Master in Artificial Intelligence (UPC-URV-UB)

Master of Science Thesis

DYNAMIC LEARNING AND REFINEMENT OF PREFERENCES THROUGH KEYWORDS

David Perelló Cruz

Advisors: Antonio Moreno Ribas, Lucas Marin Isern

September, 2012

Agraïments

Voldria agrair tot el suport que he rebut per part del Dr. Antonio Moreno i el Prof. Lucas Marín durant la realització d'aquest treball, així com l’ajut en aspectes tècnics, aportació d'idees i punts de vista, com la seva predisposició en tot moment.

També vull donar les gràcies als meus companys que en tot moment m'han animat i m'han ajudat quan m'ha fet falta qualsevol cosa d'ells.

Finalment, agrair sobretot a la meva família per estar sempre al meu costat, pel suport costant i l'oportunitat que m'han brindat de realitzar aquests estudis.

Table of Contents

List of figures

List of tables

1 Introduction

We are living with large *information systems* [1, 2, 3], which are any combination of information technology and people's activities that support operations, management and decision making. Due to the astonishing speed of growth of these information systems, sometimes decision making can be a difficult and overwhelming task for the user, who interacts with these large amounts of information and knowledge to extract accurate information according to his/her preferences.

In this scenario, users are constantly in situations in which they have to select an alternative from a set of possibilities. According to their preferences, on the one hand, users may find different environments in which to select an option and, on the other hand, it has a high temporal cost due to the necessity to discover the best alternatives and discard the other alternatives without interest.

To understand a little better this scenario, we can imagine the following situations, such as the search of music, films, books, news, images, web pages, etc. As we can see in all these situations**,** it is always required a prior search among a large set of alternatives in order to locate these that fit the user's preferences. For example, normally each user has a favourite musical style or analyzing more specifically the movies category depending on the age and sex of people, they have different preferences for action, terror, drama, comedy movies, etc.

To manage this tasks in a more optional way and to provide the user with information and knowledge in a more personalized way, the *recommender systems (RS)* where created [4, 5], which implement techniques to filter information that is of interest to a particular user with the goal to assist him in decision-making tasks.

Generally, RS use the user profile to evaluate the characteristics of the alternatives, and seek to predict a ranking according to the weight that the user would give to an item that the system hasn't considered yet.

However, the domain must be taken into account in these recommender systems. As we initially mentioned RS manage large amounts of information and knowledge of different types. There are two main possibilities*: numerical* and *categorical* [6].

On the one hand, we have the possibility to treat *a numeric domain,* as is the case where the user interest is focused on the price of an element, the days to stay in a tourist destination, age, etc. On the other hand, we have the *categorical* characteristics which are the most common, for example, in a public library the different genres of books, authors, publishers, etc. Furthermore, hybrid approaches permit to work with both formats together, giving more flexibility to the system and to the user in deciding how his/her preferences can be expressed.

Multiple-criteria decision making (MCDM) [7] is a sub-discipline of Operations Research that explicitly considers multiple criteria in decision-making environments. MCDM is concerned with structuring and solving decision and planning problems involving multiple criteria. The purpose is to support decision makers facing such problems. Typically, there does not exist a unique optimal solution for such problems and it is necessary to use the decision maker's preferences to differentiate between solutions. This technique is an example of systems that deal with linguistic and categorical domains [8].

Unlike those approaches, this work focuses on the management of unstructured information where we deal with text objects that can come from any source (for example: web content, news, forums, social networks, etc).

RS need to have a *user profile* where the relevant information about the user interests is stored. The goal is to learn this information of the user through each action, behaviour, habits and knowledge. To achieve this aim there exist two type of methods: *implicit* or *explicit feedback* [9].

Explicit feedback is obtained through actions that the user does directly indicating what is more relevant or irrelevant for him. These systems, such as [10, 11] and [12], are simple and they offer a great performance but their main inconvenient is the great temporal cost, and we must also consider that some users are reluctant to spend time giving explicit feedback.

Examples of explicit data collection include the following:

- Asking a user to rate an item on a sliding scale.
- Asking a user to rank a collection of items from favourite to least favourite.
- Presenting two items to a user and asking him/her to choose the best one of them.
- Asking a user to create a list of items that he/she likes.

Implicit feedback is obtained by observing the user behavior. The main drawback of these systems, such as [13] and [14], is the great amount of information that is collected and that the computation needed to obtain recommendations for adaptations.

Examples of implicit data collection include the following:

- Observing the items that a user views in an online store.
- Analyzing item/user viewing times.
- Keeping a record of the items that a user purchases online.
- Obtaining a list of items that a user has listened to or watched on his/her computer.
- Analyzing the user's social network and discovering similar likes and dislikes.

Some *hybrid systems* combine both implicit and explicit approaches [15, 16, 17].

In our work, we use the implicit feedback method in the collection of features. The procedure will be to observe the behaviour of the user through his selection of an item within a set of proposed alternatives. Later we will see this process in more detail.

1.1 Objectives of this work

The main goal in this work is to learn user preferences in situations where the objects to be treated are formed only by textual information and we continuously have information of selections made by the user.

This work has been divided in two major parts: the first one including the *algorithms* and *techniques to rank a set of alternatives*, and the second one including the *techniques to maintain the profile up to date*. The basic structure of these elements is shown in Figure 1.

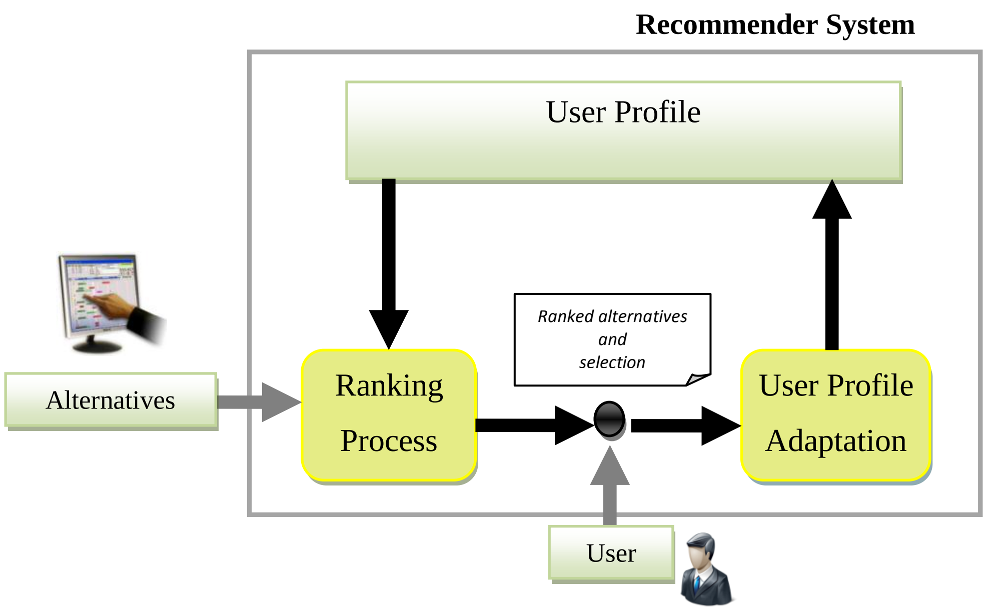


Figure 1: Steps of the recommendation process

Regarding the first part, the goal is to *evaluate an object of type text*, i.e. given the user preferences to assign the degree of potential interest on that object. This will allow us to evaluate the set of alternatives and to sort them according to the user preferences. Concerning the second part, the main goal is to design a *method to update the user profile*, given the user selection from a set of alternatives in the first part.

This method will allow to adapt a user profile in an unsupervised and dynamic way. To achieve these objectives it is necessary to fulfil the tasks discussed in this document and named below in the document organization.

1.2 Document organization

This Master Thesis is organised as follows:

Chapter 2 is divided in two parts. The first one presents the management of the information, studying current methodologies and techniques to represent information about a textual object and to extract this information to store in a structure designed to meet the needs of the user profile. The second part presents the user profile, analyzing different structures, the initialization of user profiles and the process of comparing the user profile with the information that it receives from the user interaction with the system.

Chapter 3 presents the different existing methodologies and strategies for the adaptation of user profiles. On the other hand, it describes the design and implementation of an adaptation method that automatically evolves the user profile according to the information obtained by the user selection among a set of alternatives.

Chapter 4 includes the test and evaluation of the adaptation method explained in the previous chapter in a concrete domain. In this project we have used real news from the British newspaper "The Guardian".

Chapter 5 summarizes a list of conclusions of this work and devises some lines of future work.

2 Management of information and user profile

In that section, we will see how information is represented to have a better knowledge of how it is organized and also the different techniques and methodologies that exist for processing and extracting relevant information.

In this work we have chosen to use the OpenLNP library to get the desired information and a well-known technique for assigning weights called TD-IDF to help reflect the importance of the terms obtained in the previous step by the OpenNLP library.

We also introduce in this chapter the idea of user profiles, another essential element for this work. User profiles are formed by a structure which contains user preferences, for this reason it is another important element.

Finally, we will see how to carry out the comparison between sources of information (a textual document) and the user profile to evaluate the interest of the user on these sources of information.

2.1 Representation of information

Information is anything that can be handled by a system, either as input, or as a result. In the last decade a growth of information available has occurred, especially in the late 90's thanks to the computational power and excess of textual information existing in electronic form due to the Internet and the World Wide Web. Already in the year 2000, there were over 800 million pages that covered most topics of human knowledge and, since then, the growth rate has been increasing.

If we analyze this information we can distinguish between structured information and unstructured information.

Structured information [18] sources show a homogeneous structure for each record, include the same data and fields as well as directories, transactions, records, databases, etc. Search systems and information retrieval systems are friendly and permit processing with relative ease. To obtain structured information it is necessary to provide the structure of the database and possess a tool used for indexing and classification.

When the information is structured it is usually relatively easy to search. Generally, it is easy to tell to a programs something like this “give me the list of record numbers in the Customers table where total sales is greater than 1000 and the name begins with the letter A”. So this is what we mean by *structured documents* and *structured information*: information whose parts are identified, making it accessible to the human and computer processing.

Meanwhile, unstructured sources [19, 20], unlike the above they do not present a homogeneous structure. It refers to information that either does not have a predefined data model and does not conform to relational tables. Unstructured information is typically text but it can contain data such as dates, numbers and facts. We can obtain this information from magazines, books, summaries of events and many other documents. It is therefore difficult to process.

Contrary to popular belief, the amount of unstructured information available today has a magnitude greater than the amount of structured information. Unstructured information does not fit easily into the "columns and rows" concept of relational databases, for instance the text of a note can contain a paragraph or 100, a book may have chapters of different lengths, a technical description of an Airbus aircraft requires a few hundred boxes of drawings and pages of text, etc.

Although there may be some kind of visual structure for a human reader so as to easily find the date, whether it is on the left or right or even find the topic having read a couple of paragraphs. These tasks are more complex for a program.

Having established the two types of information we can find, much of the information may be composed of a mixture of *structured information* and *unstructured* [22, 23]. There may be some structured pieces such as date, author, etc, but at the same time we may have one or more paragraphs with a description. For all practical purposes, a mixture should be considered unstructured. Information systems that can handle unstructured information generally can handle structured information and not always. The opposite is not true in general.

For this project the idea is to deal with unstructured information, where the focus is the treatment of the relevant content of a text. We mean by content any object that has some associated text. This text can be in form of title and body-related words, questions and answers or just a title associated with a photo or video. The content may have been developed by a professional or by the users of a website. We can distinguish different web content such as articles, products, blogs, forums, and much more that we can see in Table 1.

Content Type	Description	Source
Articles	Text on a particular topic. Contains a title, a body and sometimes subtitles.	Are professionally created or by users, news, aggregates of other sites.
Products	An object sold on a website. Usually consists of title, description, keywords, reviews, ratings and other attributes such as price, the manufacturer and its availability in certain geographical areas.	Created by the website or by users (eg. EBay).

Terms of classification	Terms ad hoc with keywords or associated tags. They are created to facilitate browsing.	Created professionally or by users.
Blogs	Personal daily online written to share with other users.	Site administrators or generated by the users.
Wikis	Online collaboration tools where users can edit, add or delete pages in a web.	Normally generated by the user.
Groups and message boards	Sites where you can post questions and others can answer them and qualify them for their usefulness.	Usually generated by users although more complex answers may require the help of an expert working for the website.
Media Content	Videos, photos, music, etc.	Created professionally or by users.
Polls	Questions posed by a user, the answer being a handful of options.	Created professionally or by users.
Search Terms	Queries made by users.	Generated by the users.
Profile pages	Profile page of a user. Usually created from a list of preferences or information about a user.	Generated by the users.
Job Tools	Available on the website.	Created professionally.
Chat logs	Transcripts of online chats.	Experts to users and vice versa.
Banks of questions and answers (FAQ, Frequently Asked Questions)	Answers to questions posed by users to a web site manager.	User-generated questions and answers generated by experts.

Reviews	Reviews about an object, which can belong to any of the previous contents.	Created professionally or by users.
Classifieds	Ads with a title and a body. May optionally have associated keywords.	Created professionally or by users.

Table 1: Content types

2.1.1 Preprocessing of Unstructured Text

There are many processing libraries for information management systems centered in Natural Language Text which are very useful to facilitate the recognition of structures in textual content [27].

• *Sentence analyzer and tokenizer* that identifies the boundaries of sentences in a document and decomposes each sentence into tokens. Tokens are obtained by splitting a sentence along a predefined set of delimiters like spaces, commas, and dots. A token is typically a word or a digit, or a punctuation.

• *Part of speech tagger* that assigns to each word a grammatical category coming from a fixed set. The set of tags includes the conventional part of speech such as noun, verb, adjective, adverb, article, conjunct, and pronoun. An example of POS tags attached to a sentence appears below in Figure 2.

• *Parser* that groups words in a sentence into prominent phrase types such as noun phrases, prepositional phrases, and verb phrases. The output of parsing is a parse tree that groups words into syntactic phrases. A context free grammar is typically used to identify the structure of a sentence in terms of its constituent phrase types.

• Dependency analyzer that identifies the words in a sentence that form arguments of other words in the sentence. The output of a dependency analyzer is a graph where the nodes are the words and the directed edges are used to connect a word to words that depend on it.

We can find many NLP libraries freely available for download such as IBM's Language ware, libraries from the Stanford NLP group, and several others listed under the OpenNLP.

2.1.2 OpenNLP

We have used a library of OpenNLP [28], that is an organizational center for open source projects related to natural language processing. OpenNLP is a Machine Learning based toolkit for the processing of natural language text. It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution. These tasks are usually required to build more advanced text processing services.

In this paper we focus on textual content of newspaper news and we must know well this domain to filter the content of the document parts that are considered more relevant like the title, subtitle, the body of the story, and the section of words calves, etc, removing irrelevant content such as some sections, advertising, menus, etc.

After filtering the most relevant textual content of the news, we will proceed to the application of corresponding model the OpenNLP to obtain the desired information on a text input. Thus we can apply Speech tagger to assign to each word a grammatical category. Of all the textual content analyzed the main information we are interested in carrying out this project are the words with the following grammatical categories regarding names.

Type	Description
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural

Table 2: Grammatical categories regarding names

Below we can see an example of part of Speech Tagging with OpenNLP:

"Pierre Vinken , 61 years old , will join the board as a nonexecutive director Nov. 29."

*Pierre_**NNP** Vinken_**NNP** ,_, 61_**CD** years_**NNS** old_**JJ** ,_, will_**MD** join_**VB** the_**DT** board_**NN** as_**IN** a_**DT** nonexecutive_**JJ** director_**NN** Nov._**NNP** 29._**CD***

Figure 2: Example of Speech Tagging with OpenNLP

In this case the information that we will obtain corresponds to the following words: Pierre Vinken, years, board, director and Nov.

2.1.3 TF-IDF

As expected, the number of words related to names that can be obtained from a document depends on the amount of text but it will usually be a large quantity. To deal with a limited number and manage those words of more importance, we apply the method of assignment of weights for each word.

The technique used to assign weights is called TF-IDF [29, 30, 31], in which the weight of the terms is mainly based on Term Frequency (TF) and Inverse Document Frequency (IDF).

The TF-IDF weight (term frequency–inverse document frequency) is a numerical statistic which reflects how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control the fact that some words are generally more common than others.

Term frequency is the frequency of occurrence of the term in the document.

$$tf(t,d) = \frac{t}{f}$$

with:

- t: term that appears in document d.
- d: document processed.
- f: total numbers of words of document d.

The *inverse document frequency* is a measure of whether the term is common or rare across all documents of the corpus. It is obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

$$idf(t, D) = log \frac{|D|}{|\{d \in D : t \in d\}|}$$

with:

- $|D|$: cardinality of D, or the total number of documents in the corpus.
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears $tf(t, d) \neq 0$. If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator of the formula to $1 + |\{d \in D : t \in d\}|$.

Then the *tf-idf* measure is calculated as follows:

$$tf - idf(t, d, D) = tf(t, d) * idf(t, D)$$

Figure 3 shows an example of some of the terms that would be found on a concrete news about horses and their associated *tf-idf* value, given a certain document corpus.

NEWS: 1

Section

Sport

Date

2012-04-08T10:48:29Z

Title

Talking Horses | Chris Cook

URL

http://www.guardian.co.uk/sport/blog/2012/apr/08/horse-racing-tips

Keywords

Horses_0.12645571998816815

bets_0.10307266163487579

today_0.10278969816059036

hurdle_0.0765351564694872

Anthony Devlin_0.0699801352122313

future_0.0699801352122313

novice_0.0699801352122313

chase_0.0699801352122313

Leopard stown_0.0699801352122313

....

EndKeywords

Figure 3: Information obtained

2.2 User Profile

The user profile is the basic element to design a recommender system since its success will depend, to a large extent, on the ability to represent the user's current interests. A user profile is a collection of personal data associated to a specific user that refers to the explicit digital representation of a person's preferences and can also be considered as the computer representation of a user model.

A profile can be used to store the description of the characteristics of person which can be exploited by systems taking into account the persons characteristics and preferences. For instance, profiles can be used by adaptive systems that personalize the human-computer interaction.

Profiling is the process that refers to the construction of a profile via the analysis of a set of data. In order to generate a user profile, we must define the profile structure to store the corresponding information about the interests of the user. In this section we can see how the profiles are structured and represented, as well as the generation of the initial profiles.

2.2.1 Structure of the user profile

To store the information relevant to users the first step that we must do is to define the structure of the profile. How we can appreciate in [33] several approaches have been taken to represent user profiles, such as a history of purchases, web navigation or e-mails, an indexed vector of features, a n-gram, a semantic network, an associative network, a classifier including neural networks, decision trees, inducted rules or Bayesian networks, a matrix of ratings and a set of demographic features.

History-Based Model. The main goal of user modeling is customization and adaptation of systems to the user's specific needs. It stores a list of activities resulting from the interaction between the user and the system, for example a list of purchases or a web navigation history. We can find this model on online shops such as Amazon.com or CDNow.

In the *Vector Space* Model, items are represented with a vector of features, usually words or concepts, with an associated value. This value can be a Boolean or a real number. The Boolean value represents the presence of the value of the feature, and the real number represents the frequency, relevance or probability of the feature, which is calculated using information indexing techniques. For example, Webmate [34] utilizes a multiple feature vectors representation. The basic idea is to represent each document as a vector in a vector space so that documents with similar content have similar vectors. Each dimension of the vector space represents a word and its weight as we will see later, this is the option that has been chosen in this work.

In the case of *Weighted N-Grams*, items are represented with a net of words with weights in the nodes and edges. For example, PSUN [35], based on the assumption that words tend to occur one after another a significantly high number of times, extracts fixed length consecutive series of n characters and organises them with weighted links representing the co-occurrence of different words. Therefore, the structure achieves a contextual representation of the words.

Weighted Semantic Networks [36] are able to store the meanings of words, so that a human-like use of these meanings is possible. In IfWeb [32], a semantic network base contains a collection of semantic networks describing a typical pattern of topics of interest to the user.

An *associative network* consists of a set of nodes which represent primary terms, concepts or words, in which a user is interested. A set of weighted links establishes the organization of these terms into relevant phrases. Associative networks differ from the semantic networks because these have different generic link types such as synonymy, superclass-subclass, and also possibly disjunctive and conjunctive sets of links. In contrast, associative networks have only a single link type, a weighted edge, the semantics being implicit in the structure of the network and the parameters associated with the processing [37].

Classifier-Based systems using a classifier as a user profile learning technique retain the structure of the classifier as a profile. This is the case in neural networks, decision trees, inducted rules and Bayesian networks.

User-Item Ratings Matrix: some collaborative filtering systems maintain a user-item ratings matrix as a user profile. The user-item ratings matrix contains historical user ratings on items. Each cell (*u*, *i*) of the matrix contains a rating representing the evaluation of the user *u* of the item *i*, and an empty value if there is no evaluation.

Demographic Feature systems create a user profile through stereotypes. Therefore, the user profile representation is a list of demographic features which represent the kind of user.

For this project we will have a user profile with the typical structure of the Vector Space Model, where items will be represented by a vector of features, which represent the most relevant words obtained from each textual context analyzed as described above in section 2.1. These words have an associated value, in this case a real number representing the importance for the user of that word. We can see a example in Table 3.

Terms	Values
ω_1	v_1
ω_2	v_2
.	.
.	.
.	.
ω_n	v_n

Table 3: Preferences and values in the user profile

2.2.2 Values of preferences

After analyzing the structure of the user profile in the previous section, we will now see the values that we have associated with the preferences. There, exist several forms to represent this value:

- *Numerical values*: preference values are given in a numerical range.
- *Categorical values*: preferences are given in linguistic terms.

We will deal with *numeric values* to define the degree of preference for the user, specifically for this work the value of preferences v_{pref}, is a numerical value in the interval $[-100,100]$. User preferences about a term are based on a single numerical value of preference, which can be denoted $v_{pref} \in [v_{min}, v_{max}]$. Depending on how the preference value is chosen, there are several interpretations of user preferences.

- A *neutral* interest, in which $v_{pref} = 0$
- A *positive* interest, in which $v_{pref} > 0$, will be greater if it tends to v_{max}
- A *negative* interest, in which $v_{pref} < 0$, will be greater if it tends to v_{min}

We might also have chosen a different numeric range for the definition of numerical variables, as e.g. as:

- Another range, $v_{pref} \in [0,200]$, $v_{pref} \in [0,100]$, etc
- Normalized range $v_{pref} \in [0,1]$.

Another way of representing the interest of the preference of the user profile would have been with *categorical values*. The preference domain is a set of numbers where each one represents a value of preference. To express a number in words we need to translate these numerical values into a set of linguistic descriptors [39] and this is done by membership functions. These systems are more flexible, but they are not as accurate because they handle different numerical values with the same categorical value.

2.2.3 Generating initial profiles

After defining how the user preferences are stored it is necessary to think about how to get the initial data for the profile. Below we will see the alternatives:

- *Empty profile*. These systems that begin with a structure of empty profile and are update with an automatic learning method through the analysis of the interaction with the system.
- *Manual.* The system asks users to register their interests in the form of keywords, topics and so on. One of the advantages of this method is the transparency of the system behaviour, but on the other hand, one problem with this method is that it requires much effort on the part of the user and that people cannot necessarily specify what they are interested in, because their interests are sometimes still unknown.
- *Stereotyping.* The user model is initiated by classifying users in stereotypical descriptions [38], representing the features of classes of users. Typically, the data used in the classification is demographic and the user is asked to fill out a registration form: record data (name, address, etc.), geographic data (area code, city, etc), user characteristics (age, sex, etc.), etc. The main problem of this technique is the difficulty of acquiring personal data from the

users because users often withhold personal data or provide false information.

- The *training* set is a collection of user interaction examples which is used to infer the initial user profile. One practical way to establish the training set is to ask the user to rate some concrete examples as relevant or irrelevant to his interests. This mode has the advantage of simplified handling but it has the disadvantage that someone has to select the examples which are not always representative and the results are less precise.

For this work we will start with an empty initial which will be filled through the analysis of the user's interaction with the system until a concrete profile with the user's preferences is obtained.

2.3 Comparing the user profile and a textual object

In this scenario, after having analyzed the information and its characteristics in section 2.1 and the structure of the user profile in section 2.2, we will see how the comparison of the profile and an object is performed so that later this process can be used to improve in a intensive way the user profile.

Decision problems typically appear in mathematical questions of decidability, that is, the question of the existence of an effective method to determine the existence of some object or its membership in a set. It is often the case that it is necessary to compare objects in such models, basically in order to either establish if there is an order between the objects, to establish whether such objects are “near” or other cases that are more complex than a simple 'yes' or 'no'.

Comparing two objects can be seen as looking for one of the following situations [40]:

- Object *a* is “before” object *b*, where “before” implies some kind of order between *a* and *b*, with such an order referring either to a direct *preference* (*a* is preferred to *b*) or being induced from a measurement and its associated scale (*a* occurs before *b*, *a* is longer, bigger, more reliable, than *b*).
- Object *a* is “near” object *b*, where “near” can be considered either as indifference (object *a* or object *b* will do equally well for some purpose), or as a similarity, or again could be induced by a measurement (*a* occurs simultaneously with *b*, they have the same length, weight, reliability).

From a decision aiding point of view, we traditionally focus on the first situation. Ordering relations are the natural basis for solving ranking or choice problems. The second situation is traditionally associated with problems where the aim is to be able to put together objects sharing a common feature in order to form “homogeneous” classes or categories (a classification problem).

On this work we focus on the first situation, as we will need to be able to evaluate and rank a set of objects according to their similarity to the user profile.

As we saw earlier in this document, we will have a user profile that consists of a set of terms and their value indicating the respective degrees of interest, that uses a numerical scale to express the level of preference of each term. And also, on the other hand we have seen in figure 3 the data structure to extract of the different documents and sources of textual content. We can conclude that we have two objects, the user profile and the set of alternatives.

In this work the objects that are treated are always formed by textual content, in which we will focus on the most relevant terms. In the case of the user profile these terms will lead to an associated numerical value of the preference.

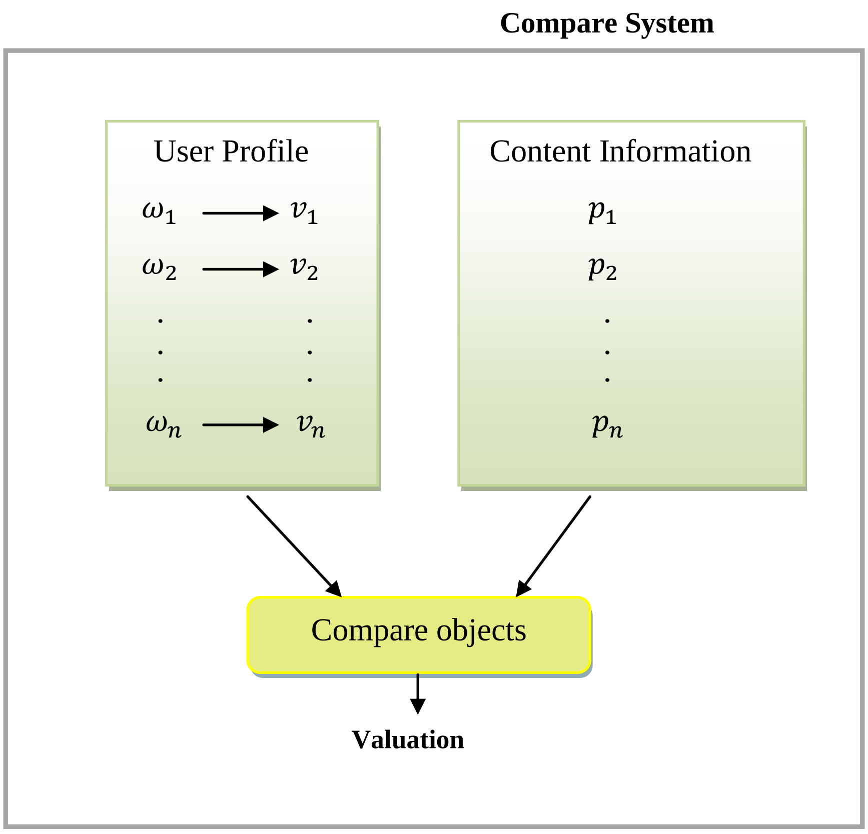


Figure 4: Compare user profile with information source

In our case we are interested in comparing the contents of the user profile with the content of the information sources. The resulting valuation is given by the following formula:

$$Valuation = \sum_{i=0}^{t} v_i \quad if\ \omega_i \,\mathcal{E}\, p_i\ .$$

with:

- v_i: preference of term, in user profile.
- ω_i: term of user profile.
- p_i: term of textual object.
- t: number of terms in the user profile.

The Valuation is the result of the summation of all the preference values of the terms of the user profile that appear in the seat of relevant terms of the analysed text.

2.4 Conclusions

In this chapter we have seen that there are large amounts of information sources which provide largely unstructured information. Focusing on this type of information there exist different techniques for processing unstructured text and we have taken particular use of the OpenLNP library and the TD-IDF method to extract the relevant information to do this work.

In addition, we analyzed the different structures of user profiles and different initializations. In this scenario, we have opted for an empty initial profile, which we will update according to the user preferences. We have also seen the structure of the profile formed by these terms and their respective valuations. We have also explained how piece of text will be evaluated to assess if it fits the user's preferences.

In the next section, we will see in detail the process of adapting the user profile and the different techniques for these cases, where will be crucial the valuation provided by the comparison of the user profile with the information sources to adapt the user profile according to his preferences and provide accurate recommendations.

3 Automatic learning of user interests

As stated before, in order to maintain a current and accurate profile of the user it is necessary to extract relevant information from the feedback generated from the interaction of user with the RS. Now let's see how to manage this information and how to learn automatically the preferences of the user's profile.

In this section we will see different adaptation techniques of the user profile as well as the automatic learning procedure from the information received from the feedback of the selected alternative to update the user profile preferences.

The principal idea is that the adaptation process should be able to deal with some questions internally, like: “Are the good alternatives proposed to the user?” or “Why the alternative selected by the user does not have a sufficiently good score to be ranked in the first place?”.

3.1 Adaptation of the user profile

Adaptation techniques of the user profile are based on the user interaction with the system and keep updated the information represented in user profiles through the analysis of the feedback obtained from the user interaction. Several approaches for profile adaptation indicated in [41] can be found on current recommender systems distinguishing between manual and automatic techniques.

On the one hand, we have the manual adjustment of the profiles which require that the user is who changes their profile manually, when he is interested in the update. An example of it is SIFT Netnews [42], in which when a user wants to include/exclude one of the interests contained in his profile, he has to modify it manually. This approach has the same problems as the manual initial profile generation: it requires an effort from the user and, furthermore, people are not able to specify accurately their own interests because sometimes they are unknown.

On the other hand, we have the automatic adjustment of the profiles, which includes automatic techniques to adapt the profile. The most common technique used is to add new information from information obtained from user feedback. The drawback is that old preferences are not forgotten. However, it can be implemented a *gradual forgetting function*, that allows the recommender system to eliminate old interests from user profiles [43].

In this work we will see the automatic adaptation of user profiles, which currently is the base for the management of user profiles, especially in Artificial Intelligence. The main objective is to minimize the effort required by the user and provide the utmost simplicity. The main idea is to perform automatic adaptation of the user profiles without direct user intervention through the analysis of their actions and behaviors.

3.2 Automatic Learning Techniques of user profile

Automatic learning aims to develop techniques that allow computers to learn. It's about creating programs able to generalize behaviours from unstructured information supplied in the form of examples.

Learning could be defined by the following formula:

Learning = Selection + Adaptation

Other definitions we can find of Machine Learning:

Herbert Simon [44]: any change in a system that allows you to do better next time, on the same task or another taken of the same population.

Rafael Bello [45]: is a subfield of Artificial Intelligence responsible for studying the problem of learning machines, that is, their object of study is the problem of how machines can acquire knowledge that enables them to solve specific problems.

In automatic learning we can distinguish between the following learning strategies [46]:

- Supervised Learning.
- Unsupervised Learning.
- Reinforcement Learning.

In supervised learning we count with a priori knowledge, which establishes a correspondence between inputs and desired outputs of the system. The task is classifying an object into a category or class in which we have already qualified models.

On the other side, unsupervised learning does not have any type of knowledge a priori. This case focuses exclusively on a set of inputs given to the system that should be able to recognize patterns in order to label the new entries.

We can also find semi-supervised methods that combine the two previous learning methods. Their input information comes from the feedback that they get from the outside world as an answer of their actions and the system learns on the basis of trial and error. The fundamental problem is to define a policy to maximize the positive stimulus.

Of the different strategies exposed, for this work we will have unsupervised learning. Initially we will not have any kind of knowledge and the system focuses exclusively on a set of inputs to the system provided by the feedback from the user interaction with the system.

3.3 Selection of relevant information

As named above, the adaptation of the profiles will obtain the relevant information from the alternative selected according to the user's interests. This section is based on the actions and behavior that a user has in function of his preferences. Each user has some preferences and, based on them, it makes an election or other, and these actions determine the user’s behaviour.

For this work we have chosen an unsupervised adaptation method that recreates the behaviour that a user would have. However, it is difficult to simulate the behavior of a user, taking into account the large amounts of information that exist and preferences that a user may have.

To select the best proposal we can distinguish two parts. On the one hand, the evaluation and ranking of the alternatives to the user, and on the other hand, the selection of the preferred alternative of the user through the interaction with the system.

In the first part, the proposal of the alternatives to the user consists in recommending a set of alternatives in order of highest to lowest interest the user, which are evaluated from the existing preferences of the user profile. These alternatives are evaluated individually as we have seen in figure 5 to get a score so you can evaluate them and show them to the user in order of preference as we can see in Figure 5.

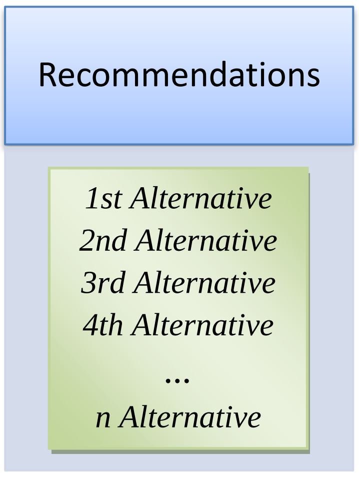


Figure 5: Recommendations alternatives

Once the alternatives have been presented to the user, he will select the desired alternative. Figure 6 exemplifies three consecutive recommendations. Here we can view the ranking of the alternatives and also the selection of each recommendation for the best alternative.

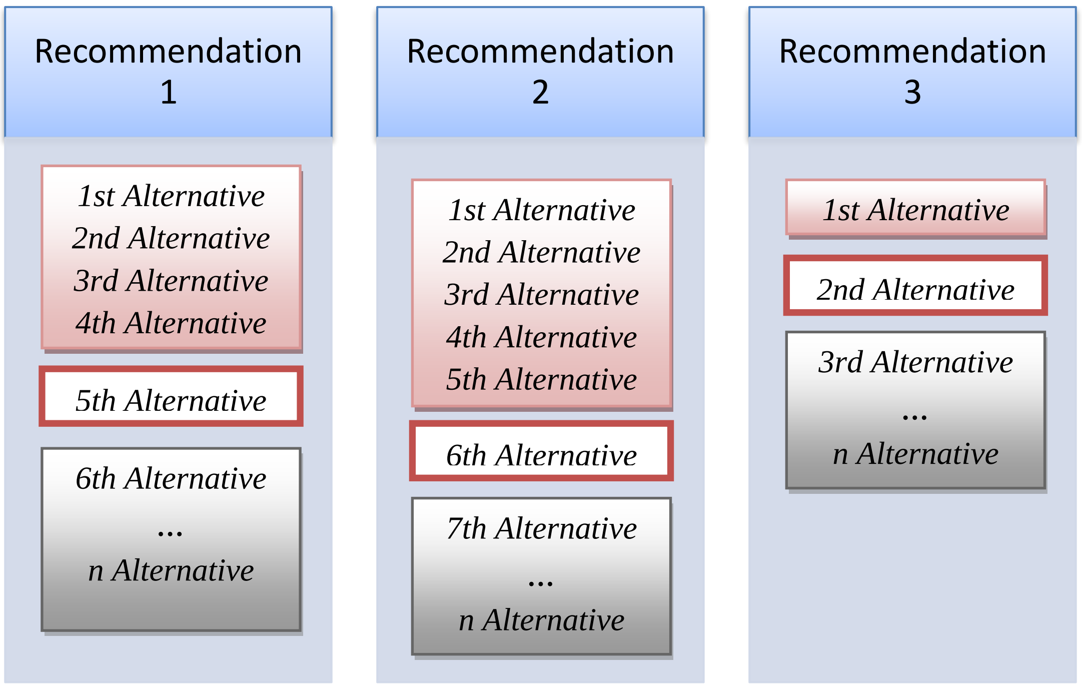


Figure 6: Recommendations alternatives and selection alternative

The selection of an alternative is only the user's choice of a proposal which he considers the best. To simulate this step in an automatic way we will require a profile that represents the real preferences of a user.

To simulate the process of content selection we have defined an ideal profile with a supposed preferences set that matches the preferences of a real user. The ideal profile is only a supposed user profile with all the preferences and values established as seen in Table 3.

In this way, on the one hand we have the evaluation of alternatives according to the current preferences of the user profile, and on the other hand we have the selection of the best alternative according to the preferences established in our fictitious profile representing the user that interacts with the system.

3.4 Method of adaptation of the user profile

Upon completion of the decision-making and after choosing an alternative as seen in the previous section, we proceed to update the user profile preferences. This process consists in adapting a user profile by using information extracted from the user interaction with the system in order to adapt or learn new user preferences to give more precise recommendations. Keep in mind that user preferences can evolve over time and therefore we must be able to adapt our user profile. For example, a user can have a strong interest in subjects related to sports, but due to personal circumstances it may change this interest by other interests such as politics or society.

In this work we will have an adaptation process on-line, in which the system will make his learning during runtime to maintain the current user's profile by evaluating each recommendation and the choice made by the user.

For the adaptation process we will have two sources of information to be evaluated.

- The selected alternative. The relevant information extracted from the selection will correspond with the desired information and, therefore, it will be positively valued inside the user profile adaptation.
- The alternatives that were ranked over the selected one. These alternatives have been overvalued and therefore they have to negatively valued inside the user profile.

The following Figure 7 illustrates these two sources of information.

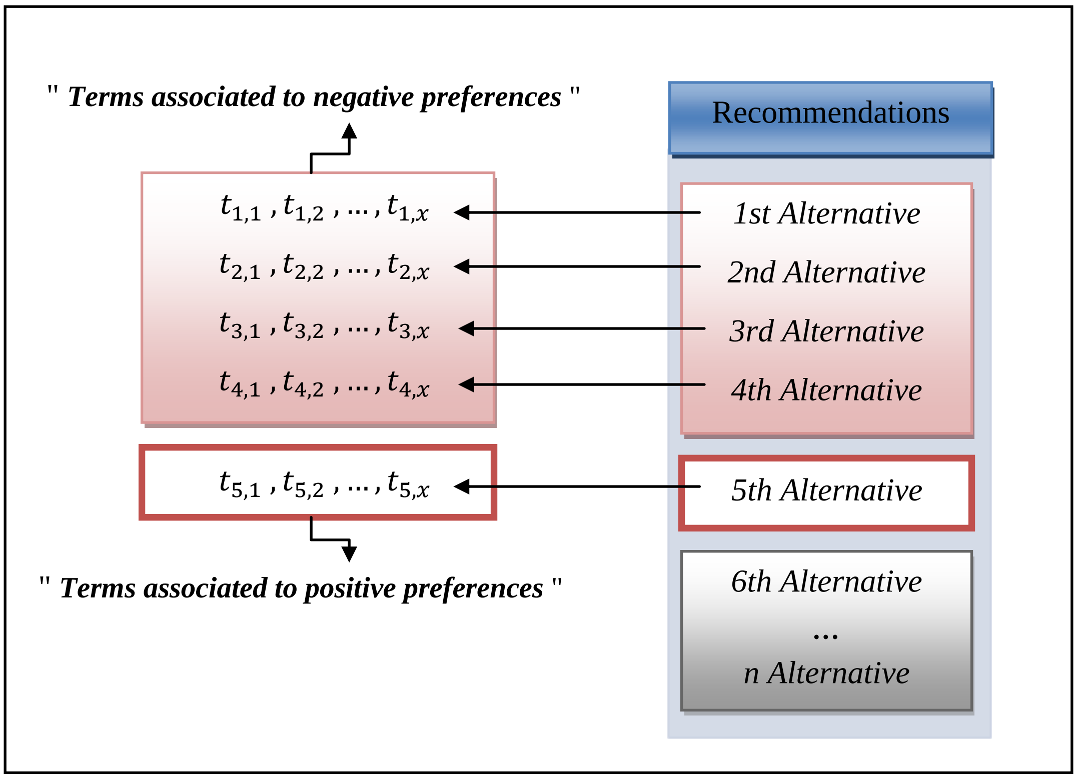
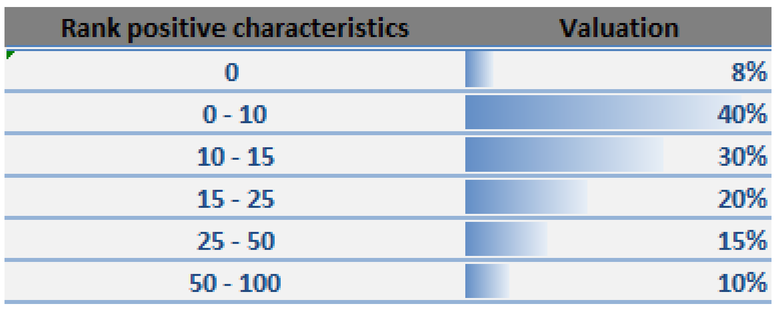

" Terms associated to negative preferences "
Recommendations
$t_{1,1}, t_{1,2}, \dots, t_{1,x}$
$t_{2,1}, t_{2,2}, \dots, t_{2,x}$
$t_{3,1}, t_{3,2}, \dots, t_{3,x}$
$t_{4,1}, t_{4,2}, \dots, t_{4,x}$
$t_{5,1}, t_{5,2}, \dots, t_{5,x}$
1st Alternative
2nd Alternative
3rd Alternative
4th Alternative
5th Alternative
6th Alternative
...
n Alternative
" Terms associated to positive preferences "

Figure 7: Source of information to be evaluated

In following tables we can see the specification of the applicable values for both the increase or decrease in the values of the terms that we will extract from the user feedback with the system. The goal is to adapt the user profile.

Table 4 corresponds to the values assigned to the terms of the selected alternative and which will have a positive valuation for adaptation of the profile. In the next section we can see the study of these values.



Rank positive characteristics	Valuation
0	8%
0 - 10	40%
10 - 15	30%
15 - 25	20%
25 - 50	15%
50 - 100	10%

Table 4: Rank positive characteristics and your valuation

The calculation of the rank is performed as follows:

$$rank = \frac{f}{n}$$

With

- *f:* Frequency of appearance of the word from the selected alternative in the set of over ranked alternatives + selected alternative.
- *n:* Position of the selected alternative - 1.

The aim is to consider both the frequency of occurrence of the term and the position of the selected alternative. In the case of the terms associated with positive preferences, the larger the distance between the selected alternative and the first proposal, the greater will be the valuation of the term. Furthermore, number of occurrences of the term in the over ranked alternatives, affects negatively, decreasing the value of the term.

Table 5 corresponds to the values assigned to the terms that we get from the overranked alternatives. In this case, these values will be negative for the adaptation of the profile.

Rank negative characteristics	Valuation
0 - 10	3%
10 - 20	5%
20 - 40	10%
40 - 60	15%
60 - 100	20%

Table 5: Rank negative characteristics and your valuation

The calculation of the value of the negative terms follows the same procedure used for the positive terms. However, in this case, the larger the distance between the selected alternative and the proposal, the shorter is the valuation of the term. Moreover, the greater the appearance of this term in the over ranked alternatives, the higher the evaluation of the term.

In the case that any term of the over ranked alternatives appears among the terms of the selected option, it will not be taken into account as a negative term.

The purpose of the calculation of these values for the terms is to adapt the profile preferences of the existing terms and, if they do not exist, to add them.

To see this process more clearly we will see a detailed example (see Table 6) of how it would make appropriate adaptations of the preferences of user profile.

Alternatives	Keywords					
1st Alternative	Football	Argentina	Player	Messi	Team	Ball
2nd Alternative	Pepe	Madrid	Defender	Fans	Team	Premier League
3rd Alternative	Phone	market	Android	company	Uk	Technology
4th Alternative	Music	Rock	Album	Band	Year	Concert
5th Alternative	Messi	Barcelona	Team	Player	Goal	Champion League
...	...	...	...	...	...	...
n Alternative	Minister	Politic	David Cameron	UK	News	Government

Table 6: Example of the recommendation alternatives and their keywords

In the above example we see how the selected alternative is the 5th option. Let's take as example one of the terms extracted from the text that provides this alternative, in this case the keyword *Messi*.

In this example we can observe that the term *Messi* also appears on one occasion in the overranked alternatives, therefore the range and valuation of the characteristic would correspond as follows according to Table 4.

$$rank = \frac{2}{4} = 0.5\ (50\ )$$

Valuation of the word *Messi* = 15

Furthermore, in the case of negative adaptations, if we take as an example the word *Pepe* the procedure would be the same. The range and valuation of the characteristic would correspond as follows according to Table 5.

$$rank \ = \frac{1}{4} = 0.25\ (25\ )$$

Valuation of the word *Pepe* = 10

The word *Team*, would be an example of the case discussed above, which will not have a negative valuation, because it is present between the terms of the selected alternative, and it is prioritized as a positive term.

The idea is to adjust preferences as much as possible without affecting notoriously isolated cases and at the same time if there is a sudden change of the preferences of a user they can also be updated both positively and negatively.

After seeing these examples we can conclude that the positive adaptation of the characteristics of the selected alternative will be greater to the extent that these terms do not appear on the over ranked proposals, suggesting that its preference level is crucial and it therefore requires a higher adaptation. Also if the selected alternative corresponds with the best alternative proposal anyway it will have a positive assessment as it corresponds to the case of rank 0.

$$rank \ = \frac{0}{x\epsilon\ N} = 0\ (0\%)$$

Valuation of the word = 8

On the opposite side, with features extracted from the over ranked alternatives, the more repeated they are, the greater their negative impact on the adaptation. It means that the features are over-valued and are of less interest.

In the treatment of this last set of negative characteristics it has been decided not to include those that appear in the set of terms obtained from the selected alternative.

3.5 Conclusion

In this section we have discussed the adaption of the user profile where we have focused in automatic learning and where we have analyzed the different learning strategies. We have seen that among the different approaches our work is based on unsupervised learning.

Furthermore, we have seen the process of proposing alternatives to the user and how to perform the selection of the best alternative according to the user preferences and we have described the method of adjustment of the user profile from the alternative selected by the user.

4 Evaluation

In the previous chapter we introduced the rating and ranking process prior to the implicit adaptation of the user profile. These processes have been implemented in a Java system simulator that eases the analysis of all the parameters according to the domain used (British newspaper The Guardian).

In this section, we will describe the domain of work in which the adaptation algorithm has been tested and the performance of the adaptation algorithm of the user profile has been evaluated. In addition, we will comment the simulation platform that has been designed and implemented.

We will see in detail the testing environment used to evaluate the platform as well as the measures to appraise the adaptation results, consisting in measuring the distance between the profile currently being adapted and the ideal profile.

Finally, the last section of the chapter discusses how to tune the system parameters to improve performance of the adaptation mechanisms.

4.1 Work domain

In order to test and evaluate our recommender method we have analyzed the dynamic learning and refinement of preferences of the user profile through keywords. In this work we have chosen to treat the sources of information from the newspaper The Guardian to perform in a more accurate and efficient way the evaluation of the system. However, it could have been any textual content domain, as well as social networks, other papers, documents, blogs, etc, because this system is intended to adapt a user profile from any type of content of textual information.

Key features of The Guardian:

- Articles published daily and an archive going as far back as 1999. There are more than 1,000,000 articles available.
- A range of media resources including pictures, video, podcasts and the Guardian's editorially curated tag database.
- A database of political information including MPs, constituencies, and election results.
- Free-to-use spreadsheets and data curated by the Guardian news editors, and a search engine for finding open data published by governments around the world.
- A framework for offering content, data, tools and rich user experiences developed by commercial partners directly into sidebar components and full pages on guardian.co.uk.

The newspaper The Guardian has a wide variety of news. We have chosen to deal with specific news so we can make the assessment tests in a more precise and controlled environment. Thanks to an API that provides the open platform of the newspaper we can filter the news in several ways, as well as by content, date, section, etc.

We will deal with news filtered in the following sections:.

SECTIONS
Sport
Football
Technology
Music
Film
Politics
Science
Society
Education
Media

Table 7: Sections of The Guardian considered in this work

4.2 Testing platform

In this work, to make appropriate tests and evaluate our system, we have designed and implemented on Java tossed platform. Its main functionalities are:

- Definition of a recommendation problem (set of user criteria and adaptation parameters).
- Definition of an initial profile.
- Definition of an ideal profile with preferences that we aim to learn.
- Upload of a data file and/or random generation of a corpus of alternatives for the problem.
- Simulation of tests of user interaction with the system.
- Visualization of the evolution of the user profile.

To facilitate loading of data, information management and reduce the computational time of the system, we have developed a local repository in which relevant data are downloaded. For our tests these data correspond to news from The Guardian. Thus, we avoid to have to download data reducing the temporal cost for all the tests that have been carried out.

4.3 Testing algorithm

Figure 8 shows the pseudo code of the algorithm that permits an automatic validation of the adaptation mechanisms by simulating how the system adapts an initial empty profile using the information extracted from the simulated user choices calculated using an ideal profile.

The algorithm requires several parameters such as the corpus of alternatives, some thresholds analysed in detail in the next subsections, and the profiles to run a simulation:

a) Corpus of testing alternatives ($E(a_{0},\ldots,a_{max})$).
b) Positive changes of preference per recommendation (*pVal*).
c) Negative changes of preference per recommendation (*nVal*).
d) Alternatives treated at each iteration (*ai*).
e) Terms treated at each alternative (*ta*).
f) Number of iterations to simulate (*MaxIter*).
g) Ideal profile (*I*).
h) Initial profile to be adapted (*P*).

At each iteration, the first thing to do is obtain the alternatives that will be treated from the whole corpus of alternatives. This subset of alternatives is limited by the parameters *beg* and *end* calculated in lines 14 and 15 of the pseudo code.

Afterwards, this subset of alternatives is rated and ranked using the current profile (*P*) in line 16. Then, the same subset of alternatives is rated and ranked using the ideal profile (*I*) in order to obtain the selected alternative *sel*, which is the alternative which a user with a profile *I* would choose (line 17).

```
1:  ADAPTATION-ALGORITHM
2:   $E(a_0,…,a_{max})$, //corpus of alternatives
7:   pVal, // positive preferences changes
7:   nVal, // negative preferences changes
7:   ta, //terms per alternative
7:   ai, //alternatives per iteration shown
8:   MaxIter, //iterations to simulate
9:   I, //ideal profile
10:  P, //initial profile
11:   begin
12:   iter, beg, end, sel=0;
13:   while (iter<MaxIter)
14:       beg=iter*ai;
15:       end=beg + ai;
16:       rating-and-ranking(P,$E(a_{beg},…,a_{end})$);
17:       sel=calculate-ideal-selection($E(a_{beg},…,a_{end})$, I );
18:        pVal=calculate-positive-values ($E(a_{beg},…, a_{sel})$);
19:        if (sel>0) then
                nVal=calculate-negative-values ($E(a_{beg},…, a_{sel})$);
20:       P=adaptation(P,pVal,nVal);
21:       iter=iter+1;
22:   end while;
23:  end;
```

Figure 8: Pseudo code of the simulator

The positive characteristics are extracted (line 18) and if the *sel* is not the best alternative, then, negative characteristics are also extracted (line 19). Finally, the adaptations are performed over the initial profile (*P*) (line 20) .

In the next section, a study of the execution of this algorithm is made and the steps followed to evaluate a concrete set of alternatives are detailed.

4.3.1 Testing environment

Two sets of tests have been made using a corpus of 6000 alternatives with blocks of 15 alternatives for iteration.

The first corpus of alternatives contains information about news from The Guardians (as the example used before in this document, Figure 3). Each of those alternatives represents news of different sections (Table 7) in which the decision maker needs aid choosing one. The second corpus of alternatives follows the same procedure as the above corpus, but alternatives are only formed by news from a concrete section.

Two types of profiles have been initialized for both tests in order to obtain comparable results. On the one hand, an ideal profile for each corpus of alternatives in which the ideal user preferences are indicated. On the other hand, the same empty initial profile for each test, which the system aims to adapt until it is very similar to the ideal one.

As formalized in the algorithm of Figure 8, each iteration step of the evaluation process with the corpus of alternatives defined above consists in the following points:

1. Obtain a block of 15 alternatives from the corpus of alternatives of the domain.
2. Rank those 15 alternatives using the current profile.
3. Rank the same 15 alternatives using the ideal profile.
4. Store as the user selection the first ranked alternative in step 3.
5. Store as over ranked alternatives the ones ranked above the selection in step 2.
6. Execute the adaptation processes.
7. Calculate and store the distance between the ideal profile and the current profile.

We calculate in each iteration the distance between the current (P) and the ideal (I) profiles, which is defined as follows:

$$dist(I,P) = \left(\sum_{i=1}^{n} \frac{|p_i - s|}{r}\right) * \frac{1}{n}$$

with:

- p_i: Valuation of the term in the ideal profile (I).
- s: Valuation of the term in the current profile (P). If the term does not exist, its valuation is 0.
- n: Number of attributes of ideal profile.
- r: Valuation range of terms: 200 (-100, 100).

The distance is 0 when both profiles are identical. The maximum distance is 1 when all the terms contained in the user and the ideal profiles have just the opposite preference values (-100 and 100). Moreover, this function is commutative ($dist(P,I) = dist(I,P)$).

In our tests, we have an empty initial profile as explained above, in which the initial values are 0.

4.3.2 Performance on-line and off-line adaptation processes

Figure 9 and Figure 10 show the general performance of the adaptation algorithm by means of the profile distance calculated using the equation $dist(I, P)$. This test has been done using the first corpus of alternatives.

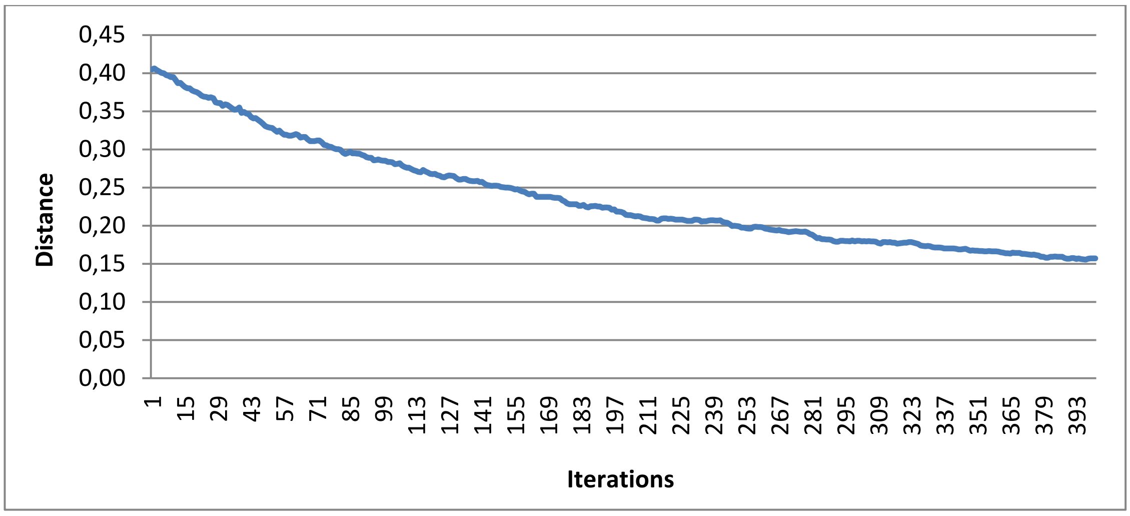


Figure 9: Example of the general performance

Figure 9 shows the distance between the current and the ideal profile through 400 iterations. The parameters used in this simulation are the ones which gave the best results in the parameter's analysis conducted in the following section.

In this test, we can see how the initial distance is gradually reduced at the same time. It can be observed that learning is faster initially.

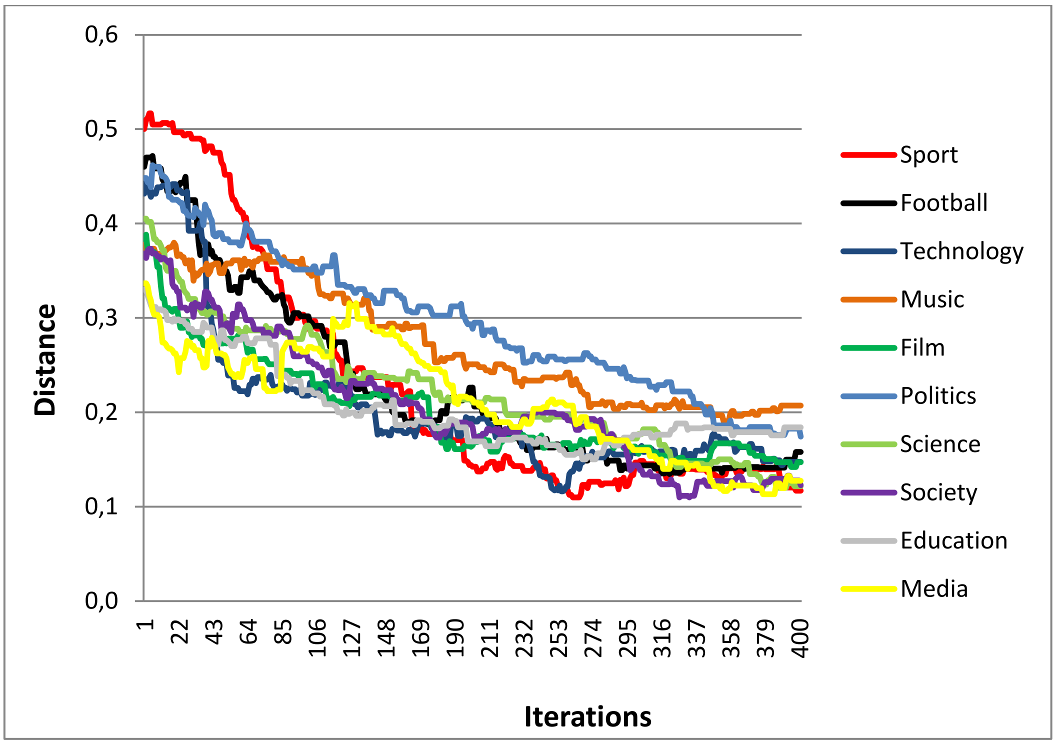


Figure 10: Example for sections of the general performance

Figure 10 shows the distance between the current and the ideal profile through 400 iterations analyzing the preferences on the different sections that make up the ideal profile. The average of these distances is the general performance seen in Figure 9.

In this section we note that the preferences set on the first 5 sections (Sport, Football, Technology, Music and Film) tend to have a positive character ($v_{pref} \in [0, 100]$). Whereas the remaining 5 sections tend to have a negative interest ($v_{pref} \in [-100, 0]$).

We can see more clearly the pace learning. We also see that preferences are adapted differently depending on the section, this is due in part to the wide range of information content that we use. Similarly we see cases where preferences are learned better than others but in general is seen a very good average learning of the user preferences.

The following images show the performance of the preference learning algorithm when considering a concrete section and a corpus of alternatives based on the section to analyze.

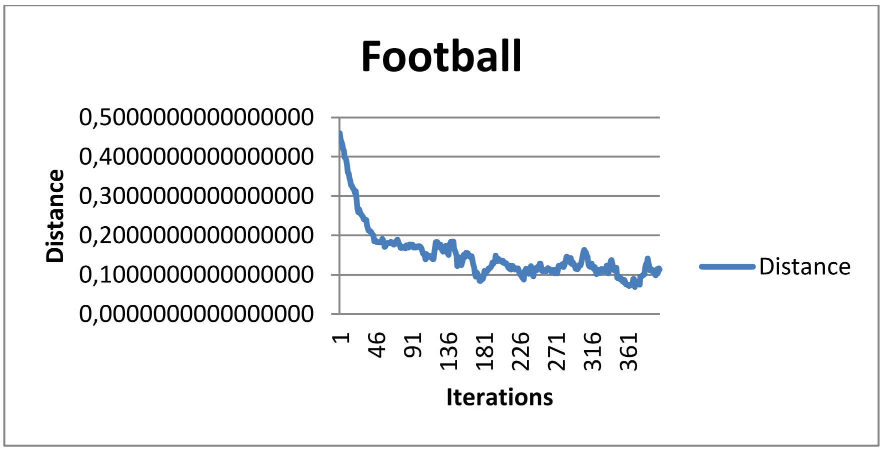


Figure 11: General performance with alternatives of Football

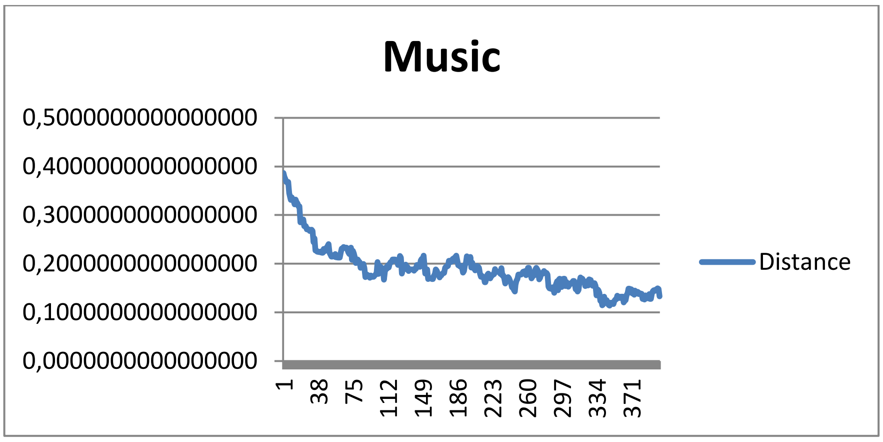


Figure 12: General performance with alternatives of Music

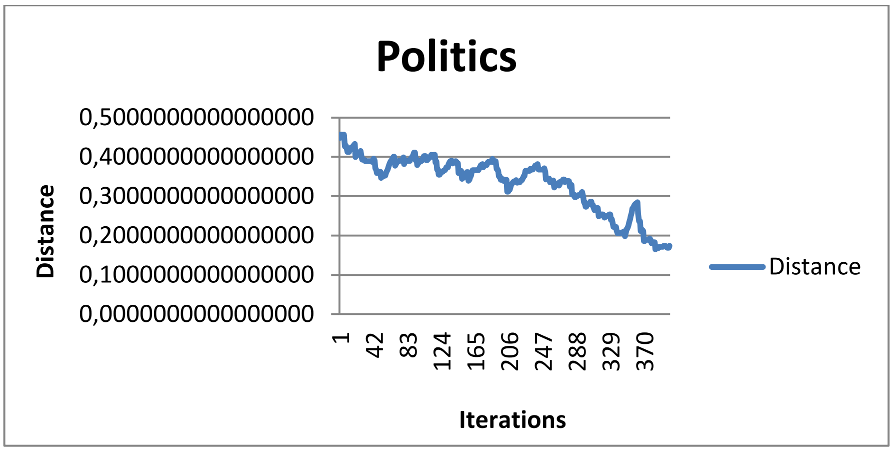


Figure 13: General performance with alternatives of Politics

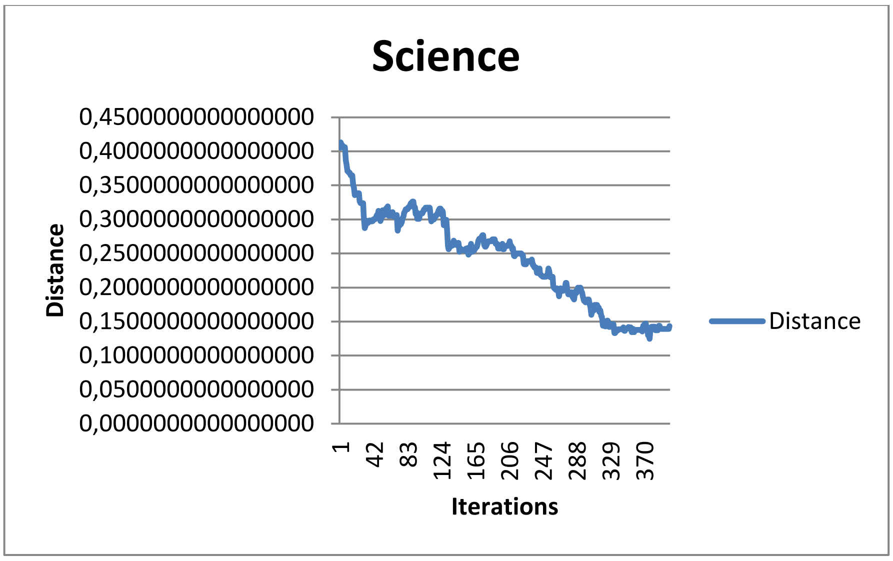


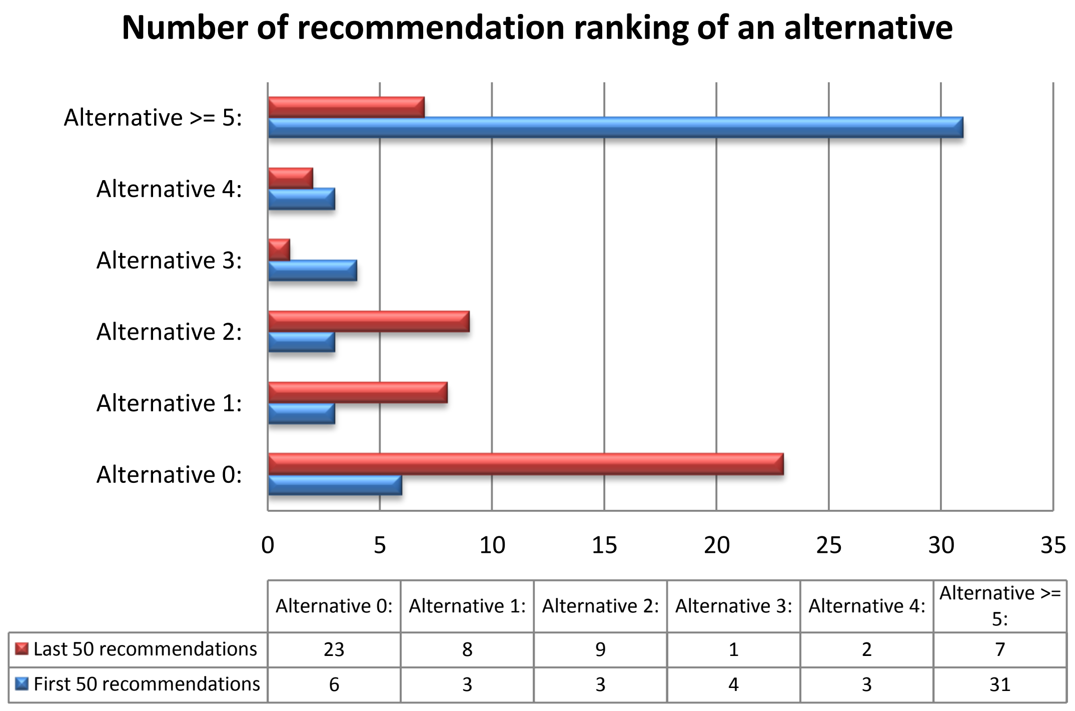
Figure 14: General performance with alternatives of Science

Analyzing the images above, we can see that if we reduce the information content to a single section of interest, learning is done faster and better. Learning is faster because we focus on a set of preferences and an environment more reduced that offers a more optimal result.

4.3.3 Position of the best alternative

The equation to calculate the distance between the current and the ideal profiles gives a general evaluation of the performance of the adaptation algorithm. However, it is also interesting to see the evolution of the alternatives recommended to the user. One of the objectives is to achieve that the alternatives recommended to the user are the most accurate according to the user preferences.

This following test has been done by analyzing the first and last 50 iterations and studying the position of the alternative selected by the user.



	Alternative 0:	Alternative 1:	Alternative 2:	Alternative 3:	Alternative 4:	Alternative >= 5:
🟥 Last 50 recommendations	23	8	9	1	2	7
🟦 First 50 recommendations	6	3	3	4	3	31

Figure 15: Position of the alternative selected by the user

In the first 50 recommended alternatives we can see how in most cases the last alternative is not located in the 5 best positions by the recommendation algorithm, because the current profile has still not sufficient features learned to provide recommendations adjusted to the user preferences.

In the last 50 recommended alternatives we can see how the selection of the best alternatives is much more accurate and better adjusted to user preferences. We can see that the majority of optimal recommendations corresponded to the first 3 ranked alternatives.

4.4 Analysis of the parameters

The proposed implicit adaptation algorithm introduces several parameters that should be properly customised. This section explains the influence of these parameters in the final result. We have conducted these tests to see how the final result is affected by decreasing or increasing the value of said parameters and to find the best value for each parameter.

To evaluate the influence of these parameters we have chosen to analyze the first corpus of 6000 news, always starting from the same corpus of alternative proposals. In this corpus the alternatives are uniformly distributed according to the provenance of information sources in relation to the sections shown in Table 7.

Each parameter value has been tested in the adaptation of the initial profile indicated in the previous section. The distances obtained through those tests is the result shown in the graphics below.

4.4.1 Number of terms for alternative

As seen above, the terms that we can obtain from an information source may vary and be huge amounts. In this step, we analyze how the number of terms that we get from each source of information affects the learning of the preferences of the user profile.

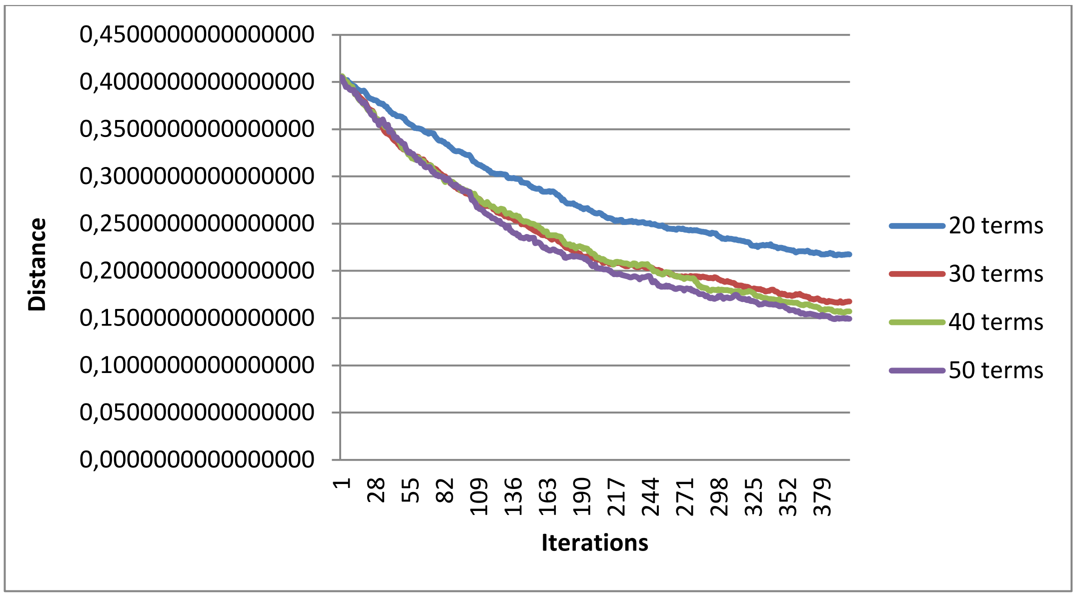


Figure 16: System performance depending on the number of terms/alternative

In Figure 16 we can see how the system has performed better learning when we have more terms. When dealing with more terms there is a larger amount of information and this improves the adaptation of the profile. However, if we increase the amount of terms to treat it, also increases the computation time for all operations.

We have chosen to treat a maximum of 40 terms, as we considered that with this number we obtained sufficient relevant information from each information source. At the same time, this amount of terms provides good results close to those obtained with 50 terms. Note also that 20 terms are insufficient to adapt the profile and there is a big difference with respect to the other values.

4.4.2 Number of alternatives considered in each iteration.

During the interaction with the platform, the set of alternative proposals must be shown to the user for his selection of the best alternative according to his preferences. Figure 17 depicts the influence of the variation of the number of alternatives in the adaptation process.

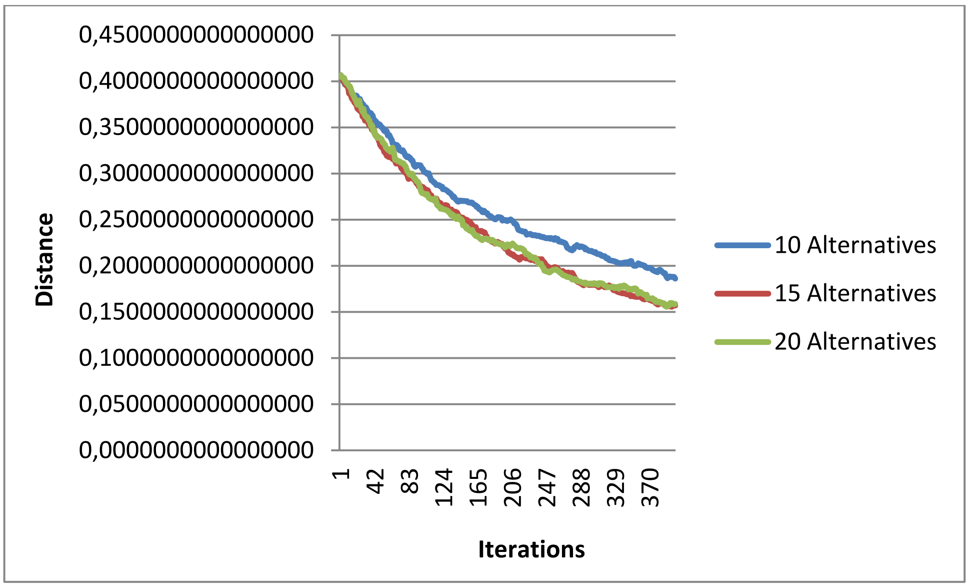


Figure 17: System performance depending on the number of alternatives/iteration

In the following image we can appreciate that the sets of 15 and 20 alternatives are the most optimal. We have chosen a set of 15 alternatives considering that it has sufficient information to facilitate the analysis and decision-making on a set of alternatives.

4.4.3 Threshold considered to learn user preferences

One of the parameters analyzed has been the threshold considered to learn user preferences. A larger threshold imply a greater distance between the first proposal alternative and the selected alternative by the user.

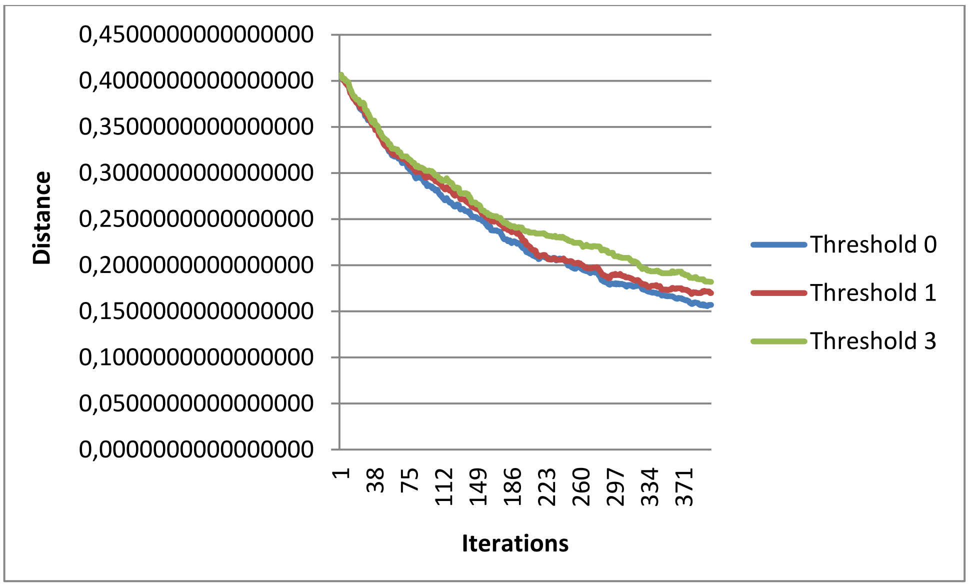


Figure 18: Influence of the extraction threshold

Figure 18 shows that after several tests (using the threshold values of 0, 1 and 3) a threshold of 0 was the one that got better results.

The problem of using a threshold to make the necessary adaptations to the user profile based on an empty initial profile is that when we have some learned preferences it is easy that the first proposals made to the user are the best because of the possibility that other proposals are of less interest, no interest or even negative interest.

Due to the application of a threshold to determine if changes are made to the user profile, the number of possible changes in the profile can limit learning user preferences. We have chosen not to apply this parameter (Threshold value of 0).

4.4.4 Adaptation values of Terms

On the adaptation mechanism section, two parameter for controlling how many increases and decreases of preference can be done at each adaptation process iteration was introduced. Following examples compares the performance of the system with different values for these parameter.

In Figure 10 we can see that learning between stated preferences more positive associated with the first 5 sections and more negatives associated with the remaining sections is quite similar and compact.

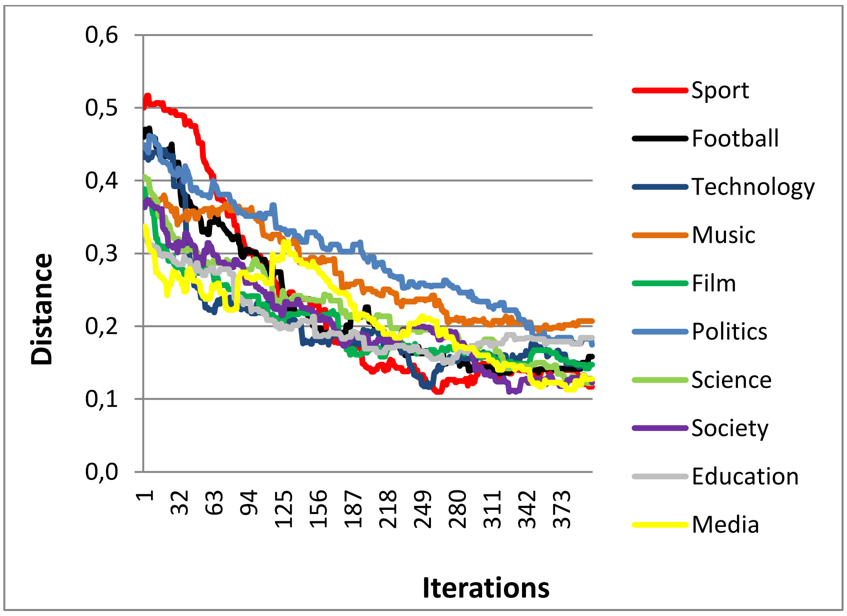


Figure 10: Example for sections of the general performance

The following figures are the result of increasing or decreasing the positive or negative values.

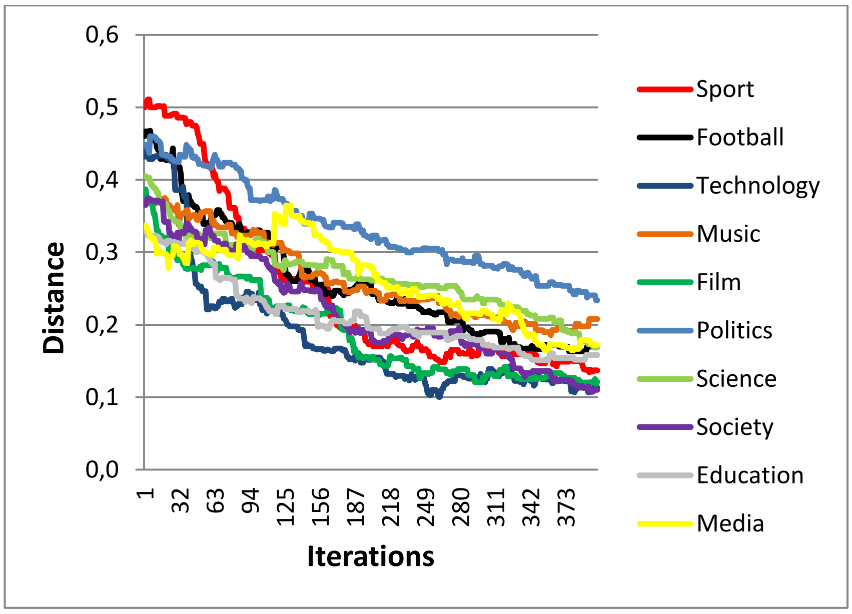


Figure 19: Example 1 for sections of the general performance

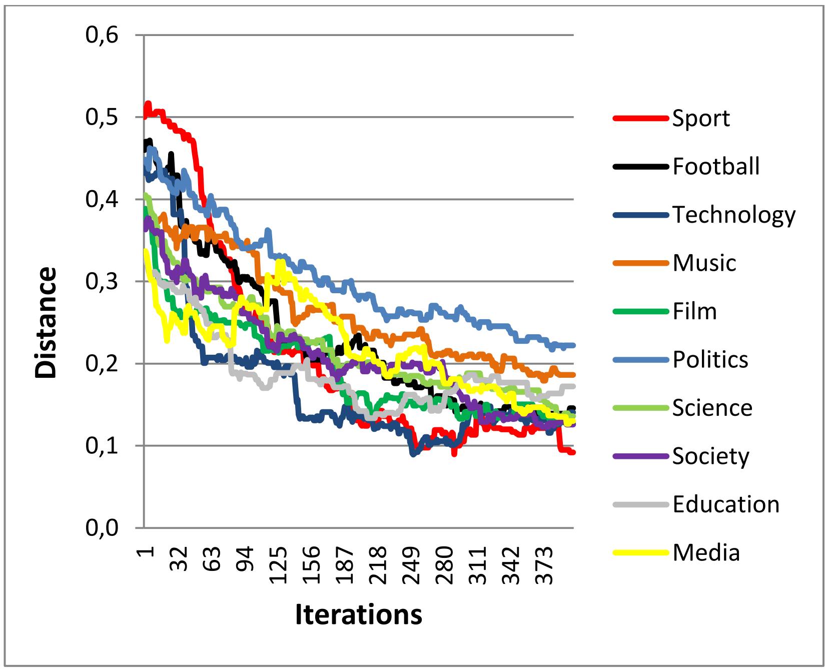


Figure 20: Example 2 for sections of the general performance

The following table shows a comparison, more detailed of the values assumed for each adaptation.

	Figure 10	Example 1	Example 2
Rank positive characteristics	**Valuation**	**Valuation**	**Valuation**
0	8%	8%	10%
0 - 10	40%	40%	42%
10 - 15	30%	30%	35%
15 - 25	20%	20%	25%
25 - 50	15%	15%	20%
50 - 100	10%	10%	15%
Rank negative characteristics	**Valuation**	**Valuation**	**Valuation**
0 - 10	5%	3%	5%
10 - 20	10%	5%	10%
20 - 40	15%	10%	15%
40 - 60	20%	15%	20%
60 - 100	25%	20%	25%

Table 8: Adaptation values of terms

We have analyzed the parameters and their corresponding tests and we can conclude that increasing adaptation values of positive terms can worsen the learning of negative terms. In the reverse case, the increase of the adaptation values of the negative terms can worsen the learning of positive terms. The parameters set are those that offer a better balance between learning positive and negative preferences.

4.4.5 Distribution of the alternatives

As we have seen above, the analysis of the different parameters that influence the adaptation of the user profile has been conducted on the corpus of 6000 alternatives with the same uniform distribution. Now we will see how it affects the adaptation of the user profile if the distribution of alternatives is random. We will not consider from what section the information source comes. In the next figure we take the average of three tests performed.

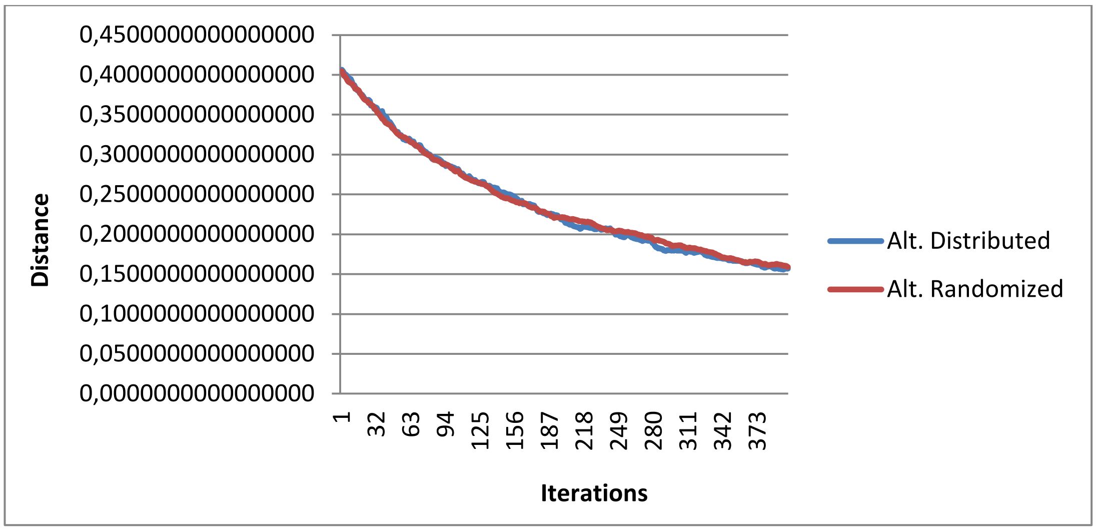


Figure 21: Influence of the distribution of the alternatives

The image shows that there is little difference between the final results. However, in a random distribution it could be the case that the alternatives proposed to user at a given moment all were of interest, or none of them had interest to the user and that forced the selection of a bad proposal according to his preferences.

4.5 Conclusions

In this chapter we have described the domain of work in which the adaptation algorithm has been tested and we have seen the main functionalities on the Java tossed platform that we have designed and implemented.

In addition, we have shown and described the pseudo code of the algorithm that permits an automatic adaptation of the user profile.

Finally, we have conducted a study on the parameters which tune the adaptation process, discussing how they affect the result and how they should be set up to obtain a faster and a more accurate adaptation process.

After conducting the study for the several parameters, we can conclude that the parameters obtained in our system should function and behave in the same way in any environment that have textual content.

5 Concluding remarks

In this Master Thesis, we have designed and implemented a recommender system, which adapts the user profile in an unsupervised way based on the textual content of the large amounts of information which currently exist. For this purpose, the objectives that were indicated at the beginning of the work have been completely accomplished, differentiating the two main parts of the work: *algorithms* and *techniques to rank a set of alternatives*, and the *techniques to maintain the profile updated*.

For this first part of the work, the following goals were formulated, all of which have been completely accomplished.

- A previous study of the information that we will handle for a better statement and knowledge of user preferences.
- Creation and design of an ideal profile to represent user preferences.
- Creation and design of a current profile to learn user preferences.
- To evaluate objects of type text, we study a technique to obtain the best content of these objects, which has dealt with a set of terms. Those have been reduced to a certain amount with the application of the method of weight " Tf-idf ".
- That system evaluates a set of alternatives and ranks them according to a user profile in which his interests and preferences are declared in the ideal profile.

The second part of the work consisted in designing techniques to automatically adapt the profile that is being used in the recommendation process to generate more accurate recommendations. Those techniques extract implicit information from the user interaction with the recommender system in order to increase or decrease some preference values of the profile.

Finally, the whole system has been tested and evaluated in a concrete environment, the British newspaper *The Guardian*. The different tests have been conducted mainly in two corpus of 6000 alternatives, the first corpus based on different types of information and the second corpus based on alternatives always of the same type of information. The following conclusions can be extracted from this work.

It is possible to learn user preferences in situations where the objects to be treated are formed only by textual information and we continuously have information of selections made by the user. The tests and evaluations conducted show that this system will allow adapting a user profile in an unsupervised and dynamic way through the most relevant terms obtained from textual objects, being able to offer appropriate alternatives to the user according to his preferences.

We can appreciate that learning user preferences depends on the contents of textual objects. A more general environment where we treat any information requires a longer process to learn the preferences, in return, in a more specific environment the process is faster. This is normal if we consider that as the environment in which we operate grows, we handle a wider and often more heterogeneous range of textual content.

Another factor involved in learning user preferences are global terms. Take the example of "football" term, which would positively influence our preferences if accompanying a club of our interest or otherwise negatively affect if accompanying a club in which we are not interested.

As recommendations for future research work, some of the following areas can be of considerable interest.

- The extraction of the main terms of the textual objects that we handle is a key element in this work. Within this process, apart from the application of methods of weights, as we used (TF-IDF) for assigning weights to obtain the most relevant terms, we could apply other techniques, as well as *Stop Words* or apply some filter to eliminate common terms.

- In our profile, the preferences are updated and if they do not exist then they are added. We may implement and study some method to handle the large number of preferences that we may have and proceed to the elimination of some of these, for example after a certain time without influencing user decision.

- Other important future research line is to study the construction of taxonomies to classify and to evaluate better the terms at different levels (a general category could be Football, which could be classified in Teams, which in time could contain Players).

6 References

[1] Gregor, S. (2006). "The Nature of Theory in Information Systems." MISQ 30(3): 611-642.

[2] Jessup, Leonard M.; Joseph S. Valacich (2008). Information Systems Today (3rd ed.). Pearson Publishing. Glossary p. 416.

[3] Lewis, P. (1991). The Decision Making Basis for Information Systems: The Contribution of Vickers' Concept of Appreciation to a Soft Systems Perspective, Journal of Information Systems, 1 (1), 33-43.

[4] Francesco Ricci, Lior Rokach, Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011, pp. 1-35.

[5] Prem Melville and Vikas Sindhwani, Recommender Systems, Encyclopedia of Machine Learning, 2010.

[6] R. K. Brouwer, "Clustering feature vectors with mixed numerical and categorical attributes," *International Journal of Computational Intelligence Systems,* vol. 4, pp. 285-298, 2008.

[7] J. Figueira, S. Greco, and M. Ehrgott, Multiple Criteria Decision Analysis: State of the Art Surveys: Springer, 2005.

[8] L.Marin, D.Isern, A.Valls, A.Moreno; Web-based recommender using linguistic preferences, In: M.F. Norese (Ed.) Proc. 71th Meeting Euro Working Group 'Multiple Criteria Decision Aiding', Torino, Italy, 2010, pp. 50-63.

[9] [9] White, R., Jose, J., and Ruthven, I. Comparing explicit and implicit feedback techniques for web retrieval: Trec-10 interactive track report. NIST SPECIAL PUBLICATION SP, (2002), 534– 538.

[10] O. Noppens, M. Luther, T. Liebig, M. Wagner, and M. Paolucci, 'Ontology-supported Preference Handling for Mobile Music Selection', in Proceedings of the Multidisciplinary Workshop on Advances in Preference Handling in conjunction with ECAI 2006, pp. 96–101, Riva del Garda, Italy, (2006).

[11] S.-F. Pan and J.-H. Lee, ‘eDAADe: An Adaptive Recommendation System for Comparison and Analysis of Architectural Precedents’, in Proceedings of the 4th International Conference of Adaptive Hypermedia and Adaptive Web-Based Systems, AH 2006, eds., V. Wade, H. Ashman, and B. Smyth, volume 4018 of Lecture Notes in Computer Science, pp. 370–373, Dublin, Ireland, (2006). Springer Berlin/Heidelberg.

[12] Jawaheer, G., Szomszor, M., and Kostkova, P.Characterisation of explicit feedback in an online music recommendation service. ACM Recommender Systems Conference 2010, Barcelona (in press), (2010).

[13] Morita, M. and Shinoda, Y. 1994. Information Filtering Based on User Behavior Analysis and Best Match Text Retrieval. In Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 272-281.

[14] L. Baltrunas and X. Amatriain, ‘Towards Time-Dependant Recommendation based on Implicit Feedback’, in *Proceedings of the Workshop Context-aware Recommender Systems in conjunction with the 3rd ACM Conference on Recommender Systems*, RecSys 09, pp. 25–30, New York, USA, (2009).

[15] D. M. Nichols, ‘Implicit Rating and Filtering’, *in Proceedings of the 5th* DELOS *Workshop of Filtering and Collaborative Filtering*, pp. 31–36, Budapest, Hongary, (1997). European Research Consortium for Informatics and Mathematics.

[16] X. Guo, G. Zhang, E. Chew, and S. Burdon. A hybrid recommendation approach for one-and-only items. AI 2005: Advances in Artificial Intelligence, pages 457-466, 2005.

[17] R. Burke, Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4):331-370, 2002.

[18] Structure, Models and Meaning: Is "unstructured" data merely unmodeled?, Intelligent Enterprise, March 1, 2005.

[19] Unstructured Data and the 80 Percent Rule, Clarabridge Bridgepoints, 2008 Q3.

[20] Structured Information, Navigation, Access, and Control. (presented at the Berkeley Finding Aid Conference, April 4-6, 1995). by . *Steven J. DeRose, Sr. System architect and* Electronic Book Technologies.

[21] Difference Between. *Structured* and. Unstructured Documents. By: *Randy Van Ittersum and Erin Spalding CDIA+* www.disusa.com. Copyright 2005 www.disusa.

[22] Cucerzan, S. and Agichtein, E. 2005 *Factoid question answering over unstructured and structured content on the web.*

[23] Marie-Francine Moens, E. 2005 *Combining structured and unstructured information in a retrieval model for avvessing lesgislation.*

[24] Cowie, J., & Lehnert, W. (1996). Information extraction. *Communications of the ACM , 1* (39), 80-91.

[25] Grishman, R. (1997). Information Extraction: Techniques and Challenges. In Lecture Notes in Computer Science, 1299:10–27. Springer-Verlag.

[26] Moens, M.-F. (2006). Information Extraction: Algorithms and Prospects in a Retrieval Context. Springer.

[27] S. Sarawagi, "Information extraction," to appear in Foundations and Trends in Information Retrieval, 2009.

[28] OpenNLP Documentation, http://opennlp.sourceforge.net/README.html [Visited 20 Julio 2012].

[29] J. Ramos. Using tf-idf to determine word relevance in document queries. In First International Conference on Machine Learning, New Brunswick: NJ, USA, 2003.

[30] Wu HC, Luk RWP, Wong KF, Kwok KL (2008). "Interpreting tf–idf term weights as making relevance decisions". *ACM Transactions on Information Systems* 26 (3): 1–37.

[31] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," in *Document retrieval systems*, ed: Taylor Graham Publishing, 1988, pp. 132-142.

[32] M. Minio and C. Tasso, "User Modeling for Information Filtering on Internet Services: Exploiting an Extended Version of the UMT Shell," UM96 Workshop on User Modeling for Information Filtering on the WWW, Kailua-Kowa, Hawaii, USA, 1996.

[33] M. Montaner, "Collaborative Recommender Agents Based on Case-Based Reasoning and Trust," PhD Thesis, Universitat de Girona, 2003.

[34] L. Chen and K. Sycara. *WebMate: A Personal Agent for Browsing and Searching*. In Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems, AGENTS ’98, pages 132–139. ACM, 1998.

[35] H. Sorensen and M. McElligot. *PSUN: A Profiling System for Usenet News*. In CKIM ’95Workshop on Intelligent Information Agents, 1995.

[36] G. Potter and R. Trueblood. *Traditional, Semantic, and Hyper-Semantic Approaches to Data Modeling*. In IEEE Computer, volume 21:6, pages 53–63, 1988.

[37] A. Riordan and H. Sorensen. *An intelligent agent for highprecision information filtering*. In Proceedings of the CIKM-95 Conference, 1995.

[38] E. Rich. *User Modeling via Stereotypes*. In Cognitive Science, volume 3, pages 329–354, 1979.

[39] S.N. Pant, K.E. Holbert. “Fuzzy Logic in Decision Making and Signal Processing”. Powerzone, Arizona State University, Abril 2004.

[40] M. Öztürk, A. Tsoukias, and P. Vincke, "Preference Modelling," in *Preferences: Specification, Inference, Applications,* ed. Dagstuhl, Germany: Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2006. Page 3.

[41] M. Montaner, B. López, and J. L. d. l. Rosa, "A Taxonomy of Recommender Agents on the Internet," *Artificial Intelligence Review,* vol. 19, pp. 285-330, 2003.

[42] T. W. Yan and H. Garcia-Molina, "SIFT: a tool for wide-area information dissemination," Proceedings of the USENIX 1995 Technical Conference Proceedings on USENIX 1995 Technical Conference Proceedings, New Orleans, Louisiana, 1995.

[43] G. I. Webb and M. Kuzmycz, "Feature Based Modelling: A methodology for producing coherent, consistent, dynamically changing models of agents' competencies," *User Modeling and User-Adapted Interaction,* vol. 5, pp. 117-150, 1995.

[44] H. A. Simon. The Sciences of Artificial. Cambridge, MA: MIT 1969.

[45] Bello RE, et al. 2002. Aplicaciones de la Inteligencia Artificial.

[46] Mitchell, T. (1997). Machine Learning, McGraw Hill.

[47] Alpaydin, E. 2004 Introduction to Machine Learning (Adaptive Computation and Machine Learning). The MIT Press.

[48] J.R. Cheng and A.R. Hurson, "Effective Clustering of Complex Objects in Object-Oriented Databases," Proc. ACM SIGMOD, pp. 22-31,Denver, May 1991.

[49] Athman Bouguettaya "On Line Clustering", IEEE Transaction on Knowledge and Data Engineering Vol 8, No. 2, Apr 1996.

[50] A. K. Jain, "Data Clustering: 50 Years Beyond K-Means" , Pattern Recognition Letters, Vol. 31, No. 8.

[51] Golfarelli, M., Rizzi, S. Data Warehouse Design: Modern Principles and Methodologies. McGraw-Hill, 2009.

[52] Kenneth D.Bailey, Typologies and taxonomies, An Introduction to Classification Techniques, Vol 102.