect. 5).

Figure 3 shows the histogram of the ratings entered by the users during the study (each bar corresponds to a user). The number of ratings entered by the users differs significantly from user to user, as it has been often observed in user-generated content (UGC). Moreover, it is worth noting that, by construction, we collected more ratings for in-context evaluations related to the most influential contextual factors. The rationale is that these factors are predicted, based on the previous analysis (Sect. 2), to have a larger impact on the ratings and therefore should bring more useful information. But, this has also another effect on the predictive model—to be described in Sect. 4: the model accuracy will be higher for predictions under these contextual conditions.

In the Table 3 in Appendix 2, we present the analysis of the data collected in this phase for a set of well-known points of interest in Bolzano. In this table, *Mean context yes* (MCY) is the average rating for all items in the specified category assuming that the given contextual condition holds. *Mean context no* (MCN) is the average rating for the same items without any contextual condition to hold. The



**Fig. 3** User statistics for phase two of the knowledge acquisition process. The x-axis enumerates the participants in the study; the y-axis gives the number of ratings by each participant

given  $p$  value is that of a  $t$  test<sup>1</sup> comparing MCN to MCY. Please note that in this table we list only the contextual conditions with a significant difference of the two means MCN and MCY.

It is notable that in the majority of the cases, context has a negative influence on the ratings of the users. In several cases in which the rating in context is higher, the  $p$  values are not significant. To find a plausible interpretation for this observation, we recall from psychology that people are more sensitive to negative than to positive influence when making decision under uncertainty [37]. Hence, the user may remember more vividly the negative impact of context rather than the positive one.

Furthermore, these results may be a consequence of the artificial set up of the experiment: doing the survey in front of a desktop PC may lead to an overvaluation of exceptional situations as they occur in some interviews. Another explanation is that most POI have a high rating without context. As a consequence, in such a situation, for the user, there is no way to rate a POI higher even if a context factor would support the user's positive attitude toward this POI.

A detailed look at the data reveals the fact that significant negative ratings can be found very often if the contextual condition under consideration describes an “extreme” situation for visiting the POI or doing the recommended activity. For example, crowded and empty are both valued negatively while the rating for “not so crowded” is above the rating without context (however without significance).

<sup>1</sup> For the ratings we assume a normal distribution; therefore the  $t$  test is appropriate to detect significant deviations of MCY from MCN.

Furthermore, ratings are significantly negative if the user probably supposed the factor would lower the value of the POI or decrease the user's comfort, e.g., "far away" leads to very low ratings of POI—as the factors "half a day" or "car" (may be users assume a car would not be helpful to visit a POI in the city center).

Context values may also be in conflict with user preferences. For instance, the contextual factor *travel goal* can take the two values *scenic/landscape* and *sports/activity*. In these contextual conditions, many POIs received low ratings because located in the city center and in general they are not suited for people who would prefer to do sport activities like trekking or climbing (very popular among the users that tried the system).

To summarize these observations, the table illustrates that for many contextual conditions statistically significant differences can be detected in the ratings compared to the ratings of the same points of interest without that conditions. We conclude that context dependency must be taken into account and is likely to improve the accuracy of the recommendations compared to a non-context-aware system. The off-line study described in the next section, and the live user evaluation of our mobile application ReRex (see Sect. 5) provides empirical evidence for this claim.

#### 4 Context-aware recommendation algorithm

An important component of the ReRex mobile context-aware recommender system, which will be illustrated in this section, is its rating prediction and recommendation algorithm. The rating prediction component computes a rating prediction for all the items, while assuming that the current user context holds. The current context is partially specified by the user, using the system GUI, and partially acquired automatically by the system (as explained in the next section). Then, the items with the highest predicted ratings are recommended. In this section, we will present an extension of the matrix factorization (MF) rating prediction technique that incorporates contextual information to adapt the recommendation to the user target context. We have designed and tuned our MF extension considering some additional requirements that we deem important for our application. In particular, the main features of our algorithm are as follows:

- Trainable with different types of data.

As we saw before, the output of the data collection procedure is a set of subjective user ratings for items under specific contextual conditions—or without any consideration of the context (default context). Each user rating is actually tagged with one contextual condition only. To give an example, the visit to Castle Mareccio is rated 4 when

this POI is not far away from the current user position. But, when a user is actually using the mobile application (as described in the next section), she may also provide a rating. In this case, the rating is recorded with all the other contextual conditions (e.g., weather) acquired by the system for that situation (and considered as relevant by the user). Therefore, this user rating may be associated with zero, one, or *more* contextual conditions. The rating prediction algorithm that we have designed is able to train the rating prediction model using both kind of data indifferently.

- Update of the predictions when the context changes.

The contextual conditions considered in our application can rapidly change. Moreover, our application allows the user to enable or disable some of the contextual factors (see Sect. 5). This means that it is up to the user to decide if the system must take into account a contextual factor or not. In fact, we believe that this is an important feature of context management, since as we have shown before, it could be rather difficult for a completely autonomous system to decide about that. Consequently, the predicted recommendations should be adapted, to any change of the contextual situation. This requires a method that can compute context-aware recommendation predictions in a short time.

- Explanations for the recommendations.

Explanations can have several positive effects on the system usability—among them enhancing the system transparency, its persuasiveness and the user trust on the system [36]. We believe that referring to the contextual conditions under which the recommendations are generated can have a strong effect on the user acceptance of the recommendation. For example, explaining that a visit to a museum is explicitly recommended because today is a cold day, can add a good and possibly crucial argument for the acceptance of the recommendation. Therefore, we included in our technique a component that isolates a single contextual condition, among those relevant in the current situation, and motivates the recommendation by referring to that contextual condition as a reason for recommending the item.

##### 4.1 The predictive model

The above listed features have shaped our algorithmic solution. As we need to compute recommendations for rapidly changing contextual conditions, we selected a model-based rating prediction approach. The predictive model is therefore trained off-line, once every hour or when a meaningful number of new ratings are collected. The trained model is then used to generate recommendations. The important feature of this approach is that it can

generate rating predictions in a constant time.<sup>2</sup> Before describing how the other system features have been implemented, we must first describe the details of the predictive model.

In [20], the authors present a matrix factorization approach to CF that uses “baseline” parameters for each user and item. Baselines are additional model parameters that are introduced for each user and item. They indicate the general deviation of the rating of a user or an item from the global average. So, for instance, the user baseline, which is learned for a user that tends to rate higher than the average of users’ population will be a positive number. Additional parameters like these slightly increase the model complexity. However, they can also improve its accuracy, as we will show later. The usage of baseline parameters also serves for taking the impact of context into account. This has been already shown by [21], where the authors have introduced several baseline parameters to model the time dependency of the ratings.

We have extended and adapted that approach, and we have incorporated more contextual dimensions into the MF model. We propose to introduce one model parameter for each contextual condition and item pair. This allows to learn how the rating deviates from the classical personalized prediction as effect of the context. This deviation is the *baseline* for that condition and item combination. Broadly speaking, the system computes a rating prediction for a user-item pair and then adapts that to the current context using the learned context-dependent baselines. Suppose that the user  $u$  rates the item  $i$  as  $r_{ui}$ . Now, the user is asked to rate the same item  $i$  in a given contextual condition  $c$ , producing the rating  $r_{uic}$ . The difference of the two ratings,  $r_{ui} - r_{uic}$ , indicates how much  $c$  influences the rating, and this information can be exploited in the model learning stage. Clearly, this is an oversimplified example, as there is typically noise in the data. Moreover, the model must simultaneously learn other parameters for correctly modeling the dependency of the rating on  $u$ ,  $i$  and  $c$ . In our data set of context-aware ratings, a rating  $r_{uic_1, \dots, c_k}$  indicates the evaluation of the user  $u$  of the item  $i$  made in the context  $c_1, \dots, c_k$ , where  $c_j = 0, 1, \dots, z_j$ , and  $c_j = 0$  means that the  $j$ th contextual condition is unknown, while the other index values refer to possible values for the  $j$ th contextual factor. The tuples  $(u, i, c_1, \dots, c_k)$ , for which rating  $r_{uic_1, \dots, c_k}$  is known, are stored in the data set  $R = \{(u, i, c_1, \dots, c_k) \mid r_{uic_1, \dots, c_k} \text{ is known}\}$ . Note, that in our collected data set, only one contextual condition is known and all others are unknown; hence, in  $R$ , there are ratings for which only one among the indices  $c_1, \dots, c_k$  is different from 0.

We recall that MF aims at factorizing the ratings matrix into two  $m \times d$  and  $n \times d$  dimensional matrices  $V$  and  $Q$  respectively. A user is then represented with a column vector  $v_u \in V$  and an item  $i$  with a column vector  $q_i \in Q$ . One can balance the capacity and generalization capability of the predictive model by increasing or decreasing the dimensionality  $d$  of  $V$  and  $Q$ . We propose to compute a personalized context-dependent rating estimation using the following model.

$$\hat{r}_{uic_1, \dots, c_k} = v_u q_i^\top + \bar{i} + b_u + \sum_{j=1}^k b_{ic_j} \quad (1)$$

where  $v_u$  and  $q_i$  are  $d$  dimensional real-valued vectors representing the user  $u$  and the item  $i$ .  $\bar{i}$  is the average of the item  $i$  ratings in the data set  $R$ ,  $b_u$  is the baseline parameter for user  $u$  and  $b_{ic_j}$  is the baseline of the contextual condition  $c_j$  and item  $i$ . If a contextual factor is unknown, i.e.,  $c_j = 0$ , then the corresponding baseline  $b_{ic_j}$  is set to 0. In this way, one can learn the influence only of the known contextual conditions.

This model needs further explanations. Clearly, when designing any model, one must take into account the amount of available training data and guess how complex the relationships in the underlying data are. More complex models could better fit the data; however, if there are not enough training data, this can have a negative effect. Since we do not have a large amount of bootstrapping data, in comparison to other data sets such as Netflix<sup>3</sup> or MovieLens,<sup>4</sup> we opted for this simple linear model that is shown in (1). In that model, the contextual information is exploited by using a set of model parameters,  $b_{ic_j}$ . In the learning phase, which is illustrated later on, the influence of the contextual factor on the ratings is fully described by these parameters.

Note, that this model assumes the independence of the contextual factors. This could appear as a limitation, as one can easily find examples that do not satisfy this assumption. However, this assumption was implicitly made when we collected the training data as we asked the users to provide their ratings assuming a single contextual condition per time. Hence, the model bias fits the data acquisition bias. Nevertheless, the model can capture the influence of multiple contextual factors in the prediction stage but can still be learned with our particular training data. This is done by summing up over all the baselines of the known contextual factors. In addition, this model is simple and introduces less parameters, compared to another where the dependencies among the contextual factors are explicitly modeled. This is beneficial for the small data set that we have used. In the

<sup>2</sup> In fact, as we will see later on, it is constant with respect to the number of input data and linear in the number of contextual conditions.

<sup>3</sup> <http://www.netflixprize.com>.

<sup>4</sup> <http://www.movielens.org>.



future, we plan to test this model also on larger and maybe more complex training data.

The flexibility of this approach is also shown by the observation that the proposed model could be extended to incorporate more information, if available, e.g., adding a parameter that depends only on the context, but not on item, would allow the system to learn more general rules about context, such as, on a sunny day, all the items are better rated. Adding such kind of dependency can make the model even more accurate without increasing its complexity, but will have *no effect* on the items’ ranking, since the ratings of all the rating predictions will be altered by the same quantity.

### 4.2 Model training

In order to generate rating predictions, the model parameters should be learned using the training data. We define the learning procedure as an optimization problem:

$$\min_{v_u, q_i, b_u, b_{ic_j}} \sum_{r \in R} \left[ \left( r_{uic_1, \dots, c_k} - v_u q_i^\top - \bar{r} - b_u - \sum_{j=1}^k b_{ic_j} \right)^2 + \lambda \left( b_u^2 + \|v_u\|^2 + \|q_i\|^2 + \sum_{j=1}^k b_{ic_j}^2 \right) \right]$$

where  $r = (u, i, c_1, \dots, c_k)$ . For better generalization performance, a regularization term is added, as it is usual in this type of models. Regularization is controlled by the  $\lambda$  meta parameter. As  $\lambda$  grows, the model becomes more “rigid” and fits less the variability in the training data. We have used stochastic gradient descent for solving this problem. This has been proved to be an efficient approach [20]. We update all the parameters of the model as follows:

- $b_u \leftarrow b_u + \gamma_{b_u} (err - \lambda b_u)$
- $b_{ic_j} \leftarrow b_{ic_j} + \gamma_{b_{ic_j}} (err - \lambda b_{ic_j}), \quad \forall c_j \neq 0, \quad j = 1, \dots, k$
- $v_u \leftarrow v_u + \gamma_{v_u} (err \cdot q_i - \lambda v_u)$
- $q_i \leftarrow q_i + \gamma_{q_i} (err \cdot v_u - \lambda q_i)$

This procedure updates one after another all parameters that are moving in the opposite direction of the gradient, while all the other variables are kept unchanged. Actually, to be precise, the first entry of the two vectors  $v_u$  and  $q_i$  is learned until these vectors do not further improve the model accuracy. Subsequently, the second entry is adapted in the same way. This process is repeated until all the entries are considered. The learning rate depends on the meta parameter  $\gamma$  that was set in our experiments to 0.001. For each rating in the training set, we computed the error, i.e., the difference between the actual and the predicted rating  $err = r_{uic_1, \dots, c_k} - \hat{r}_{uic_1, \dots, c_k}$ . The larger the error, the bigger is the adaptation of the parameters, and the sign of this error indicates in which direction the adaptation must be performed.

Note that some of the parameters in the model are updated more frequently than others. This is due to the fact, that some users have more ratings, or some contextual factors appears more often. This could make the model unbalanced as more frequent data will be overrepresented in it. To solve the problem, we adapted the learning rate  $\gamma$ , computing  $\gamma^*$  (e.g.,  $\gamma_{b_u}$ ).  $\gamma^*$  is obtained dividing  $\gamma$  by the logarithm of the number of updates of the corresponding parameter. For example, imagine that a user  $u$  has 20 ratings in his profile. In this case learning rate for variable  $b_u$  is  $\frac{\gamma}{\log(20)}$ . We observe that still will be influenced more by the most popular contextual factors in the rating data set, as we normalize with the log of the frequency and not with the frequency. This is important as these are also the most influential factors (as we discussed earlier).

In this learning procedure, the parameters are updated while cycling through the data. In Fig. 4, the model error while executing the learning procedure is shown. On the  $x$  axis, we show the number of runs through the data that we call epochs. In an epoch, all the training data are used once. Here, the training data comprise a random selection of 80% of the total and the remaining part is used for testing. For a baseline reference, we have also plotted the error of a simple model that predicts the user rating for an item as the average of the ratings for the item provided by the other users. This is a non-personalized, non-contextualized approach. It can be clearly seen that the more train epochs are executed, the better is the performance on the training set. However, as usual, we see a different behavior on the test set. At the beginning, the error decreases, but at some point it starts to increase again. This overfitting could be avoided by increasing the value of  $\lambda$  or learning less parameters for the user and item representation. Note that sometimes one can see a small increase in the error. This is the point where the algorithm starts to learn new

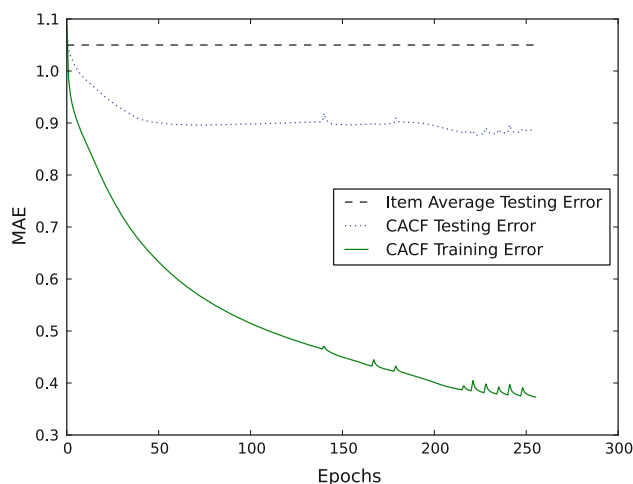


Fig. 4 Training and test errors

parameters for a user and item vectors ( $v_u$  and  $q_i$ ). As this parameters are initialized with random values, this causes a temporal increase in the error. We observe that in our experiments, we used  $d = 10$  factors for the item and user representation  $v_u$  and  $q_i$ , since using more factors did not improve the quality of the model.

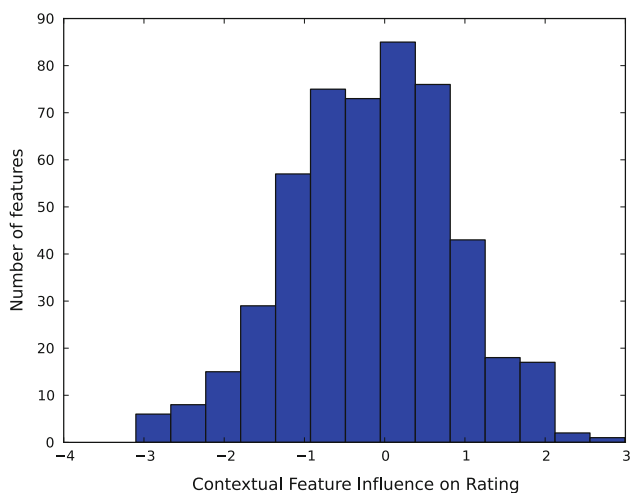
We have also analyzed the performance of some variations of the proposed model. We observed how much each additional set of parameters influences the model accuracy. The model was altered by removing some of the parameters. For example, the model for a non-personalized, but contextualized rating prediction is defined as follows:

$$\hat{r}_{uic_1, \dots, c_k} = \bar{r} + \sum_{j=1}^k b_{ic_j} \quad (2)$$

Here, the rating prediction is the same for all the users and depends only on the current context. Note that if the context is unknown, the model becomes equal to the item average. In addition, we considered a model that provides personalized, but non-context-aware predictions.

Figure 5 shows how the values for the learned parameters ( $b_{ic_j}$ ) are distributed when the non-personalized model is computed [see (2)]. For instance, there are approximately 70 parameters with a value between  $-0.5$  and  $0$ . The distribution of the parameter values is close to a Gaussian distribution with average  $-0.23$ . Most of the contextual factors have small influence on the rating of a user and item pair. Moreover, the contextual factors that we have selected have a mixed effect on the rating prediction; some of them tend to increase the rating for an item and some of them not. But the majority of the contextual factors have a negative impact on the rating. This corresponds to the results reported in Sect. 3.

We estimated the performance of the considered models using repeated random subsampling validation. We did 100



**Fig. 5** Learned contextual influence

splits into training and testing set with the training set containing 90% of the data. We recall that the rating data set consists of 20 POIs, 24 users, and 1,484 ratings (as described in Sect. 3). We compared the proposed context-aware methods with the simple non-personalized prediction model given by the item average and with a user-based CF method [15] that is named “Personalized KNN” in all the Figures. We used the cosine angle between user rating vectors as user-to-user similarity.

Figure 6 shows the mean absolute error (MAE) of the models. The largest improvement with respect to the simple model based on the item average is achieved, as expected, by personalizing the recommendations (MF personalized). This gives an improvement of 6.6%. This improvement is a bit smaller than what has been obtained in other cases comparing personalized vs. non-personalized rating predictions (e.g., [7]). But we observe that a direct comparison of this result with other studies cannot be done since the benefit of personalization is strictly dependent on the quantity of the rating data, i.e., on the specific characteristics of the data set.

The personalized model can be further improved by contextualization producing an improvement of 11.9% with respect to the item average prediction (Personalized MF + Context) and a 5.6% improvement over the personalized model (MF Personalized). Note, that the non-personalized but contextualized predictions improve the accuracy of the item average prediction by 2.1% (non-personalized + Context). The user-based CF algorithm (Personalized KNN) has the worst MAE performance. This can be explained by the fact, that it can use only partial information from the training set, i.e., only the ratings that were gathered without any contextual information. Probably, better approaches could be identified. For instance, it might be that averaging all the ratings in the training set that a user gave to an item—in all the registered contextual conditions—could provide a better guess of the general rating of a user for an item irrespectively of the contextual condition. However, this is out of the scope of our study to design a new non-context-aware KNN algorithm for collaborative filtering using contextually tagged data.

We also measured the ranking performance of the proposed algorithms. Figure 6 shows Precision/Recall curves for all the considered methods. As it was proposed in Herlocker [15], we considered items rated as 4 and 5 as relevant ones. To achieve different recall levels, we computed precision and recall when the system recommends items with a predicted rating higher than  $\{1.1, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.2, 4.5, 4.8\}$ . The clear winner for all the recall levels is our proposed personalized CF that takes into account contextual information. As usual in recommender systems, we are mainly interested in increasing precision, even if this will have a negative effect on recall. Assuming

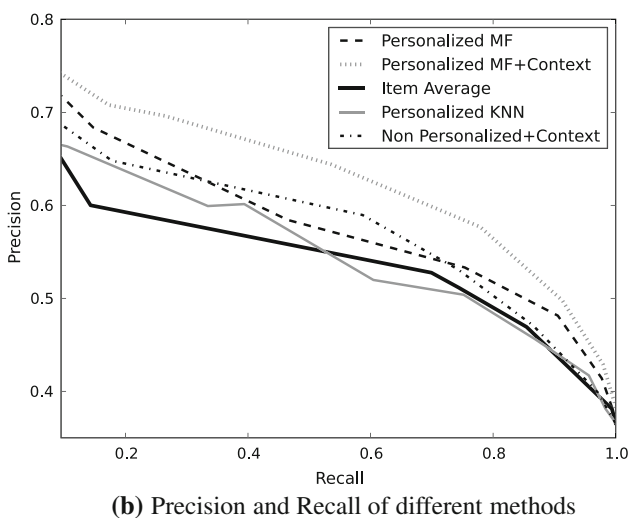
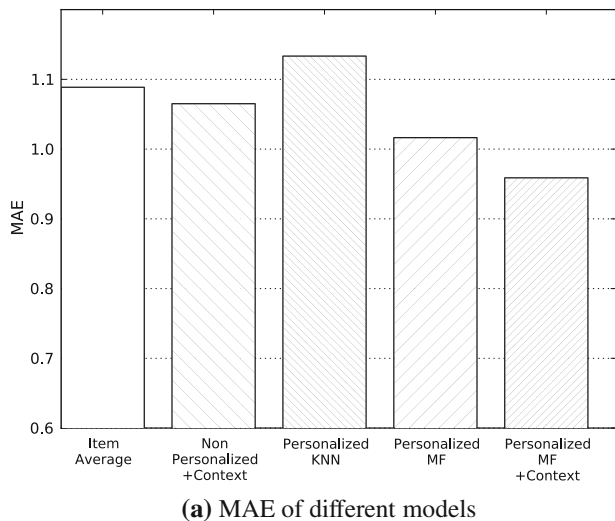


Fig. 6 Performance of different methods

this, the second best method is CF based on matrix factorization followed by the proposed non-personalized method that takes context into account (non-personalized + context). The worst performing methods, similarly to what was observed for MAE, are Item average and user-based CF (Personalized KNN).

In conclusion, we can state that the proposed modeling approach can substantially improve the rating prediction accuracy when taking into account the contextual information.

The above described modeling approach was used also for generating explanations for the recommendations. Analyzing the learned parameters, one can generate explanation based on the values of these parameters. More specifically, given an item  $i$ , user and contextual situation  $(c_1, \dots, c_k)$ , we identify the contextual factor  $j$ , among those specified in the contextual situation, with the highest

impact, i.e., with the largest estimated parameter  $b_{ic_j}$ . Then, we generate a positive explanation for recommending item  $i$  using the contextual condition  $c_j$ . For example, if item  $i$  (let us say Castle Mareccio) is recommended in the contextual situation “temperature is cold and user is with children”, we observe if the parameter  $b_{ic_j} > b_{ic_l}$ , assuming that index  $j$  refer to “temperature” and  $l$  refers to “companion” factors, and we explain that Castle Mareccio is recommended because it is cold. Other examples of explanations are presented in the next Section. Note that in other context-aware CF approaches such as the reduction-based approach [2], there is no straightforward way to generate such explanations.

### 5 Context-aware place of interest recommendations

In this section, we illustrate the main features of ReRex, a mobile context-aware recommender system. ReRex is an iPhone application that allows the user to obtain recommendations adapted to the recommendation context. The recommendations are computed by a server component that implements the predictive model described in the previous section. The iPhone application interacts with the server by means of a custom designed XML-based protocol; the client makes a recommendation request specifying contextual conditions and the server replies with a list of POI recommendations (including pictures and descriptions). The recommendations are determined by two types of information: the in-context ratings for POIs (provided by the data collection described in Sect. 3) and the average rating for the POIs given by the same user population. We will now describe this system illustrating a typical interaction.

In the initial step of the interaction with ReRex, the user normally sets the context of the visit. Figure 7 shows the GUI for enabling and setting the values of the selected contextual factors. The user can switch on/off some of these factors, e.g., “Temperature” or “Weather”. When these factors are switched on, the recommender system will take into account their current values (conditions) by querying an external weather forecast service. For other factors, the user is allowed to enter a value, for instance, the user can specify the group composition as in Fig. 7 (right). The full set of contextual factors is the same as in the web application described earlier, and their values could be found in Table 1. The contextual conditions: time of the day, day of the week, distance to POI, weather, season, and temperature are automatically obtained from other server components and are not entered by the user. The remaining contextual conditions, if the user has enabled them, must be entered manually by the user.

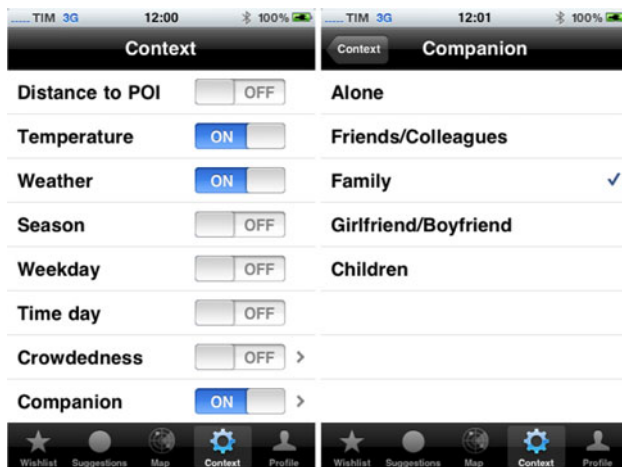


Fig. 7 Context specification UI

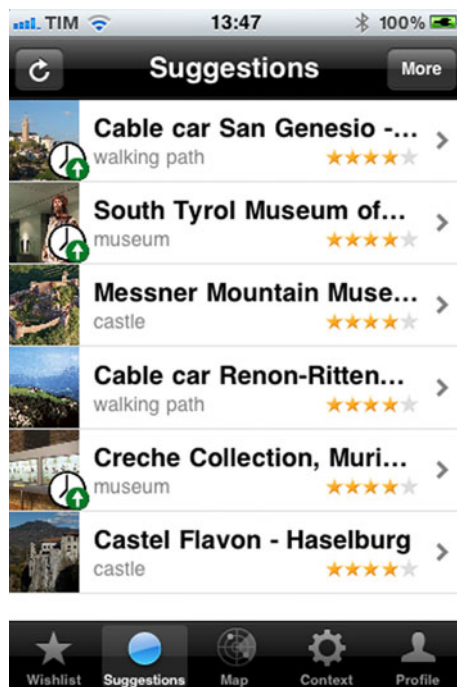


Fig. 8 Suggestions GUI

After the user has enabled some contextual factors and entered the contextual situation, the system can be requested to provide recommendations. A short number of suggestions, namely five, are provided to the user as depicted in Fig. 8. If the user is not happy with these suggestion, he can click on the “More” button (upper right), and more suggestions are provided, after the user has specified earlier the type of POIs he is looking for. In the suggestion visualization screen, the user can touch any suggestion to have a better description of the POI as illustrated in Fig. 9. It is worth noting that some of these suggestions, e.g., the top recommendation “Cable car San

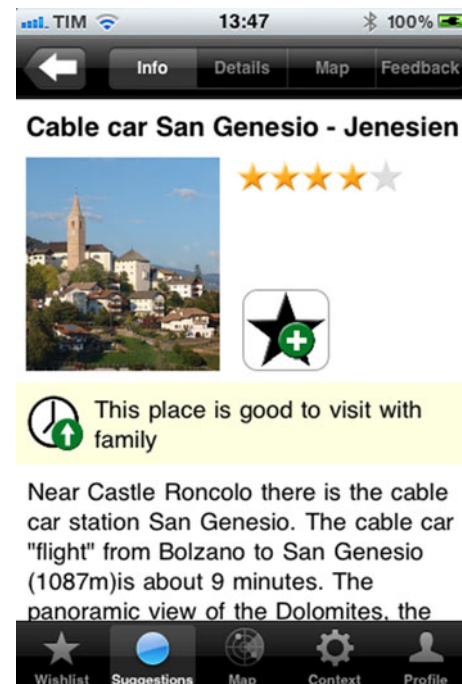


Fig. 9 Details for a suggestion

Genesio” are marked with an icon showing a small clock and a green arrow. These recommendations are particularly suited for the context. In the description of these recommendations, there is an explanation like “This place is good to visit with family”. This is the contextual condition largely responsible for generating a high predicted rating for this item (as it was explained in the previous section).

ReRex offers some additional functionalities. Firstly, each recommended item can be saved in the user wish list. This is achieved by clicking on the star like icon with a green plus shown in the POI details screen (Fig. 9). The wish list interface is similar in appearance to the suggestion list; it shows the selected suggestions and enables the user to browse their details. The user can also remove an items from the wish list. When a POI is visualized, as in Fig. 9, the user can touch one of the buttons in the top part of the screen: details, map, or feedback. Pressing Details provides more detailed information about the item. The Map button shows a Google map with the position of the POI and the current position of the user. Finally, if the user touches Feedback, he is presented with a form where he can enter his context-dependent rating on the selected POI. We note that in the live user evaluation of ReRex, which is described later, we did not use the ratings entered by the users by means of this functionality. To compute the recommendations, we instead used the rating data set acquired with the web application described in Sect. 3, and the predictive model was computed off-line before the experiment took place. The rationale here is that during the

evaluation experiment, we wanted to keep the predictive model stable for all the users.

The last important feature of ReRex is related to its active behavior. If a contextual condition is changing, e.g., the temperature is getting cold, the user is notified in a simple way as shown in Fig. 10. Moreover, it is worth noting that the user can at any moment enter new values for the context status and the suggestion list is updated accordingly. So, ReRex offer always new recommendations and let the user to explore the system and evaluate the quality of the recommendations for different values of the contextual variables.

## 6 Evaluation

Previous assessments conducted on CARS were mainly based on off-line evaluations [4], and the system effectiveness was measured as the precision in predicting correct recommendations, or as the accuracy in predicting the users's true ratings for items (as we did in Sect. 4). In these off-line evaluations, the system predictions for a target user, either the recommendations or the ratings, are evaluated on a test set of items extracted from those already evaluated by the target user. So, in these cases, the system is considered accurate if it correctly predicts the user's evaluation for a subset of items that the user actually rated (test set). To compute these predictions, knowledge contained in a complementary subset of the ratings of the user (training set) is exploited [34]. This type of evaluation, notwithstanding its large diffusion, is severely limited, as the test set, i.e., the items used to measure the system effectiveness, is a small fraction of all possible recommendations the system can make to the users. More importantly, the test set is not a random sample of these (future) recommendations. Hence, the performance of the

deployed system actually could be rather different from the one observed in this type of analysis [15, 25].

For these reasons, we have followed here also a second approach, and we have measured the *subjective user satisfaction* for the system when the knowledge of some contextual variables is exploited: in the computation of the recommendations and in the visualization of these suggestions to the user [34]. We have measured the user satisfaction with a standard usability questionnaire [22]. Moreover, in order to isolate the effect of the contextual factors on the user satisfaction, we decided to generate the recommendations using a prediction model (as described in Sect. 4) that is not personalized, i.e., the recommendations are not dependent on the target user ratings. So, for instance, the system does not exploit how the target user evaluates castles, churches, or markets neither in Bolzano nor elsewhere, before recommending some other POIs in Bolzano, and all the users will receive the same recommendations if they also entered the same contextual conditions.<sup>5</sup>

The recommender system that we have tested integrates two knowledge sources:

1. The general knowledge of a population of users that live in the city. Their ratings have been averaged to produce a popularity based ranking of what is more interesting and significant to suggest to a tourist. This corresponds to the average rating term in the model shown in (2).
2. The overall effect of the context on the evaluation of the POIs. That knowledge is derived from the in-context ratings data of the users that tried the web application described in Sect. 3. This knowledge is used to learn the remaining term in the model shown in (2).

In other words, we implemented a recommender that knows which are the most important attractions of the city and exploits the knowledge of the contextual factors listed in Sect. 5 to adapt its suggestions to the specific context of the request and situation of the target user.

In order to measure the effectiveness of this approach, we developed two variants of our ReRex iPhone mobile recommender system. The first is that described previously, and its rating prediction model is the non-personalized, but context-aware system described in (2). The second variant is neither personalized nor context aware, i.e., there is no possibility for the user to specify his current context, and the UI screens shown in Fig. 7 were removed. The recommendations are sorted according to the item average rating and are essentially those that the Tourist Office

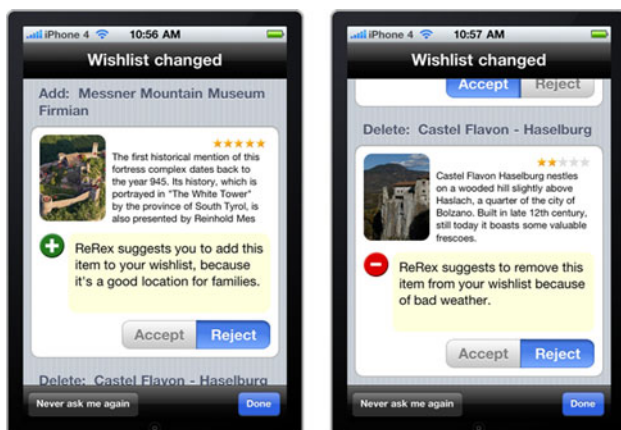


Fig. 10 Notification of a recommendation change

<sup>5</sup> We note that this is actually a big advantage for the proposed recommender, as it does not suffer from the “new user problem”, i.e., the impossibility to deliver a recommendation to a user new to the system, i.e., that has not entered yet any rating [32].

would make to a visitor without any special knowledge of the context of the visit. This is the non-personalized non-context-aware model that is mentioned in Sect. 2.

In this way, we could measure if the proposed context-aware prediction model, which is based on the in-context ratings collected with our methodology, can improve the user perceived relevance of the recommendations with respect to those generated by a non-context-aware model. We note that this is not a trivial question, since the in-context ratings, i.e., the influence of the context on the ratings, are those expressed by a generic population of users, and not those of the target user. Moreover, these evaluations and ratings are based on subjective evaluations of the POIs in contextual conditions that were only *imagined* by the users.

### 6.1 Usability test

The recommender system was trained with the rating data (1664 ratings for 30 POIs) collected with the application described in Sect. 3. The test participants (20 users—working or studying at the Faculty of Computer Science in Bolzano) experimented with both variants of the system, in a random order, and executed, supported by each system, similar, but different tasks. For instance, in one of these two tasks, the user was instructed to: *Imagine you are living in Bolzano. Suppose that it is a cold, rainy Wednesday and you are alone. Select a single point of interest of your choice and add it to your wish list. Then, suppose that your parents are coming to visit you. If needed, revise your previous selection and add another point of interest to your wish list.* A second similar but different task was assigned when the user was asked to try the second variant; hence, each user tried both variants and each variant was used to solve one (randomly assigned) of these tasks.

After the user completed the assigned task using one system, she was requested to fill out a usability questionnaire including the following statements:

- Q1: It was simple to use this system.
- Q2: The interface of this system is pleasant. We wanted to know how easily users can find their preferred points of interest using the system and get information about them.
- Q3: The organization of information provided by the system is clear.
- Q4: It was easy to find the information I needed.
- Q5: The system is effective in helping me to complete the scenario.
- Q6: It was easy to learn to use this system.
- Q7: Overall, I am satisfied with this system.
- Q8: I like using this system.

- Q9: This system has all the functions and capabilities I expect it to have.
- Q10: I am satisfied with the suggested points of interest.
- Q11: I can effectively find interesting suggestions when using this system.
- Q12: I understood the benefit of using the contextual conditions.
- Q13: It was easy to specify the desired contextual conditions.
- Q14: I am satisfied with the provided contextual explanations.
- Q15: I believe that the contextual explanations are useful.
- Q16: The contextual explanations provided by this system are clear.

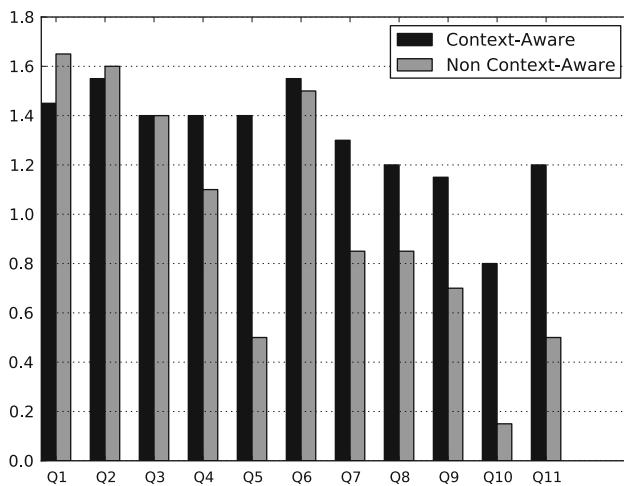
Statements 12–16 were provided only in the questionnaire filled out after testing the context-aware version. The user could express a level of agreement to the statement ranging from  $-2$  (strongly disagree) to  $2$  (strongly agree). These questions were extracted, and slightly adapted to the scope of our investigation, from the IBM Computer System Usability Questionnaire [22]. At the end of the interaction with the two variants, we asked which system the user preferred (Q17) and which one suggested more appropriate points of interest (Q18).

The average response to the first eleven questions, which were asked after testing both interface variants, by the 20 users that evaluated the two system variants are shown in Fig. 11. There is a clear preference of the users for the context-aware variant. Not all the observed differences are statistically significant ( $t$  test); those significant are 5, 7, 9–11.

Hence, we can observe that both systems are considered easy to use and their interface is nicely organized, but the context-aware system is:

- more effective in helping the user to complete the task;
- produces overall higher satisfaction;
- is more complete, in term of functionality;
- suggests POIs that make the user more satisfied;
- suggests more interesting POIs.

Regarding questions Q12–Q16, which were asked only after the context-aware variant was tested, the average results are: Q12 1.30; Q13 1.10; Q14 1.05; Q15 1.50; Q16 1.20. So, also with respect to the specific context-aware aspects, the users were globally positive. We note that among these aspects, the one that it is more interesting is the explanation support. Paradoxically, the quality of the explanations scores low (Q14 1.05) but its importance scores high (Q15 1.50). This seems to indicate that the explanation is a very important component, and the user is particularly sensible to the quality of these explanation; but the formulation of these explanations can be surely further improved.



**Fig. 11** Usability analysis: average replies to the questions

Finally, when the users were requested to directly compare the two variants (Q17 and Q18), 85% of the users preferred the context-aware version, while 95% of the users considered the context-aware recommendations more appropriate. In conclusion, although the number of testers was limited, this evaluation provided a clear evidence that the context-aware recommendations produced by the system were more effective than those produced by the non-context-aware version, i.e., the recommendations normally provided to visitors of the city of Bolzano by the tourist office.

## 7 Related work

Mobile phones are becoming primary platform for information access [31]. They enable users to access information services wherever they are, and especially on the move. This leads to more involving, always online types of communication. Information services are often consumed while traveling, e.g., detailed descriptions of points of interest can be found online. However, portable devices usually have a small display and they offer limited input capabilities [18]. The limited characteristics of these devices make information overload [24] even more critical, and RS technology can play an important role to overcome these limitation. Within the tourism domain, tourist guides received the largest attention [16, 31]. These applications usually support travelers in planning, organizing, and executing a journey, i.e., helping them to find relevant attractions and services or supporting the exploration of an area. These systems usually recommend places of interest (e.g., museums, art galleries, churches) as their primary type of items [13]. Moreover, most current mobile devices are equipped with sensors that enable a context-aware

information access. Thus, mobile RSs can largely benefit from the exploitation of information relative to the users' and items' current context [1, 8, 30, 38]. But RSs can also benefit from information about the media context, i.e., describing the device used to access the system (e.g., a mobile phone or a kiosk) as well as the type of media that are browsed and personalized (ordinary text, or music, images, movies) [12, 19, 39].

The work on Cyberguide [1] opened the research on mobile tourist guides and gave initial contributions to the design of these systems. The proposed model implements spatial awareness of the mobile device, history tracking, but it lacks the recommendation function. A more sophisticated mobile context-aware city guide is COMPASS [38]. The authors use context both as a soft and hard criterion for recommendations. Hard criteria filter out irrelevant recommendations and soft criteria modify the final recommendation list for a user. This work concentrates more on the system architecture and the user study, evaluating the usefulness of a context-aware guide, whereas the recommendation step is described very briefly. The authors propose different prediction strategies for different classes of POIs. As the types of POIs are described by an ontology, the recommendation engine is aware of the class hierarchy of each POI. The user can browse a map indicating her current location and a selection of nearby buildings, buddies, and other objects whose relevance is determined from the user profile. The map with these objects is updated when the user moves (context changes) or when the user profile or the goal changes. In a user study, the authors found several interesting results:

1. In general, most users evaluate context awareness (time and location) as useful;
2. Half of the users do not like the “last time visited” feature, which lowers the predicted relevance of restaurant if user yesterday has visited a similar restaurant;
3. For many users, “the application becomes too intelligent”. It is worth noting that in this system, users can deliberately specify the contextual conditions that matter.

Ahn et al. [5] present an approach to mobile context-dependent recommendations that extends the classical collaborative filtering (CF) method by using information about the user and item location, the time of the recommendation, and the type of the user needs: either hedonic, or neutral or utilitarian. The recommendation process starts by collecting the user position, time, and needs and filtering out the items that are not located close to the user position. Then, in order to apply CF, it searches for similar users, using a particular similarity metric. This metric combines the standard-adjusted cosine metric between the active user

and a neighbor user, with a measure of the similarity of the current time, position, and needs between the two users. The authors collected their own rating data about shopping locations in Korea and compared several CF algorithms with their proposed model showing a slightly better performance (mean absolute error) of their approach. The idea of using the location of the user to tune the user-to-user similarity function has also been exploited by Horozov [17]. In their restaurant, recommender system authors assume that people who live in the same neighborhood are likely to visit the same nearby places. Hence, since people can be correlated in CF only if they have co-rated items, they infer that there is a higher probability of correlating people who live close to each other than correlating people who live further apart.

In this paper, we are marginally concerned with *media context* adaptation, since, with that respect, we performed only simple adaptations of the POI descriptions and recommendation explanations to the mobile device (limited interface). In [39], however, all the three context categories, user preference, situation context, and capability context (as an example of media context) are actually exploited; the authors provide media recommendations for smart phones based on all three mentioned context categories. The model can generate multidimensional output, i.e., it can recommend not only the item but also, for instance, the best media for that item. They used a hybrid recommendation approach, using content-based, Bayesian-classifier, and rule-based methods. They applied this complex model to support media recommendation, adaptation, and delivery for smart phones.

We observe that [28] already tried to estimate the impact of contextual conditions on the user evaluations by asking the user to imagine a given contextual condition. They have shown that this method must be used with care as users rate differently in real and supposed contexts. When the context is just supposed, there is a tendency of the users to exaggerate its importance. In fact, in the first interface proposed in this paper, we were trying to measure only if a contextual factor influences the user's decisions and not the real value of the user's ratings. For instance, we wanted to understand if the proximity to a POI is influential and not how the rating for a precise POI changes as a function of the user proximity. Then, our statistical approach can predict that a contextual factor does influence the user with a given strength (Mutual Information). So, considering only factors with high influence one can reduce significantly the number of false positives, i.e., contextual factors that do not have any relevance for the user decision making task. So, the first phase of our method is proposed as a tool for selecting potentially relevant contextual conditions. Then the true evaluations/ratings of the items under the selected contextual conditions can be acquired in both ways:

1. by asking the users to rate items when they are really experienced in a contextual situation, which is the standard approach and is supported by the mobile application ReRex;
2. by imagining the user to be in a simple contextual situation characterized by a single contextual factor, as we did in the second web-based interface (see Sect. 3).

Hence, the proposed approach gives more flexibility in using several kind of data, i.e., those acquired in supposed context and those collected in real context.

## 8 Conclusions and future work

In this paper, we have argued that context is known to have a large impact on the user decision making and information systems' usage, but often the relationship between context and the computer-supported decision process is unknown and uncertain. Which contextual factor is relevant, in a specific decision making situation, is hard to predict and wrong assumptions may lead to unnecessary and misleading reasoning models.

Hence, in this paper, we have illustrated a methodology and three tools supporting the application of this methodology. Two of the tools were used for acquiring data about the dependency of the users' evaluations for items on the context status and the third is a rating prediction model for providing context-aware recommendations. Finally, the data collected with the second tool and the rating prediction model have been used to build an iPhone mobile recommender system, ReRex that suggests POIs to tourists according to the value of several contextual conditions. ReRex prioritises (or deprioritises) items that are more (or less) suitable for the particular context of the target user. This is achieved by changing the item ranking. In a live user experiment, we have compared ReRex with a similar system, i.e., having the same user interface, but not adapting the recommendations to the context. We have observed a clear preference of the users for the context-aware variant and the users consider the recommendations offered by the context-aware system more interesting and more relevant for their decision task. This indicates that the proposed methodology is able to deliver effective recommendations.

In conclusion, we have shown how the uncertain relationships between context and decisions can be explored and effectively used even if these data, in the form of ratings for items in-context were not as precise as those used by previous CARS systems. In fact, we have acquired the dependency between context and ratings by asking to a population of users to imagine the effect of the context on their evaluations. This is known to produce data that are



different from those collected by observing the evaluations of items after they have been actually consumed or experienced in a contextual situation. But, we have shown, notwithstanding these differences and the limitations of the used data, that the final recommender is perceived by the user as more effective than a non-context-aware system. We have hence shown the practical advantage of the proposed approach. We stress that in our approach, we greatly reduce the cost of the context-dependent rating acquisition phase, since it is easier to collect ratings in contextual situations that are only imagined by the user, still providing useful and effective recommendations.

We observe that we are now applying the proposed approach in a different decision making scenario, namely music recommendation for a group of passengers in a car, to understand to what extent the approach can be generalized to other tasks. Moreover, as future work, we plan to conduct another experiment in which a recommender that, as usual, exploits the ratings of the target user to produce the recommendations is compared to the same recommender that we have illustrated in this article, i.e., one that uses only the contextual situation of the user and exploits a general model, i.e., non-personalized, of the influence of the context on the items’ evaluations. In other words, we want to understand how important is to adapt the recommendations to the context, compared to the classical adaptation performed by recommender systems, i.e., that based on the personal ratings of the user on a set of items.

Another final remark and future work relates to the rating acquisition process. Providing explicit evaluations for items in alternative contextual conditions remains an expensive task for the user, even if the contextual conditions are just imagined rather than experienced. One solution for that is to build more engaging interfaces where users can rate by playing with the system, for instance by rating items in a virtual world (e.g., Second Life). Another solution consists of the exploitation of techniques for extrapolating the item ratings from user actions on the items; e.g., visiting a POI in a contextual situation may be interpreted as a sign that this is a good context for that POI. The challenge here is to filter noisy indicators and to build a reliable predictive model of the rating using the user actions as predictive features.

**Appendix 1: User preferences for categories of points of interest**

In phase 1 of our study on the relevance of contextual factors (see Sect. 2), we measured the relevance of a contextual factor by the normalized mutual information between the response of the user and each contextual factor: the higher the mutual information, the better the contextual factor can explain the response of the user to the questions in the interviews. In the Table 2 to follow, we present an overview of the contextual factors ordered by different POI category:

**Table 2** An overview of the contextual factors ordered by different POI category

Castle	Church or monastery		Cycling or mountain biking		Folk festival, arts and crafts event		Museum		
Distance	1.00	Distance	0.71	Budget	0.66	Distance	1.00	Distance	1.00
Knowledge of surroundings	0.47	Time available	0.58	Time available	0.66	Temperature	0.62	Budget	0.47
Time available	0.47	Mood	0.35	Crowdedness	0.47	Knowledge of surroundings	0.55	Knowledge of surroundings	0.39
Season	0.46	Day time	0.33	Season	0.43	Weather	0.41	Temperature	0.39
Say week	0.42	Transport	0.29	Weather	0.42	Time available	0.30	Time available	0.39
Crowdedness	0.39	Travel goal	0.29	Temperature	0.39	Companion	0.30	Companion	0.34
Day time	0.39	Temperature	0.27	Mood	0.34	Season	0.29	Weather	0.28
Mood	0.23	Companion	0.22	Knowledge of surroundings	0.27	Budget	0.27	Crowdedness	0.27
Companion	0.23	Weather	0.20	Day time	0.27	Day time	0.27	Travel goal	0.25
Travel goal	0.22	Crowdedness	0.18	Transport	0.27	Crowdedness	0.24	Season	0.23
Transport	0.21	Season	0.18	Companion	0.23	Day week	0.21	Day week	0.21
Temperature	0.13	Knowledge of surroundings	0.13	Travel goal	0.19	Travel goal	0.15	Day time	0.19
Weather	0.11	Budget	0.13	Day week	0.08	Transport	0.13	Transport	0.15
Budget	0.05	Day week	0.00	Distance	0.00	Mood	0.12	Mood	0.12

**Table 2** continued

Music event		Nature wonder		Spa		Theater event		Walking	
Crowdedness	0.77	Distance	1.00	Distance	1.00	Time available	0.77	Distance	0.67
Day week	0.67	Day week	0.67	Knowledge of surroundings	0.61	Day time	0.61	Budget	0.58
Time available	0.52	Temperature	0.62	Crowdedness	0.43	Distance	0.60	Temperature	0.52
Mood	0.35	Crowdedness	0.49	Season	0.32	Budget	0.58	Crowdedness	0.49
Companion	0.34	Season	0.40	Time available	0.29	Temperature	0.58	Knowledge of surroundings	0.45
Distance	0.30	Time available	0.39	Weather	0.27	Knowledge of surroundings	0.33	Time available	0.39
Day time	0.25	Weather	0.38	Temperature	0.27	Day week	0.30	Weather	0.38
Budget	0.21	Companion	0.29	Companion	0.26	Mood	0.25	Season	0.35
Transport	0.21	Day time	0.25	Travel goal	0.24	Travel goal	0.19	Day time	0.29
Temperature	0.19	Mood	0.24	Day time	0.24	Season	0.17	Mood	0.21
Travel goal	0.15	Travel goal	0.23	Mood	0.23	Crowdedness	0.13	Day week	0.19
Season	0.14	Transport	0.16	Budget	0.21	Companion	0.10	Transport	0.18
Knowledge of surroundings	0.09	Budget	0.12	Transport	0.21	Weather	0.09	Travel goal	0.14
Weather	0.08	Knowledge of surroundings	0.00	Day week	0.00	Transport	0.07	Companion	0.10

## Appendix 2: ratings for points of interest in different contexts

The Table 3 in this section presents a comparison between ratings of POI in Bolzano without any context and ratings of the same items assuming a certain contextual condition to hold. In order to keep the sampling of the necessary probability distributions by means of the user study in Sect. 3 tractable and to stay in line with the linear predication model for collaborative filtering, we assume the contextual factors to be independent:

$$P(R|C_1, \dots, C_k) \approx \prod_{i=1}^k P(R|C_i)$$

In order to find out which contextual condition  $C_i$  where considered as relevant for their context-dependent rating of POI, we compare  $P(R|C_i)$  to  $P(R)$  with the help of a  $t$  test. With this test, we determine the contextual conditions that induce a statistically significant difference on the average rating of the POIs of a certain category:

**Table 3** The influence of contextual conditions on ratings for POI in Bolzano

Contextual condition	Factor	No. ratings	$p$ value	MCN	MCY	Effect
Castle						
Far away	Distance	15	0.000408107	3.802857143	2.466666667	↓
Winter	Season	11	0.022165228	3.80754717	2.636363636	↓
Cold	Temperature	7	0.06437363	3.897590361	2.857142857	↓
Church or monastery						
Night time	Day time	5	0.045900071	2.583333333	1.2	↓
Far away	Distance	6	0.055531068	2.35	1.166666667	↓
Museum						
Sad	Mood	14	5.11E–05	2.789655172	1.642857143	↓
Activity/sport	Travel goal	6	0.000734366	2.640625	1.333333333	↓
Active	Mood	9	0.000864708	2.644067797	1.444444444	↓
Far away	Distance	25	0.001561059	2.776493256	1.92	↓
Lazy	Mood	6	0.011122893	2.9921875	1.833333333	↓
Half day	Time available	15	0.01491846	2.731958763	1.933333333	↓

**Table 3** continued

Contextual condition	Factor	No. ratings	<i>p</i> value	MCN	MCY	Effect
Warm	Temperature	17	0.015995573	2.701949861	2.058823529	↓
Night time	Day time	15	0.023894116	2.933753943	2	↓
Clear sky	Weather	10	0.029247374	3.085714286	2.3	↓
Rainy	Weather	17	0.034039257	3	3.764705882	↑
With friends/colleagues	Companion	5	0.035327383	2.614457831	1.4	↓
Visiting friends	Travel goal	6	0.036527638	2.694915254	1.833333333	↓
Budget traveler	Budget	10	0.038209253	3.093457944	2.1	↓
Hot	Temperature	10	0.050943446	3.168224299	2	↓
Weekend	Day week	14	0.057615252	2.593333333	2.071428571	↓
Morning time	Day time	10	0.070290314	2.796116505	1.9	↓
Snowing	Weather	6	0.092024039	3.328358209	4.166666667	↑
Walking path						
Night time	Day time	7	4.01E−79	3.78343949	1	↓
Far away	Distance	13	1.10E−05	3.863157895	2.384615385	↓
Cold	Temperature	8	0.000243597	3.8	1.875	↓
Winter	Season	6	0.000274175	3.9140625	2.333333333	↓
With friends/colleagues	Companion	6	0.000849203	3.850746269	4.833333333	↑
Crowded	Crowdedness	8	0.002333888	3.879310345	2.75	↓
Working day	Day week	12	0.00238071	3.943548387	2.75	↓
Half day	Time available	5	0.003014991	4.00990099	1.6	↓
More than a day	Time available	5	0.006695087	3.891891892	4.8	↑
Lazy	Mood	7	0.008503253	4.02919708	4.714285714	↑
alone	Companion	6	0.091310886	3.830769231	2.833333333	↓

## References

- Abowd GD, Atkeson CG, Hong J, Long S, Kooper R, Pinkerton M (1997) Cyberguide: a mobile context-aware tour guide. *ACM Wirel Netw* 3:421–433
- Adomavicius G, Sankaranarayanan R, Sen S, Tuzhilin A (2005) Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans Inf Syst* 23(1):103–145
- Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowl Data Eng IEEE Trans* 17:734–749
- Adomavicius G, Tuzhilin A (2011) Context-aware recommender systems. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 217–256
- Ahn H, Kim K-J, Han I (2006) Mobile advertisement recommender system using collaborative filtering: Mar-cf. In: *Proceedings of the 2006 conference of the Korea society of management information systems*, pp 709–715
- Amatrian X, Jaimes A, Oliver N, Puriol JM (2011) Data mining methods for recommender systems. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 39–71
- Anderson M, Ball M, Boley H, Greene S, Howse N, Lemire D, McGrath S (2003) Racofi: a rule-applying collaborative filtering system. In *Proceedings of COLA03. IEEE/WIC*, October
- Ardissono L, Goy A, Petrone G, Segnan M, Torasso P (2003) Intrigue: personalized recommendation of tourist attractions for desktop and handset devices. *Appl Artif Intell Special Issue Artif* 17(8-9):687–714
- Asoh H, Motomura Y, Ono C (2010) An analysis of differences between preferences in real and supposed contexts. In: *Proceedings of the 2010 workshop on context-aware recommender systems*
- Baltrunas L, Ricci F (2009) Context-based splitting of item ratings in collaborative filtering. In: Burke R, Felfernig A, Schmidt-Thieme L (eds) *RecSys '09: proceedings of the 2009 ACM conference on recommender systems*. ACM Press, New York, pp 245–248
- Baltrunas L, Ricci F, Ludwig B (2011) Context relevance assessment for recommender systems. In: *Proceedings of the 2011 international conference on intelligent user interfaces*, 13–16 Feb 2011, Palo Alto. pp 287–290
- Cena F, Console L, Gena C, Goy A, Levi G, Modeo S, Torre I (2006) Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Commun* 19(4):369–384
- Dunlop MD, Morrison A, McCallum S, Ptaskinski P, Risbey C, Stewart F (2004) Focussed palmtop information access combining starfield displays and profile-based recommendations. In: Crestani F, Jones M, Mizzaro S (eds) *Proceedings of workshop on mobile and ubiquitous information access, LNCS v2954*. Springer, Berlin, pp 79–89
- Herlocker J, Konstan J, Riedl J (2000) Explaining collaborative filtering recommendations. In: *Proceedings of ACM 2000 conference on computer supported cooperative work*. 2–6 Dec 2000 pp 241–250
- Herlocker JL, Konstan JA, Terveen LG, Riedl JT (2004) Evaluating collaborative filtering recommender systems. *ACM Trans Inform Syst* 22(1):5–53

16. Höpken W, Fuchs M, Zanker M, Beer T (2010) Context-based adaptation of mobile applications in tourism. *J Inform Technol Tour* 12(2):175–195
17. Horozov T, Narasimhan N, Vasudevan V (2006) Using location for personalized poi recommendations in mobile environments. In: SAINT '06: proceedings of the international symposium on applications on internet. IEEE Computer Society, Washington, pp 124–129
18. Jones M, Marsden G (2005) *Mobile Interaction Design*. Wiley, London
19. Kaminskas M, Ricci F (2009) Matching places of interest with music. In: Orio N, Rauber A, Rizo D (eds) *Workshop on exploring musical information spaces (WEMIS 2009)*, in conjunction with ECDL 2009. Corfu, Oct 2009, pp 68–73
20. Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: *KDD 08: proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, New York, pp 426–434
21. Koren Y (2009) Collaborative filtering with temporal dynamics. In *KDD 09: proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, New York, pp 447–456
22. Lewis JR (1995) IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *Int J Human Comput Interact* 7(1):57–78
23. Lloyd CJ (1999) *Statistical analysis of categorical data*. Wiley, London
24. Maes P (1994) Agents that reduce work and information overload. *Commun ACM* 37(7):30–40
25. Marlin BM, Zemel RS (2009) Collaborative prediction and ranking with non-random missing data. In: *RecSys '09: proceedings of the third ACM conference on recommender systems*. ACM, New York, pp 5–12
26. McNee SM, Riedl J, Konstan JA (2006) Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*. ACM Press, New York, pp 1097–1101
27. Mcsherry D (2005) Explanation in recommender systems. *Artif Intell Rev* 24:179–197
28. Ono C, Takishima Y, Motomura Y, Asoh H (2009) Context-aware preference model based on a study of difference between real and supposed situation data. In: *User modeling, adaptation, and personalization, 17th international conference, UMAP 2009, Trento, Italy, 22–26 June 2009*, pp 102–113
29. Palmisano C, Tuzhilin A, Gorgoglione M (2008) Using context to improve predictive modeling of customers in personalization applications. *IEEE Trans Knowl Data Eng* 20(11):1535–1549
30. Partridge K, Price B (2009) Enhancing mobile recommender systems with activity inference. In: *User modeling, adaptation, and personalization, 17th international conference, UMAP 2009, Trento, Italy, 22–26 June 2009*, pp 307–318
31. Francesco Ricci (2011) *Mobile Recommender Systems*. *J IT Tourism* 12(3):205–231
32. Ricci F, Rokach L, Shapira B (2011) Introduction to recommender systems handbook. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 1–35
33. Rubens N, Kaplan D, Sugiyama M (2011) Active learning in recommender systems. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 735–767
34. Shani G, Gunawardana A (2011) Evaluating recommendation systems. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 257–298
35. Swarbrooke J, Horner S (2006) *Consumer behaviour in tourism*. Butterworth-Heinemann, 2nd edn
36. Tintarev N, Masthoff J (2011) Designing and evaluating explanations for recommender systems. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) *Recommender systems handbook*. Springer, Berlin, pp 479–510
37. Tversky A, Kahneman D (1991) Loss aversion in riskless choice: a reference-dependent model. *Q J Econ* 106(4):1039–1061
38. van Setten M, Pokraev S, Koolwaaij J (2004) Context-aware recommendations in the mobile tourist application compass. In: Bra PD, Nejd W (eds) *AH*, volume 3137 of *lecture notes in computer science*. Springer, Berlin, pp 235–244
39. Yu Z, Zhou X, Zhang D, Chin C-Y, Wang X, Men J (2006) Supporting context-aware media recommendations for smart phones. *IEEE Pervasive Comput* 5:68–75