Automated Case Generation for Recommender Systems Using Knowledge Discovery Techniques

Patrick Clerkin, Conor Hayes, Pádraig Cunningham

Department of Computer Science
Trinity College Dublin
Patrick.Clerkin@cs.tcd.ie
Conor.Hayes@cs.tcd.ie
Padraig.Cunningham@cs.tcd.ie

Abstract

One approach to product recommendation in e-commerce is collaborative filtering, which is based on data of users' consumption of assets. The alternative case-based approach is based on a more semantically rich representation of users and assets. However, generating these case representations can be a significant overhead in system development. In this paper we present an approach to case authoring based on data mining methods. Specifically, we focus on clustering algorithms. Having demonstrated the feasibility of this approach, we go on to consider what benefits such techniques might confer on the recommendation system. In this context we distinguish three levels of interpretability of cluster formations or concepts, and go on to argue that, while the first two levels offer no immediate advantages over each other in the recommendation domain, moving to the third level allows us to overcome the bootstrap problem of recommending assets to new users.

Introduction

A key role for intelligent systems in e-commerce is product recommendation (Cunningham et al., 2001). Conventionally, there are two alternatives to product recommendation; there is the content-based approach (case-based) and the collaborative recommendation approach. From one perspective, these approaches are opposites since the first is representation-based and the second is representation-less. Alternatively, the difference is one of degree, where the collaborative recommendation approach is based on a raw representation of users and assets and the content-based approach is based on a more semantically rich representation. (Hayes, Cunningham & Smyth, 2001). The objective in this paper is to explore the mechanisms for taking the raw structures on which collaborative recommendation is based and automatically eliciting the more semantically rich cases that can be used for content-based recommendation.

This is worthwhile in order to overcome the shortcomings inherent in both approaches. The content-based approach to recommendation has the disadvantage that the representations on which the approach is based need to be determined at design time. On the other hand, the collaborative approach has bootstrap problems where there is no basis for recommending new items and there is no basis on which to make recommendations to new users.

In this paper, we propose a means of overcoming these limitations whereby the data that underpins the collaborative recommendation is mined to discover appropriate representations to underpin content-based recommendation. We show how clustering can be used to generate case representations that can produce good quality recommendations. However these representations lack interpretability so they cannot overcome the bootstrap problem because a new user cannot use this representation to set up their profile. The paper concludes with some discussion on how this case authoring might be improved to produce case descriptions with interpretable descriptions.

Recommendation

As stated in the introduction, there are two approaches to recommendation on the Web. The recommendation process can be content based as represented by the upper path in Figure 1 where an appropriate representation of the assets and users requirements is determined at design time and recommendation is based on this representation. In the Case-Based Reasoning community this is referred to as case-based recommendation. The alternative lower path in the figure is automatic collaborative recommendation (ACF) which works with raw data on user's ratings and behaviour and uses this data to produce recommendations. The focus of this paper is on how knowledge discovery techniques can be applied to this raw data to establish the appropriate representations for content-based recommendation (see Figure 1). First, we will present brief descriptions of content-based and collaborative recommendation.

Content Based Recommendation

In the next section ACF, a representation-less recommendation process, is introduced; before that, we will describe a CBR-like content-based recommendation system that we can use for comparison purposes.

Table 1 shows a case-like description of a film (movie) and Table 2 shows the corresponding description of a user of the recommendation system. In this scenario recommendation is based on how well a film matches a user's profile. In producing recommendations for a user, the matching score for each film in turn would be determined and the highest scoring films not already viewed would be recommended. As will be clear in the next section, this process has advantages over ACF in working well for assets of minority interest or for new assets and users. However, the major drawback is the problem of coming up with appropriate descriptors such as Genre.

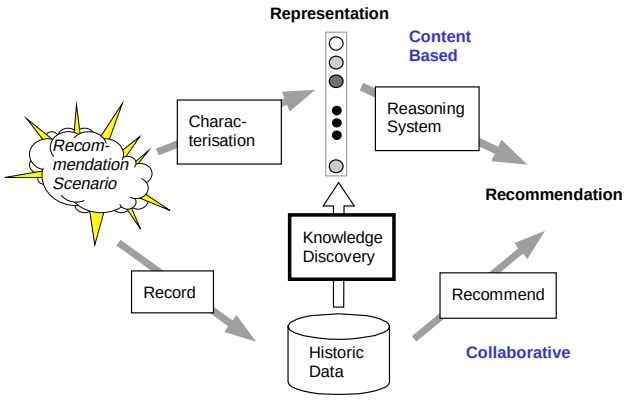

fig. 1 An overview of content-based and collaborative recommendation and the role for knowledge discovery in exploiting the benefits of both approaches.

Table 1. A case-like description of a film for content based recommendation

4W&1F	
Title:	Four Weddings and a Funeral
Year:	1994
Genre:	Comedy, Romance
Director:	Mike Newell?
Starring:	Hugh Grant, Andie MacDowell
Runtime:	116
Country:	UK
Language:	English
Certification:	USA:R (UK:15)

Automatic Collaborative Filtering

The basic idea of ACF can be shown using a simple example. If we have three users who have all shown an interest in assets as follows:

User 1: A,B,C
User 2: A,B,C,D,E, F
User 3: A,B,C,D,E

The high level of overlap indicates that these users have similar tastes. Further it seems a safe bet to recommend assets D and E to User 1 because they are 'endorsed' by Users 2 and 3 that have similar interests to User 1.

Table 2. A case-like description of a user's interests

JB-7	
Name:	Joe Bloggs
Preferred Era:	1988 →
Genre:	Thriller, Comedy, War, Romance
Director:	S. Spielburg, F. F. Coppola.
Actors:	Sharon Stone, Sylvester Stallone, Julia Roberts, Keanu Reeves, Liam Neeson, Andie MacDowell
Runtime:	< 150
Country:	UK, US
Language:	English
Certification:	Any

One of the great strengths of ACF is that, if enough data is available, good quality recommendations can be produced without needing representations of the assets being recommended. The amount of data required depends to some extent on type of data available. In this context, there are two distinct approaches to the ACF idea that are termed *invasive* and *non-invasive*. With the invasive approach the user is explicitly asked to rate assets. This is the approach adopted by PTV (ptv.ucd.ie) (Smyth & Cotter, 1999) for instance and clearly the data contains more information (see Table 3). Amazon.com employs both approaches allowing its customers to rate both their purchases and other items in the catalog. Non-invasive data contains less information and can be noisy in the sense that customers may not like some of their purchases. This can be seen in Table 4, which is a non-invasive version of Table 3. The information that User 2 dislikes asset D is lost in the non-invasive approach. Because of this data noise and loss of information more data is needed to produce good recommendations with the non-invasive approach.

Whether the data available is binary (non-invasive scenario) or contains an explicit rating, the basic structure of the recommendation process will have two distinct phases: First the neighbourhood of users that will produce the recommendations must be determined. Then recommendations must be produced based on the behaviour or ratings of these users – see (Hayes et al. 2001) for details on how that can be done.

Authoring of cases

Case-based reasoning is a powerful approach to problem solving, but building the case-base is an expensive task. Typically, expert knowledge is required to author cases, and this constitutes a bottleneck in CBR development. Let us consider the concrete example that will occupy us throughout this paper. Smart Radio is a web-based song recommendation system which relies on users' ratings of songs. We would like to be able to represent users as cases which we construct on the basis of their song ratings. If authoring these cases entails acquaintance with all of the songs in the database, the enormity of the task is clear. Obviously, it would be of benefit for us to be able to build

cases automatically for this domain. This is what we attempt to do using knowledge discovery techniques.

Table 3. Data for use in ACF where users have explicitly rated assets.

	Song 1	Song 2	Song 3	Song 4	Song 5	Song 6
User 1	0.8	1.0		0.8	0.2	
User 2	0.2	1.0	0.0		0.2	1.0
User 3	0.6	0.6	1.0			0.4
User 4					0.8	0.2
User 5	0.4	0.6	1.0	1.0		
User 6		0.8	0.0	0.6	1.0	0.4
User 7	0.0			0.6		
User 8		0.4	0.0	0.8	0.6	1.0

Table 4. ACF data from Table2 where users have not explicitly rated assets**.**

	Song 1	Song 2	Song 3	Song 4	Song 5	Song 6
User 1	1	1		1	1	
User 2	1	1	1		1	1
User 3	1	1	1			1
User 4					1	1
User 5	1	1	1	1		
User 6		1	1	1	1	1
User 7	1			1		
User 8		1	1	1	1	1

It is important to note at this stage that we do not intend characterising the cases we generate in terms of problem and solution parts. Following Burkhard and Lenz we maintain that cases are *views* upon the underlying raw data. From this perspective, the problem-solution paradigm is just one of several possible conceptualisations of inherently unstructured data. We will leave the cases we construct uninterpreted, since no natural conceptualisation presents itself.

Knowledge Discovery

Data mining is the extraction of implicit, previously unknown, and potentially useful information from data. The objective is to build computer programs that automatically detect regularities or patterns in databases. Useful patterns, if found, should generalise to make accurate predictions on future data. Thus, the final objective of data mining activity is knowledge discovery.

Machine learning provides the technical basis of data mining. It is used to extract information from the raw data in databases. The process is one of abstraction in order to find patterns. Usually, we also require that the system should provide us with an explicit structural description, so as to provide the observer with an explanation of what has been learned and an explanation of the basis for new predictions. In cases where such a description is available, it can take a number of forms. One popular form is, of course, the rule.

As an example, let us consider the effect of running the Apriori algorithm on the Smart Radio database. Each song in the database can be assigned a value for each user. We define a song to be *good* for a user if and only if she gives that song a rating of 60% or more; below this, the song is *bad* for the user. The association rules generated by Apriori are:

1. BeesKnees=good & FreeBeer=good 77
   ==> PadraigC=good 73 conf:(0.95)
2. FreeBeer=good & PadraigC=good 78
   ==> BeesKnees=good 73 conf:(0.94)
3. FreeBeer =good 86
   ==> PadraigC=good 78 conf:(0.91)

BeesKnees, FreeBeer, and PadraigC are user names and the first rule states that if BeesKnees and FreeBeer like a song then PadraigC probably will as well. The number preceeding the ==> symbol indicates a rule's support – that is, the number of items covered by its premise. The number immediately following a rule is the number of items for which its consequent holds true. The number in parentheses is the confidence rating for the rule – i.e., the second figure divided by the first.

The important thing to note about this example is that these rules are immediately comprehensible to humans, even with little or no knowledge of the method that generated them.

However, there do exist data mining systems that do not yield explicit descriptions. For example, neural networks, though sometimes used in data mining, do not provide a comprehensible explanation of their behaviour, since they are 'black boxes'.

Our concern in the current paper is with the application of data mining techniques in the context of recommender systems. We focus in particular on clustering techniques. In describing such techniques and their application, we also describe the nature and degree of their resultant descriptions. This provides us with two bases from which to evaluate the merit of the differing approaches. In the context of a recommender system, we may ask if the application of a given technique improves the quality of recommendation. But we may also ask if the degree of explicitness and comprehensibility of that technique confers any additional advantages to the system.

Clustering

A clustering task has at its goal the unsupervised classification of a set of objects. Classification is unsupervised in the sense that there are no *a priori* target classes used during training. Clustering techniques rely on the existence of some suitable similarity metric for objects.

Numeric taxonomy is one approach to clustering. Objects are represented as vectors of *n* feature values, and the similarity of two objects is defined as the Euclidean distance between them in *n*-dimensional space.

An example of such an approach is the *k*-means clustering algorithm. It forms categories by:

1. Randomly selecting *k* objects, each of which initially represents a cluster mean.
2. Assigning each object to the cluster to which the object is most similar, based on the mean value of the objects in the cluster
3. Updating the cluster means.
4. Repeat steps 2 and 3 until no more reassignments take place.

Note that there are quite a few variants of the *k*-means method. These can differ in the selection of the initial *k*-means, the calculation of dissimilarity, and the strategies for calculating cluster means.

Conceptual Clustering

The clustering algorithms discussed above represent clusters *extensionally*, which means that a class is defined by the enumeration of its members. However, it is also possible to achieve clustering in such a way that it yields *intensional* representations, which means that a cluster will have an associated description defining what it is to be a member of that cluster. Understanding such a description amounts to grasping a concept. Clustering which yields such definitions we refer to as *conceptual clustering*.

An example of such an approach is CLUSTER/2 (Michalski and Stepp 1983), which directly couples the concept characterisation task to the partitioning task (Fisher et al 1991). That is to say, the quality of a given partitioning of a set of objects is not judged merely by a numeric measure, but by the quality of the concepts it produces. The actual concepts yielded by CLUSTER/2 are logical conjunctions of features common to all or most of the members of the underlying cluster; and they are judged according to two criteria, namely, the *simplicity* of the descriptions, and the *fit* of the descriptions to the data.

Incremental conceptual clustering is conceptual clustering with the added constraint that clustering be incremental. This means that objects are not processed en masse, but rather sequentially, one at a time. It also means that the agent does not extensively reprocess previously encountered instances while incorporating the new one, since, without this constraint, one could make any non-incremental method "incremental" simply by adding the new instance to an existing set and reapplying the non-incremental method to the extended set.

The concept formation system, COBWEB, is presented in (Fisher, 1987). COBWEB was initially inspired by research on *basic level* effects. For example, humans can typically verify that an item is a bird more quickly than they can verify the same item is an animal, vertebrate, or robin. Thus, the concept of birds is said to reside at the basic level. COBWEB's design assumes that principles which dictate basic concepts in humans are good heuristics for machine concept formation as well.

COBWEB is designed to produce a hierarchical classification scheme. It carries out a hill-climbing through a space of schemes, and this search is guided by a heuristic measure called *category utility*, which was originally developed by Gluck and Corter (1985) to predict the basic level in human classification categories.

Whereas CLUSTER/2 represents concepts in terms of logical conjunctions, COBWEB represents concepts probabilistically. Instead of defining a concept in terms of a set of values that must be present for each feature of an object falling under the concept, it represents concepts in terms of the conditional probability that a feature value is satisfied, given that an object falls under the concept.

Authoring of cases using clustering

Clustering techniques may be used to generate cases automatically. In this section, we describe our approach in the context of the Smart Radio application. Smart Radio is a web-based song recommendation system which relies on users' ratings of songs. Each song in the database can be assigned a value for each user. Thus, each song may be characterised as an object with a set of real-valued attributes in the range 0.0 to 100.0. It should be noted, however, that in this domain we actually expect that most users will not have rated most songs, and that therefore there is a very large number of missing values. This is important, since the performance of many clustering algorithms falls off dramatically in the face of missing data. For this reason, we have chosen to work initially with the k-means clustering strategy, which - as a distance-based method - is not affected adversely in such contexts.

Using the Weka[1] implementation of *k*-means, we generated seven clusters from the Smart Radio data.

The following is an example line from the data file, representing user ratings for the song, where question marks denote missing values:

100 100 ? ? ? ? 40 ? ? ? ? 80 ? ? 80 ? ? ? ? ? ? ? ? ? ? ? ? ? ? 83 6 ? ? ? ? 60 60 ? ? 20 100 ? ? ? ? ? 20 ? ? ? ? ? 0 ?

Of the seven clusters which resulted from *k*-means, the following is the smallest:

['Break On Through (To The Other Side)', 'Dance Me To The End Of Love', 'Back in the USSR', 'Doo Wop (That Thing)', 'Face a la mer', 'My Brave Face', 'Concertina', 'Around The World', "La Femme D'Argent", 'The Shining Of Things', 'Hound Dog', 'Success Has Made A Failure Of Our Home', 'Hanging Around']

Once these clusters are generated, we construct a case for each user. For each user, we assign a score for each cluster. We do so by calculating the sum of their scores for each of the songs in the given cluster, and dividing by the maxi-

[1] Open source software issued under the GNU General Public License. Available from http://www.cs.waikato.ac.nz/~ml/weka/

mum possible score for the rated songs, i.e., the number of songs rated multiplied by one hundred. The value thus obtained is then normalised with respect to the user's average score.
One of the generated cases is presented here:

```
<casedef casename="BeesKnees">
<attributes>
<attribute name="CLUSTER_1">698</attribute>
<attribute name="CLUSTER_2">726</attribute>
<attribute name="CLUSTER_3">615</attribute>
<attribute name="CLUSTER_4">466</attribute>
<attribute name="CLUSTER_5">157</attribute>
<attribute name="CLUSTER_6">664</attribute>
<attribute name="CLUSTER_7">537</attribute>
</attributes>
</casedef>
```

fig. 2 A case representing the user BeesKnees. The case is represented in CBML (Hayes et al.)

It is important to note at this point that each of the cluster names above, CLUSTER_1 through CLUSTER_7, cannot be understood apart from an exhaustive listing of the members of the clusters those names denote. That is, they are defined extensionally. To be able to provide something more than this, in the form of an intensional definition, we need to employ *conceptual* clustering.
We ran COBWEB on the same dataset as we used above. This time, though, since, COBWEB cannot handle continuous values, we used the nominal values {good, bad}, just as we did when running Apriori. However, it should be noted that an extension of COBWEB called CLASSIT is designed to handle real-valued variables, and could have been used in this situation. The following is an example line from the modified data file:

good bad ? ? good ? good ? ? ? ? ? ? ? good ? ? ? ? ? ? ? ? good ? ? ? ? ? good bad ? ? ? ? good good ? ? ? bad ? ? bad ? bad good ? ? ? bad ? bad ?

Since COBWEB is hierarchical, and since every object is included at every level, there are several sets of clusters that we could choose. The extremes are the root – which consists of one cluster containing all objects - and the leaves – which are singleton clusters equinumerous to the number of objects. We chose to work with the fourth level of the tree, which has five nodes.

The following is an example cluster, named C112:

['Break On Through (To The Other Side)', 'Mr Tambourine man', 'Say hello wave goodbye', 'Hallelujah', 'Unfinished Sympathy', 'Where The Wild Roses Grow', 'Please Forgive Me', 'The Girl From Ipanema', 'Gin Soaked Boy', 'Street Spirit (Fade Out)', "La Femme D'Argent", 'Right Here, Right Now', 'Redemption Song']

As in the previous example we can go on to construct cases for each user, the difference this time being only that there are five rather than seven cluster names to include in each case.
But COBWEB has also constructed a concept corresponding to each cluster, which means that we can not only understand the content of *C112* extensionally, but also intensionally, via the probabilistic representation of *C112*. For Smart Radio the representation of a concept will consist of 108 probability values, since there are 54 users in the dataset and there are two possible values (good/bad) for the rating a user gives a song.

Using the cases for recommendation

As we have discussed in our introduction, the ACF approach to recommendation is a lazy one where all the underlying user data is used. The ACF user profile represents the accumulative consumption behaviour of each user. As such each user profile is a a sparsely populated vector where each slot represents an item in the domain set. Unlike CBR profiles where each case feature has a clear semantic value, sparse ACF profiles are not easily interpretable. Furthermore, generating recommendations from large communities of users becomes a question of scale.
Our implementation seeks to generate case based profiles from the data used for ACF and then make recommendations using CBR retrieval to find nearest neighbours. The advantages over standard ACF are that our user profiles are considerably compressed, and though they do not provide explicit intensional interpretability, examination of the clusters does allow us to assign meaningful descriptors to each. Secondly the implementation addresses a scalability issue in that a *k*-nearest neighbour retrieval can be performed quickly for each target profile.
The technique has three stages:

- Clustering of the user data.
- Building CBR-like user profiles using the generated clusters.
- Making recommendations using the new user profiles.

As described in the previous section we ran a *k*-means clustering algorithm on the SmartRadio data. User profiles were then generated which represented each user in terms of their degree of belonging to each cluster. Figure 2 illustrates such a case. The cases were represented using an XML based case mark up which we had developed earlier (Hayes et al. 1998).
The goal is to produce recommendations for a target case profile. Firstly we load the case base of generated CBR profiles into memory. We use a spreading activation structure where, for each value of a particular feature, a list that contains pointers to all cases that have that value for that feature is created. Each such feature-value node is linked to other feature-value nodes of the same feature type by a simple similarity measure. Activation spreads out from the initially activated feature value-nodes to similar feature value-nodes. The overall activation for a case is the sum of

the activation of its constituent feature value-nodes. We take the *k* most activated cases as the nearest case profiles to the target case.

The recommendation step involves a process of *unpacking* the retrieved profile cases in order to find items to recommend to the target user. We retrieve the full user profiles from the database for each of the *k* cases. Using these profiles we make recommendations in a similar way to a typical ACF process. We make predictions for items in the neighbour profiles according to equation 1 which illustrates the expected value of user *U*'s rating for an item *x*:

$$U_x = \overline{U} + \frac{\sum_{J \in Raters\ of\ x} (J_x - \overline{J})\sigma_{UJ}}{\sum_{J \in Raters\ of\ x} |\sigma_{UJ}|} \qquad (1)$$

The *Raters of x* are neighbours that have rated item *x*, an item that would be new to *U*. In this process an expected value for *U's* rating is aggregated from his neighbours. In the aggregation their ratings are normalized using their average rating, $\overline{J}$ and weighted using their similarity to *U*, σ_{UJ} . In our implementation we use a normalised measure of case activation as a measure of similarity.

We are currently evaluating this technique in comparision to a standard ACF.

Conclusions and research directions

Building on the observations of previous sections, we may distinguish three levels of interpretability of concepts.

The first level is extensional. Given a concept, we may determine to which objects it applies, and, in this way arrive at an understanding of the concept. It should be clear that, generally speaking, this is of limited use in explaining how human beings actually understand concepts since it is hardly ever possible to list all the items a term applies to in the real world. However, as was evidenced above, this is quite practical in the bounded domain of such system assets as songs, and in such limited domains there is no difficulty in this mode of representation for either a computer or a human being.

The second way of interpreting concepts is by representing them by rules, descriptions, etc. We have called such representations intensional. While such an approach certainly offers us more than an extensional definition, it is important to note that we should not expect to be able to relate such intensional definitions to our everyday grasp of concepts. The probabilistic representation of the concept *game* is, presumably, quite complex. We have a perfect grasp of the concept of *game* in our everyday lives, but if we were presented with the probabilistic concept, we would be at a loss to relate the two. To understand how this may be we need to realise that it is an empirical hypothesis that humans represent concepts probabilistically, and it may be hoped that this hypothesis can be confirmed or suitably revised by cognitive scientists. However, it is clear that if there is some such complex representational process at work in the human mind, it is not accessible to consciousness. From this point of view, we may conclude that, in the context of recommender systems, the fact that a conceptual clustering algorithm offers intensional representations is of no distinct advantage.

The third level of interpretability is achieved by mapping clusters to concepts already employed and understood by humans. Having generated cases from cluster membership relations in the manner described above, we are already somewhat closer to this goal than if we only had available the raw user profile data used in ACF. It is possible for someone with knowledge of the songs in the SmartRadio database to get some sense of the music liked by users by examining the cases and the clusters manually. If the first cluster consists predominantly of electronic music, and a given user gives that cluster a high score, then we may safely assume that that user likes electronic music. The ideal would, of course, be for our system to generate such concepts as *electronic music* automatically, but this is, at present, an unrealised goal. However, if we consider some extensions of Smart Radio, we can see how we may mimic such a situation.

The idea is to supply users with a set of checkboxes and radio buttons with comprehensible terms used to describe the songs they are presented with. This might be along the lines of www.allmusic.com:

fig. 3 Music Expert Check encourages user feedback.

In this way, we might be able to automatically construct human understandable labels for clusters. A cluster of songs might then be described in terms of what percentage of its songs are dense and thick, and what percentage light, free and transparent. Of course, we could even include such information in the clustering process, but one of the immediate advantages this approach may entail is directly tied to its mirroring of human semantics. We refer to its proposed efficacy as a solution to the bootstrap problem.

The bootstrap problem is the problem of offering recommendations to a new user. The obvious problem is that until a user has rated several songs, the reliability of the recommendations is poor. But, if we were to ask the new user to describe what type of music they like in relation to the terms we have chosen to characterise our clusters, then we could immediately start to recommend songs from the clusters to which we determine them to have the strongest

membership relation. One strategy would be to recommend the most popular items from that cluster.

References

Burkhard, H-D., (1998) extending Some Concepts of CBR – Foundations of Case Retrieval Nets, in case Based reasoning Technology from foundations to applications, eds Lenz, M., Bartsch-Spörl B., Burkhard, H-D., Wess, S., Lecture Notes in Artificial Intelligence 1400, pp17-50, Springer-Verlag.

Cunningham, P., Bergmann, R., Schmitt, S., Traphöner, R., Breen, S., Smyth, B., WEBSELL: (2001) Intelligent sales assistants for the World Wide Web. *KI - Künstliche Intelligenz*, pp28-31, Vol.1, 2001.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2, 139-172.

Fisher, D.H., Pazzani, M.J., Langley, P. *Concept Formation: Knowledge and Experience in Unsupervised Learning*. Morgan Kaufmann, San Mateo, CA, 1991.

Gluck, M.A., Corter, J.E., (1985) Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference on Artificial Intelligence*, pp. 831-836, Detroit, MI: Morgan Kaufmann.

Han, J., Kamber, M., *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2000.

Hayes C., Cunningham, P., Smyth, B., (2001) A Case-Based Reasoning View of Automatic Collaborative Filtering, to be presented at ICCBR 2001, Vancouver, Canada, August, 2001.

Hayes, C., Cunningham, P., Doyle., M., (1998) Distributed CBR using XML. Workshop for Intelligent Systems and Electronic Commerce Bremen as part of the German Conference on Artificial Intelligence (KI-98). Published in the proceedings of the Workshop, ed. Wilke, W

Konstan, J.A., Miller, B.N., Maltz, M., Herlocker, J.L., Gordon, L.R., & Riedl, J., GroupLens: Applying collaborative filtering to Usenet News. *Communications of the ACM*, Vol. 40, No. 3, pp77-87.

Lenz, M., Auriol E., Manago M., (1998) Diagnosis and decision Support, in Case Based Reasoning technology from foundations to applications, eds Lenz, M., Bartsch-Spörl B., Burkhard, H-D., Wess, S., Lecture Notes in Artificial Intelligence 1400, pp17-50, Springer-Verlag.

Michalski, R. S. and Stepp, R. E. *Learning from Observations: Conceptual Clustering.* In R. S. Michalski, J. G. Carbonell, T. M. Mitchell (Eds.), Machine Learning: An Artificial Intelligence Approach, Tioga, Palo Alto, 1983.

Shardanand, U., and Mayes, P., (1995) Social Inofrmation Filtering: Algorithms for Automating 'Word of Mouth', in Proceedings of CHI95, 210-217.

http://www.acm.org/sigchi/chi95/Electronic/documnts/papers/us_bdy.htm

Smyth, B. & Cotter, P., (1999) Surfing the Digital Wave: Generating Personalised TV Listings using Collaborative, Case-Based Recommendation, in Proceedings of *3rd International Conference on Case-Based Reasoning* eds K-D. Althoff, R. Bergmann, L. K. Branting, Lecture Notes in Artificial Intelligence 1650, V pp561-571, Springer Verlag.

Witten, I.H. and Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations*, San Francisco, CA: Morgan Kaufmann, 1999.