Recommender systems for dynamic packaging of tourism services

Michael Schumacher and Jean-Pierre Rey

Institute of Business Information Systems
University of Applied Sciences Western Switzerland (HES-SO), CH-3960 Sierre
michael.schumacher@hevs.ch, jpierre.rey@hevs.ch

Abstract

Based on a case study in Valais (Switzerland), this paper discusses recommender system technologies used to help clients choose tourism service packages online. Different recommender systems are first presented and then analysed in relation to dynamic packaging. Five solutions are finally proposed.

1 Introduction

Tourism packaging offers an important potential for tourism destinations wanting to develop a reservation platform that includes the different services available in a region. Potential clients thus have access to a consolidated offer that may facilitate the planning of their trip.

The eComTour project, carried out by the Institute of Business Information Systems and the Institute of Tourism of the University of Applied Sciences Western Switzerland (HES-SO), analyses the technical requirements and potentials of such a platform with different tourism service providers in the Valais (Switzerland). Different functionalities and a design for the booking creation process were defined. The present document analyses how recommender system technologies can be used to improve personalised packages for clients.

The main proposal is to integrate a recommender system that suggests products to the clients. Several technologies are possible, and the problem must be thoroughly analysed in order to choose the right one. We will also discuss the integration of user profiles and their preferences.

For each of the above cited aspects, we will present the general technologies and discuss their value and applicability for a tourism packaging platform, i.e. their potentials and pitfalls.

The paper is structured as follows: Our case study is presented in section 2. Section 3 introduces recommender systems in general. Section 4 explains collaborative systems, section 5 content-based systems and section 6 knowledge-based systems. In section 7, the use of recommender systems for dynamic packaging of tourism products is analysed. Section 8 is the conclusion.

2 Case study

Valais Tourisme[1] has commissioned us to evaluate the need for and the technical possibilities of an e-commerce platform that offers *dynamic packaging* of tourism services for the destination Valais in Switzerland, with a common entry platform for all tourist services offered in Valais (cf. Fig. 1).

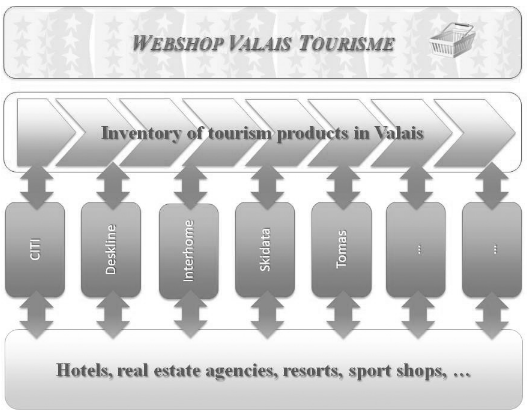
WEBSHOP VALAIS TOURISME
Inventory of tourism products in Valais
CITI
Deskline
Interhome
Skidata
Tomas
...
...
Hotels, real estate agencies, resorts, sport shops, ...

Fig. 1. Tourism e-commerce platform for Valais Tourisme

Dynamic packaging can be considered an electronic system that guides the consumer (or the travel agent) through the design, the booking and the payment of their holiday or trip, according to their needs or desires. The user can dynamically assemble the different components of their choices and then complete the transaction in real time.

Such a dynamic packaging solution requires the creation of an electronic window that combines the entire tourism offer in the Valais. The solution can be used by all service providers and is based on the services developed by them. These services will not be replaced, but integrated into a new or existing e-commerce solution.

Nowadays, it is relatively easy for IT providers to propose dynamic e-commerce solutions if they are based on their own software components[2]. If this is not the case,

1 http://www.valais.ch/

2 Examples in French-speaking Switzerland: eLiberty by Bemore (http://www.bemore.ch), Villars (http://www.villars.ch), etc.

each integration of a new external partner requires a tailored development within an e-commerce solution, generally by means of web services.

The ultimate goal of a dynamic packaging portal is to give clients the opportunity to choose the service package they like, at a suitable price and with the best possible quality. There are a great number of possible package combinations. An intelligent system that recommends individual services for a package or even entire tourism service packages can therefore help the client choose among the many services offered. The aim of this paper is to determine which recommender systems can be used on a package creation platform.

3 Recommender systems

The main objective of recommender systems (RS) (Adomavicius & Tuzhilin, 2005) (Zanker & Jannach, 2010) is to help users find the products that are best suited for them. These systems facilitate the presentation of the available information and assist the client during the purchase process by providing targeted advice. Many online shops such as Amazon.com use recommender systems.

A recommender system can have two main functions: it can improve the quality of choice and decrease the decision time, and, at the same time, increase product sales. It has the following advantages: i) reduction of the cost of research (extraction of data) by offering only suitable products; ii) serendipidity, i.e. the platform can suggest products the client did not know before.

Recommender systems mainly differ in the conditions of the system, which they have to implement: What product data is available and what are the data characteristics? Do users leave feedback, e.g. ratings? Does the website have regular subscribed users or do users only visit occasionally? All these aspects must be taken into account when choosing which RS technique to implement into a platform. Not every method is suitable for every problem. We will hereafter present the main recommender systems and then determine which of their features are essential for a packaging platform.

Formalisation. In this section, we introduce a formalisation of the recommender systems problem such as presented in (Adomavicius & Tuzhilin, 2005):

- C represents all *users* (a potentially large group);
- S represents all *items* (or products) that can be recommended, i.e. hotel bookings, ski rentals (a potentially very large group).

u is a *utility function* that measures if an item s is useful for a user c:

$$u : C \times S \rightarrow R$$

R is an ordered set (e.g. real values). In many application cases, it is not necessary to define a score (utility) for all items, but only for the most important ones.

The aim is to choose for every user *c* of *C* the item *s'* of *S* that maximizes the utility for the user:

$$\forall c \in C, s'_c = \arg\max_{s \in S} u(c,s)$$

The utility of an item for a user can be defined using a function specified by the application or be represented by user ratings.

Every user *c* of *C* can be described by a *profile*, which includes information about their age, their gender, or a simple identifier. Similarly, every item *s* of *S* can be defined by a number of features. A hotel booking, for instance, can be described by the surface of the hotel room, etc.

The *main problem* of RS lies with the fact that the utility of all combinations of *c* and *s* cannot be predicted. It is therefore necessary to estimate this utility for all new cases. This extrapolation is defined by an estimated optimisation function of certain criteria or by an empirical law. Once this law is defined, i.e. once all user feedback has been extrapolated, the first item(s) that maximize their utility can be recommended to the user, such as described in the above function. The different RS methods differ in the way non-existing user ratings are extrapolated.

Main paradigms. Different RS paradigms exist. *Collaborative RS* recommend items that people with similar preferences have liked in the past. *Content-based RS* recommend similar products to the ones the users have liked in the past. *Knowledge-based RS* recommend items that correspond to user needs on the basis of existing items, user profiles (and possibly contextual parameters of these users) and knowledge models. *Hybrid RS* combine several technologies of the other paradigms.

These different paradigms represent multiple and very different technologies. Therefore, the problem (in our case dynamic packaging for the tourism industry) has to be thoroughly analysed in order to choose the technologies that are best suited for this problem. The relevance of the recommendations should also be measured by separating the model training data from the data resulting from their evaluation, and by analysing real recommendations. Quantitative measures of the relevance of these recommendations should then be analysed, i.e. the clients' satisfaction with a recommendation or "online reconversion" of a recommendation (did the user follow the recommendation, did the recommendation result in a purchase?). The following sections present the main paradigms in detail.

4 Collaborative recommender systems

Collaborative recommender systems predict the utility of an item on the basis of the opinion of *other users,* i.e. by using the *wisdom of the crowd.*

A collaborative RC estimates the utility *u(c,s)* of an item *s* for a user *c* using all utilities *u(c_j,s)* estimated by all users c_j of *C* that are "similar" to user *c.* For instance, to make recommendations to a user looking for a package including "wellness" and "snowshoes", packages of similar users can be chosen.

Collaborative filtering. *Collaborative filtering* CF (Goldberg, Nichols, Oki, & Terry, Using collaborative filtering to weave an information tapestry, 1992) is used by most collaborative RS and by online shops such as amazon. CF puts forward two basic *hypotheses:* i) users rate items/products; ii) users have similar behaviour that does not change significantly, i.e. they will like the same things in the future than they like now.

Many technologies have been developed that differ mainly in their definition of *similarity* and *prediction.* According to (Breese, Heckerman, Evans, Gladish, & Pazzani, 1998), two main methods can be distinguished. *Memory-based CF* directly uses the rating matrix to make recommendations, i.e. runtime analyses. These methods are not suitable for large data sources. *Model-based CF* is an offline-based method that learns a model using rating matrices. During runtime, this model is then used to make recommendations. These latter methods are much more run-time efficient, even though the development of the model, which needs regular updating, is expensive.

Memory-based CF: the example of item-based CF. *Item-based CF* is one of the most efficient memory-based methods. It uses the similarity between items (and not users) to make predictions.

To define the utility of an item *i* for a user *u*, this methods searches for all similar items and uses the ratings by *u* for this subset of items to predict the utility of *i.*

We have already mentioned the problem of the complexity of memory-based methods. Another well-known problem is the "cold start", i.e. how to recommend items that have only just been introduced or how to make recommendations to new users. This problem can be solved by obliging new users to rate certain products or by using other methods (demographic data).

Model-based CF: the example of association-based CF. A classical model-based method is the one that uses *association rules (affinity analysis).* This method defines "what goes with what" and is ideally used for online sales of products. Its aim is to define rules such as "if the client books a golfing holiday, he will book a 4- or 5-star hotel and a wellness package".

Various rules can be defined based on these transactions, for instance: {*golf, 4*hotel, wellness*} could mean "*If golf, then 4*hotel and wellness*", but also "*If golf and 4*hotel then wellness*", etc. The part before *if* is called *antecedent,* the part after *if* is called *consequence.* The first step consists in generating all possible association rules using the Apriori algorithm. This generation is exponential to the number of items. To decrease this complexity, Apriori counts up the frequencies, called the supports, of each member item separately. Once all possible rules have been generated, the rules that indicate a strong dependence between antecedent and consequence have to be chosen. The dependence of every rule can be defined by a confidence index, which is calculated as the number of transactions of items of the antecedent and the consequence divided by the number of transactions of items of the antecedents only. The rules that are finally kept are the ones with the largest confidence index.

Discussion. A large number of varieties of collaborative recommender systems exist, which all use different methods (e.g. clustering…). These methods have the advantage of being well-known and of not requiring specific knowledge of the subject area. However, a user group and a sufficiently large database are required.

5 Content-based recommender systems

Collaborative RS do not need any information on the recommended items. However, this information can prove very useful as, for instance, a golf course can be recommended to a person who has already used this service in the past. This is exactly what *content-based recommender systems do.* They use information about items (their *content*) and a user profile that describes what the user likes (*preferences).*

Next, the user preferences have to be learned so that items can be recommended that are similar to the user's preferences. User profiles are either explicitly defined by interrogating the user or implicitly learned using, for instance, transactions. A content-based RS calculates the utility $u(c,s)$ of an item s for a user c using the utilities $u(c,s_i)$ that this same user c has attributed to the items s_i of S that are *similar* to s.

The description/content of an item s (that can be written as *content(s))*, can be described by the attributes of the item. These attributes are typically key words. According to the item description, three content-based RS types can be distinguished: *structured items* (data based on a precise model, typically organised in relational data bases), *non-structured items* (text data) and *semi-structured items* (a mixture of structured and non-structured items). In the first case, a *preference-based approach* can be used (cf. below). In the two other cases, structured data has to be extracted automatically, using *Information Retrieval (IR)* or *Machine Learning* methods, such as the Naive Bayes Classification (Pazzani & Billsus, 1997) or *Support Vector Machines* (Vapnik, 1995).

Similarity measure. For these algorithms, it is thus essential to measure the similarity between items. One of the most common (IR) methods is *TF-IDF*, which encodes a document into a multi-dimensional Euclidean space. TF *(Term Frequency*) measures the frequency of a term in a document (according to the length of the document, this measure is often standardised). IDF *(Inverse Document Frequency)* measures the importance of a term by dividing the total number of documents by the number of documents containing the term and then by taking the logarithm of this division. The resulting matrix is generally very large, with a lot of “holes”. It can be improved by omitting terms such as articles (the, a, etc.).

Profiles and recommendations. Based on this representation of items, content-based RS will create *a content-based profile(c)* for every user, which is derived from their preferences. One method to create these profiles consists in extracting the key words of all items the user has liked. For a total number of *k* set key words, this profile can be described as a weight vector $(w_{c1},...,w_{ck})$, where the weight w_{ci} describes the importance of the item k_i for the user *c*. This vector can be calculated differently, using the ratings (appreciated/not appreciated) of all items.

The recommendation problem for a user can thus be described as follows (using the *k-nearest neighbour algorithm*). For every item *i* of the catalogue, it must be decided if it can be recommended to a user *c*. From the ratings of a group of items *D made* by *c*, the system finds the items out of *D* that are most similar to *i* (using similarity measure). The result, a subset of similar items, is then used to predict the rating of *i* (e.g. by majority).

Preference-based approach. This method directly uses the structured data, i.e. data from different databases from different tourism service providers (dynamic packaging). With the preference-based approach, the creation of recommendations is considered a *constraint satisfaction problem or CSP* (Tsang, 1993). Multi-variable problems are described as constraints between these variables. An optimal solution must be found among all possibilities based on the preferences of a user. We have the following data tuple *(X, D, C, I):*

- *X* represents the attributes $\{x_1, ,x_p\}$ that describe all items; e.g. *X={type, numberOfRooms, surface, ratePerWeek};*
- *D* represents the authorised domain values $\{D_1, ,D_p)$, where every D_i represents the set of possible values for x_i; e.g. $D_{Type} = \{chalet, apartment\}$, $D_{NumberOfRooms} = [1,8]$, $D_{Surface} = [10,300]m^2$, $D_{RatePerWeek} = [0,10'000]CHF$;
- *C* represents the constraints $\{c_1, ,c_p\}$, where every c_i is a constraint function that describes the values that a subset of *X* can have; e.g. $C_{Type,Size}$: if *type = chalet* then $surface > 70m^2$;
- *I* is the set of items that will be recommended to the user; it is part of the Cartesian product $D = D_1 \times D_2 \times ... \times D_p$. e.g. {chalet, 7, 220 m^2, 2’500}.

With the preference-based approach, the user preferences must first be defined (expressed as strong and weak constraints). Based on the *declarative* description of a problem, a CSP[3] solver will find a set of values for the attributes (variables) that fulfil the preferences (constraints).

[3] Many different solvers are available, also as open source (e.g. http://jacop.osolpro.com/).

Discussion. There are different limitations to content-based RS. In order for these methods to work, training data is needed, which is not always possible. These methods also tend to be too specialised and suggest items that are really too similar. The main limitation however is the need for keywords that have no semantic representation (knowledge). We will see in the next section which solutions are proposed to solve this problem.

The preference-based approach has the advantage of being applicable to structured items. Its main problem though is the time-consuming and complex interaction with the users to collect their preferences.

6 Knowledge-based recommender systems.

Knowledge-based recommender systems use technologies based on the representation of knowledge of items and users. Three types can be distinguished: conversational RS, taxonomy-based RS and ontological filtering.

Conversational RS. These recommender systems use *case-based reasoning* (Leak, 1994). Like the FindMe system (Burke, 1997), this method constructs a two-step conversation with the user. First, the system asks the user about their preferences. New preferences are then implicitly construed through *critiques* of the recommendations (e.g. too expensive). The system allows the user to navigate through suggestions without having to know all of the items' criteria. The aim of the system is to resemble a conversation with a salesperson.

It is also possible to add constraints to user preferences, e.g. "if the user books the tourism service A, they must have booked the tourism service B". These constraints can be strong or weak and must be used by the RS.

Taxonomy-based RS. It can be disadvantageous to use only key words to describe the items, as it is very probable that the key words defined by a user cannot be found in the items/documents. A *taxonomy* describing the concepts can therefore be very useful to complete the search. Middleton, for instance, uses a taxonomy to complete user interest profiles (Middleton, Shadbolt, & Roure, 2004). *Collaborative filtering* is then applied to these complements to create recommendations.

Ontological filtering. Even though using a taxonomy proves to be very relevant, Middleton and also (Ziegler, Lausen, & Schmidt-Thieme, 2004) assume that the taxonomy preexists, i.e. that it is static and cannot dynamically react to the addition of new items to the electronic catalogue. This limitation is to be overcome with *ontological filtering*[4] (Schickel-Zuber, 2007). This RS uses an ontology to enrich the catalogue and to deduct the missing preferences. This deduction makes the collection of

[4] Ontological filtering is protected by patents and led to the creation of the Swiss company Prediggo, which deploys its software in a large number of online catalogues.

user preferences unnecessary. To generate recommendations, the knowledge of items and user preferences are used instead of collaborative filtering. If an ontology describing a catalogue is missing, it is automatically learned. In addition, an ontology can be customised for specific users.

Discussion. Knowledge-based RS are used very little, probably because collaborative filtering produces rather good results. One of its disadvantages is the requirement to manually model the subject area into a representation of the knowledge. This disadvantage was overcome by ontological filtering, which automatically constructs the needed ontologies.

7 Recommender systems for dynamic packaging of tourism services: synthesis

Considering the potential of RS, online tourism services and especially dynamic packaging of tourism services, have very distinctive characteristics. Two features need to be pointed out. A user who books a tourism service (a package or part of a package) is, in general, not a regular visitor of the website. That is, his profile is not known in advance, he has no purchase history and he has probably never rated any other items.

Recommendations can be made for accommodation types or for packages. In the first case, the integration site will have different RS for every type of service. In the second case, a recommendation will be made for a service package. These two types of recommendations correspond to two types of architectures that will be discussed hereafter.

Solutions for individual services. If recommendations are made for *individual services*, a system is created where, step by step, recommendations are made for every single service. If, during a booking, the client first wants to book accommodation, an accommodation recommendation will be made. In a second step, a recommendation for winter sports will be made, and so forth. For these individual recommendations, the following RS could be used:

- **Solution 1**: If ratings of individual services can be obtained, a *memory-based CF* can be used. However, this method is necessarily based on a user profile, so that similar user profiles can be found (cf. item-based CF). The user must thus be asked to rate certain offers or to create a basic profile.
- **Solution 2:** With ontological filtering, more precise recommendations can be made, without having to collect user preferences, because this RS uses an ontology to deduce missing preferences.

The implementation of these solutions presents an additional complexity: several RS have to be integrated into a single information system (i.e. the packaging platform). Recommendations for individual services also present the following fundamental problem: they do not take into account the entire booking process with its constraints. If, for instance, a family with four children wants to book a package, the presence of the children will have an influence on the services (type of accommodation, activities, etc.) and on the constraints on these services (e.g. budget constraints). A user should be able to determine an upper price limit for the package. For this reason, recommendations for service packages seem to be the more interesting option (from the client's point of view).

Solutions for service packages. The second possibility is to make recommendations for tourism service packages (Ricci, 2002). There are mainly three different solutions, the last two being the solutions generally found in the literature:

- **Solution 3**: This relatively simple approach consists in making association rules. This method requires an extensive history of purchased packages in order to find out which services have been bought together. The advantage of this method is that the rules can be calculated offline (e.g. every night), so that recommendations can be made very efficiently when the client is online.
- **Solution 4:** Most research suggests the use of conversational RS using case-based reasoning. As with the example presented by (Ricci, Mirzadeh, & Venturini, 2006), a conversation system finds out user preferences or suggests examples that the user rates. The system then suggests products to the user and uses their feedback to improve the recommendations. One of the advantages of this system is the fact that it does not require much user feedback. That is, it can immediately be used (no cold start issues).
- **Solution 5:** The preference-based approach uses structured data, i.e. the kind of data integrated into packaging platforms. This method presents the disadvantage of requiring user profile information. In a second step, a solution is applied to a constraint satisfaction problem. The VIBE system by ConfigWorks[5] is an example of such an application that uses a conversational RS such as CSP to recommend packages in the tourist destination of Warmbad-Villach[6] (Jannach, Zanker, & Fuchs, 2009).

8 Conclusions

In this paper, we have analysed the different recommender systems that could be used for a dynamic packaging application for the tourism industry. After presenting the different methods, we have suggested five solutions for the application of RS methods. Solutions 3, 4 and 5 are, in our opinion, the best options. For an optimal choice, the components of the package and their number would have to be defined more precisely. It could be useful to work in two steps. First, the packaging platform could be developed, web services integrated and all transactions recorded. Then, a thorough analysis of this data could be made and a feasibility study with different RS methods

[5] www.configworks.com
[6] www.warmbad.at

could be carried out in order to develop a functional prototype. This study would imperatively have to measure the calculation time of a recommendation, the precision and the utility of the recommendation, and the cost for the implementation and the maintenance of the system.

9 References

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

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. *Proc. of the 20th Very Large Data Bases Conference* , 487--499.

Balabanovic, M., & Shoham, ,. Y. (1997). Fab: Content-Based, Collabora- tive Recommendation (Vol. 40). *Commnications of the ACM*.

Breese, J. S., Heckerman, D., Evans, C., Gladish, B., & Pazzani, M. (1998). Adaptive Algorithms for Collaborative Filtering. *Proceedings of the 14th Conf. Uncertainty in Artificial Intelligence* .

Burke, R. (1997). The FindMe approach to assisted browsing. *IEEE Expert: Intelligent Systems and Their Applications* , 12 (4), 32-40.

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

Jannach, D., Zanker, M., & Fuchs, M. (2009). Constraint-based recommendation in tourism: A multi-perspective case study. *Information Technology & Tourism* , 11 (2), 139-156.

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2011). Recommender Systems – An Introduction. To appear. *Cambridge University Press*.

Keeney, R., & Raiffa, H. (1993). Decisions with Multiple Objectives: Preference and Value Tradeoffs. *Cambridge University Press*.

Leak, D. B. (1994). Case-based reasoning. (C. U. Press, Ed.) *The Knowledge Engineering Review* , 9, 61-64.

Middleton, S. E., Shadbolt, N. R., & Roure, D. C. (2004). Ontological user profiling in recommender systems. *ACM Transactions on Information Systems* (TOIS) , 22 (1), 54-88.

Pazzani, M. J., & Billsus, D. (1997). Learning and revising user profiles: The identification of interesting Web sites. Machine Learning , 27 (3), 313-331.

Ricci, F. (2002). Travel Recommender Systems. *Intelligent Systems* , 55-57.

Ricci, F., Mirzadeh, N., & Venturini, A. (2006). Case-based travel recommendations. In D. R. Fesenmaier, K. W. Woeber, & H. Werthner, *Destination Recommendation Systems: Behavioural Foundations and Applications* (pp. 67-93). Oxford: CAB Publishing.

Schickel-Zuber, V. (2007). Ontology Filtering - PhD. Thesis (Vol. 3934). Swiss Federal Institute of Technology (EPFL).

Schumacher, M., Helin, H., & Schuldt, H. (2008). CASCOM: Intelligent Service Coordination in the Semantic Web. Basel: *Birkhäuser Verlag*.

Shmueli, G., Patel, N. R., & Bruce, P. C. (2007). Data Mining for Business Intelligence. *Wiley*.

Tsang, E. (1993). Foundations of Constraint Satisfaction. *Academic Press*.

Vapnik, V. (1995). The Nature of Statistical Learning Theory. *Springer Verlag*.

Viappiani, P., Faltings, B., & Pu, P. (2006). Preference-based search using example-critiquing with suggestions (Vol. 27). *Journal of Artificial Intelligence Research*.

Zanker, M., & Jannach, D. (2010). Introduction to Recommender Systems. Sierre: Tutorial at ACM Symposium on Applied Computing.

Zhang, J., & Pu, P. (2004). Survey of solving multi-attribute decision problems. Rapport technique, *Ecole Polytechnique Fédérale de Lausanne*.

Ziegler, C.-N., Lausen, G., & Schmidt-Thieme, L. (2004). Taxonomy-driven computation of product recommendations. *Proceedings of the thirteenth ACM international conference on Information and knowledge management* , 406-415.