

A Relevance Feedback Method for Discovering User Profiles from Text

M. Degemmis, P. Lops, S. Ferilli, N. Di Mauro, T.M.A. Basile, and G. Semeraro

Dipartimento di Informatica
Università di Bari
Via E. Orabona, 4 - 70125 Bari - Italia
{degemmis,lops,ferilli,nicodimauro,basile,semeraro}@di.uniba.it

Abstract. The huge amounts of data on the Internet often make difficult the user's search for relevant information. For this reason, systems that are able to support users in this task could be a valuable help in this activity. Unfortunately, being able to catch user interests and represent them in a structured form is in general a problematic activity. Our research deals with the application of supervised Machine Learning methods for user profiling in the e-commerce area. The paper presents a new method, based on the classical Rocchio algorithm for text categorization, able to discover user preferences from the analysis of textual descriptions of items in online catalogues of e-commerce Web sites. Experiments have been carried out on a dataset of real users, and results have been compared with those obtained using an Inductive Logic Programming (ILP) approach and a probabilistic one.

1 Introduction

E-commerce sites often recommend products they believe a customer is interested in buying. Many web sites have started to embody recommender systems as a way of personalizing their content for users. Recommendation algorithms use input about a customer's interests to generate a list of recommended items. Users are swamped with information and have difficulty in separating relevant from irrelevant information. This *relevant information problem* leads to a clear demand for automated methods able to support users in searching large Web repositories in order to retrieve relevant information with respect to users' individual preferences. A possible way to achieve personalization is to use static profiles that must be manually updated by users when their interests change. These limitations clearly call for alternative methods that infer preference information implicitly and support automated content recommendation. Machine Learning techniques are being used to recognize regularities in the behavior of customers and to infer a model of their interests, referred to as user profile.

The paper presents a new method, based on the classical Rocchio algorithm for text categorization [1], able to discover user preferences from the analysis of textual descriptions of items in the catalogue of an e-commerce Web site.

The novelty of the method can be summarized as follows:

a) positive and negative examples are weighted differently for each user, according to the rates given during the training phase. The classical Rocchio method uses specific control parameters that allow setting the relative importance of *all* positive and negative examples;

b) the method is able to manage documents structured in different slots, each corresponding to a specific feature of an item, for example title, authors, abstract. This strategy permits to give a different weight to words on the basis of the slot in which they appear (according to the idea that words appearing in the title maybe more indicative of user preferences than words appearing in the body of the document).

In order to evaluate the effectiveness of the proposed approach, a comparison with different learning strategies has been carried out, namely an ILP approach and a naïve bayes method. Motivation behind our research is the realization that user profiling can be used to tackle the *relevant information problem* already described. Our experiments evaluated the effects of the above mentioned methods in learning intelligible profiles of users' interests. The experiments were conducted in the context of a content-based profiling system for virtual bookshop on the World Wide Web. In this scenario, a client side utility has been developed in order to download documents (book descriptions) for a user from the Web and to capture users feedback regarding his liking/disliking on the downloaded documents. Then this knowledge can be exploited by the three different learning techniques so that when a trained system encounters a new document it can intelligently infer whether this new document will be liked by the user or not.

The paper is organized as follows: Section 2 describes the main principles for learning user profiles from textual description, and gives details about the way for representing documents. Section 3 describes in more details the new Rocchio-based algorithm for inferring user profiles and gives an overview of the systems used for comparison: INTHELEX and Item Recommender. Section 4 presents the detailed description of the experiments. Finally, some conclusions are drawn in Section 5.

2 Learning User Profiles from Textual Descriptions

Recent research on intelligent information access and recommender systems has focused on the content-based information recommendation paradigm that exploits textual descriptions of the items to be recommended and relevance rates given by users to infer a profile used to recommend items of interest [2]. Text categorization aims to automatically assign categories (classes) to unseen text documents. The task is commonly described as follows: given a set of classes $C = \{c_1, \dots, c_n\}$ and a set of training documents labelled with the class the document belongs to, the problem consists in building a classifier able to assign to a new document the proper class. We consider the problem of learning user profiles as a binary classification task: each document has to be classified as interesting or not with respect to the user preferences. The set of classes is restricted to c_+ , that represents the positive class (user-likes), and c_- the neg-

ative one (user-dislikes). The application of text categorization methods to the problem of learning user profiles is not new (see LIBRA system [3] and Syskill & Webert [4]). The experiments carried out by the authors have shown that the naïve Bayesian classifier offers several advantages over other learning algorithms (also in terms of accuracy and efficiency). Therefore, we have decided to compare the proposed Rocchio-based algorithm with the naïve Bayesian classifier implemented in our Item Recommender system. Moreover, our research aims at comparing these techniques with a symbolic approach able to induce profiles that are more readable from a human understandability viewpoint. The representation of a problem has a strong impact on the generalization accuracy of a learning system. For the problem of inducing a profile of documents liked by a user, documents have to be transformed in a form that is suitable for the learning algorithm.

2.1 Documents Representation

The representation that dominates the text classification literature is known as *bag of words* (BOW). In this approach each feature corresponds to a single word found in the training set. Usually a list of *stop words* (that are assumed to have no information content) is removed from the original text. In order to make the features statistically independent, typically a *stemming* algorithm is used to remove suffixes from words. In our application scenario, item to be recommended are books. Each book is represented by a set of *slots*, where each slot is a textual field corresponding to a specific feature of the book: title, author and textual annotation, that is the abstract of the book. The text in each slot is represented using the *BOW* model taking into account the occurrences of words in the original text. Thus, each instance is represented by three BOWs, one for each slot. This strategy considers separately the occurrences of a word in the slots in which it appears. The idea behind this approach is that by considering the number of occurrences separately in each slot could supply a more effective way to catch the discriminatory power of a word in a document. For example if the word *java* appears only once in the document, but in the slot title, probably it is more indicative of user preferences than if the same word appears in the body of the document. Stemming and stop words removal have been applied to the documents used in the experiments.

3 Learning Profiles Using the modified Rocchio method

Some linear classifiers consist of an explicit profile (or prototypical document) of the category [5]. The Rocchio algorithm is one of the most popular learning methods from Information Retrieval. It has also been widely used for document classification. In this algorithm, documents are represented with the vector space representation and the major heuristic component is the TFIDF (Term Frequency/Inverse Document Frequency) word weighting scheme. The computation of the weights for terms reflects empirical observations regarding text.

Terms that appear frequently in one document (term frequency), but rarely on the outside (inverse document frequency), are more likely to be relevant to the topic of the document. Following the definition of TF and IDF. Let N be the total number of documents in the training set and n_i be the number of documents in which the term t_i appears. Let $freq_{i,j}$ be the raw frequency of term t_i in the document d_j (i.e. the number of times the term is mentioned in the text of the document). The normalized frequency $f_{i,j}$ of term t_i in document d_j is given by

$$f_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}} \quad (1)$$

where the maximum is computed over all terms which are mentioned in the text of document d_j . If the term t_i does not appear in the document d_j then $freq_{i,j} = 0$. Further, let idf_i , inverse document frequency for t_i , be given by

$$idf_i = \log \frac{N}{n_i} \quad (2)$$

The best known term-weighting schemes, called TFIDF, use weights given by the product of term frequency and the inverse document frequency. Learning is achieved by combining document vectors (of positive and negative examples) into a prototype vector \vec{c} for each class in the set of classes C . Items within a certain distance from the prototype (for example determined by the cosine similarity measure) are considered interesting. More formally, Rocchio's method computes a classifier $\vec{c}_i = \langle \omega_{1i}, \dots, \omega_{|\mathcal{T}|i} \rangle$ for category c_i by means of the formula:

$$\omega_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}}{|POS_i|} - \gamma \cdot \sum_{\{d_j \in NEG_i\}} \frac{\omega_{kj}}{|NEG_i|} \quad (3)$$

where ω_{kj} is the tfidf weight of the term t_k in document d_j , POS_i and NEG_i are respectively the set of positive and negative examples in the training set for the specific class. β and γ are control parameters that allow setting the relative importance of all positive and negative examples. The vector model gives the possibility to evaluate the degree of similarity between two vectors using the concept of correlation. This correlation can be quantified, for instance, by the *cosine of the angle* between these two vectors. In order to assign a class \tilde{c} to a document d_j , the similarity between each prototype vector \vec{c}_i and the document vector \vec{d}_j is computed and \tilde{c} will be the c_i with the highest value of similarity. We propose a modified version of this method, that is able to manage documents represented using different slots. As described in the Section 2.1, a document is represented by a set of three slots: title, authors and annotation. More formally, if m is the index of the slot ($m = 1, 2, 3$), a book is represented by the concatenation of the three bag of words:

$$d_j = \langle w_{1j}^m, \dots, w_{|T_m|j}^m \rangle$$

where $|T_m|$ is the cardinality of the vocabulary for the slot s_m and w_{ij}^m is the weight of the term t_i in the document d_j , in the slot s_m . Each weight w_{ij}^m is

computed as follows:

$$w_{ij}^m = f_{ij}^m \cdot idf_i^m \quad (4)$$

where f_{ij}^m is the frequency of term t_i in the document d_j in the slot s_m computed according to the equation (1) and idf_i^m is the inverse document frequency of the term t_i in the slot s_m , computed according to the equation (2) as the logarithm of the ratio between the total number of documents N and the number of documents containing the term t_i in the slot s_m .

In the e-commerce area, items are often grouped in a fixed number of categories. Our goal is to learn a profile of items preferred by a user in a specific category. Thus, given a user u and a set of rated books in a specific category of interest (for example *Computing & Internet*), the goal is to learn a profile able to recognize books liked by the user in that category. The learning process consists in inducing a prototype vector for *each slot*: these three vectors will represent the user profile. The rationale of having distinct components of the profile is that words appearing in a "heavy" slot such as the title could be more indicative of preferences than words appearing in other slots such as the annotation, having a low discriminatory power. For these reasons, each prototype vector of the profile could contribute in a different way to the calculation of the similarity between the vectors representing a book and the vectors representing the user profile. Our modified version of the Rocchio algorithm learns a profile $\vec{p}_i = \langle \omega_{1i}^m, \dots, \omega_{|T|i}^m \rangle$, for a user u and a category c_i by taking into account the rates given by the user on documents in that category during the training phase. The rate $r_{u,j}$ on the document d_j is a discrete judgement ranging from 1 to 10. More formally:

$$\omega_{ki}^m = \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}^m \cdot r'_{u,j}}{|POS_i|} - \sum_{\{d_j \in NEG_i\}} \frac{\omega_{kj}^m \cdot r'_{u,j}}{|NEG_i|} \quad (5)$$

where $r'_{u,j}$ is the normalized value of $r_{u,j}$ ranging between -1 and 1 (respectively corresponding to $r_{u,j} = 1$ and 10), $POS_i = \{d_j \in T_r | r_{u,j} > 5\}$, $NEG_i = \{d_j \in T_r | r_{u,j} \leq 5\}$, and ω_{kj}^m is the weight of the term t_k in the document t_j in the slot s_m computed as in equation (4) where the idf_k^m factor is computed over POS_i or NEG_i depending on the fact that the term t_k is in the slot s_m of a book rated as positive, or negative (if the term is present in both positive and negative books two different values for idf_k^m will be computed). Computing two different idf values for a term, led us to consider the rarity of a term in positive and negative books, in an attempt to catch the discriminatory power of a term in recognizing interesting books. Equation (5) differs from the classical Rocchio formula reported in equation (3) in the fact that the parameters β and γ are substituted by the ratings $r'_{u,j}$ that allow to give a different weight to each document in the training set. As regards the computation of the similarity between a profile \vec{p}_i and a book \vec{d}_j , the idea is to compute three partial similarity values between each couple of corresponding vectors in \vec{p}_i and \vec{d}_j . Then, a weighted average of the three values is computed, by assigning to the similarity for the slots *title* and *authors* an heavier weight than the one assigned to the *annotation*.

$$\text{sim}(\vec{d}_j, \vec{p}_j) = \sum_{K=1}^3 \text{sim}(\vec{d}_j^K, \vec{p}_j^K) \cdot \alpha_k \quad (6)$$

where α_k reflects the importance of a slot in classifying a book. In our experiments we used $\alpha_1 = 0.5$ (title), $\alpha_2 = 0.4$ (authors) and $\alpha_3 = 0.1$ (annotation). If $\text{sim}(\vec{d}_j, \vec{p}_j) > 0$ then d_j is considered as interesting.

3.1 INTHELEX

INTHELEX (INcremental THEory Learner from EXamples) is a learning system for the induction of hierarchical theories from positive and negative examples. It is fully and inherently incremental: this means that, in addition to the possibility of taking as input a previously generated version of the theory, learning can also start from an empty theory and from the first available example. INTHELEX can learn simultaneously various concepts, possibly related to each other, expressed as (sets of) function free clauses to be interpreted according to the Object Identity assumption [6]. Examples describe the observations by means of only basic non-negated predicates of the representation language, and specifies all the classes for which the observed object is a positive example and all those for which it is a negative one. It is important to note that a positive example for a concept is not considered as a negative example for all the other concepts (unless it is explicitly stated). INTHELEX incorporates two inductive operators, one for generalizing definitions that reject positive examples, and the other for specializing definitions that explain negative examples. Both these operators, when applied, change the set of examples the theory accounts for. In particular, when a positive example is not covered, completeness is restored in one of the following ways:

- replacing a clause in the theory with one of its generalizations against the problematic example;
- adding a new clause to the theory, obtained by properly turning constants into variables in the problematic example;
- adding the problematic example as a positive exception.

When, on the other hand, a negative example is covered, consistency is restored by performing one of the following actions:

- adding positive literals that are able to characterize all the past positive examples of the concept (and exclude the problematic one) to one of the clauses that concur to the example coverage;
- adding a negative literal that is able to discriminate the problematic example from all the past positive ones to the clause in the theory by which the problematic example is covered;
- adding the problematic example as a negative exception.

We were led by a twofold motivation to exploit INTHELEX on the problem of learning user profiles. First, its representation language (First-Order Logic) is more intuitive and human readable than values exploited and provided by numeric/probabilistic approaches. Second, incrementality is fundamental in the given task, since new information on a user is available each time he issues a query, and it would be desirable to be able to refine the previously generated profile instead of learning a new one from scratch. Moreover, a user’s interests might change in time, a problem that only incremental systems are able to tackle.

Each book description is represented in terms of three components by using predicates `slot_title(b,t)`, `slot_author(b,au)`, and `slot_annotation(b,an)`, indicating that the objects `t`, `au` and `an` are, respectively, the title, author and annotation of the book `b`. Any word in the book description is represented by a predicate corresponding to its stem, and linked to both the book itself and the single slots in which it appears. For instance, predicate `prolog(slott, slottitleprolog)` indicates that the object `slottitleprolog` has stem “prolog” and is contained in slot `slott`; in such a case, also a literal `prolog(book)` is present to say that stem “prolog” is present in the book description. Formerly, INTHELEX was not able to handle numeric values; thus, a discretization was needed. In the new version, it can represent numeric information and manipulate numeric intervals, so the number of occurrences of each word in each slot was represented by means of a predicate `occ(Y,X)`, indicating that term `Y` occurs `X` times. Instead of learning a definition for each of the 10 possible votes, just two possible classes of interest are learnt: “likes”, describing that the user likes a book (rates from 6 to 10), and its opposite “not(likes)” (rates from 1 to 5). Such a discretization step is automatically carried out by an abstraction operator embedded in INTHELEX, whose cost is negligible since each numeric value is immediately mapped onto the corresponding discretized symbolic value.

Figure 1 shows an example for the class `likes`. The clause means that the user *likes* the book with `id=50147799` that contains:

- in the slot *title* a word whose stem is *practic* (one or two times) and a word whose stem is *prolog* (one or two times);
- in the slot *authors* the word *Lsterling* (one or two times);

```
likes(50147799) :-
  slot_title(50147799,slott), practic(slott,slottitlepractic),
  occ_1(slottitlepractic), occ_12(slottitlepractic),
  prolog(slott, slottitleprolog), occ_1(slottitleprolog),
  occ_12(slottitleprolog), slot_authors(50147799,slotau),
  l_sterling(slotau,slotauthorsl_sterling), occ_1(slotauthorsl_sterling),
  occ_12(slotauthorsl_sterling), slot_annotation(50147799, slotan),
  l_sterling(50147799), practic(50147799), prolog(50147799).
```

Fig. 1. First-Order Representation of a Book.

3.2 Item Recommender

ITR (ITem Recommender) implements a probabilistic learning algorithm to classify texts, the naïve Bayes classifier [7]. The prototype is able to classify text belonging to a specific category as interesting or uninteresting for a particular user. For example, the system could learn the target concept "textual descriptions the user finds interesting in the category Computer and Internet".

According to the Bayesian approach to classify natural language text documents, given a set of classes $C = \{c_1, \dots, c_{|C|}\}$, the conditional probability of a class c_j given a document d is calculated as follows:

$$P(c_j|d) = \frac{P(c_j)}{P(d)}P(d|c_j)$$

In our problem, we have only 2 classes: c_+ represents the positive class (user-likes, corresponding to ratings from 6 to 10), and c_- the negative one (user-dislikes, ratings from 1 to 5). Since instances are represented as a vector of documents, (one for each BOW), and assumed that the probability of each word is independent of the word's context and position, the conditional probability of a category c_j given an instance d_i is computed using the formula:

$$P(c_j|d_i) = \frac{P(c_j)}{P(d_i)} \prod_{m=1}^{|S|} \prod_{k=1}^{|b_{im}|} P(t_k|c_j, s_m)^{n_{kim}} \quad (7)$$

where $S = \{s_1, s_2, \dots, s_{|S|}\}$ is the set of slots, b_{im} is the BOW in the slot s_m of the instance d_i , n_{kim} is the number of occurrences of the token t_k in b_{im} .

In (7), since for any given document, the prior $P(d_i)$ is a constant, this factor can be ignored if the only interest concerns a ranking rather than a probability estimate. To calculate (7), we only need to estimate the terms $P(c_j)$ and $P(t_k|c_j, s_m)$, from the training set. Each instance is weighted according to the user rating r , normalized in order to obtain values ranging between 0 and 1:

$$w_+^i = \frac{r-1}{9}; \quad w_-^i = 1 - w_+^i \quad (8)$$

The weights in (8) are used for weighting the occurrences of a word in a document. Weights are used for estimating the two probability terms from the training set TR according to the following equations:

$$\hat{P}(c_j) = \frac{\sum_{i=1}^{|TR|} w_j^i}{|TR|} \quad (9) \quad \hat{P}(t_k|c_j, s_m) = \frac{\sum_{i=1}^{|TR|} w_j^i n_{kim}}{\sum_{i=1}^{|TR|} w_j^i |b_{im}|} \quad (10)$$

In (10), n_{kim} is the number of occurrences of the term t_k in the slot s_m of the i^{th} instance, and the denominator denotes the total weighted length of the slot

s_m in the class c_j . The length of b_{im} is computed by summing the occurrences of the words in the slot s_m of the i^{th} instance. Therefore, $\hat{P}(t_k|c_j, s_m)$ is calculated as a ratio between the weighted occurrences of the term t_k in slot s_m of class c_j and the total weighted length of the slot. The final outcome of the learning process is a probabilistic model used to classify a new instance in the class c_+ or c_- . The model can be used to build a personal profile including those words that turn out to be most indicative of the user's preferences, according to the value of the conditional probabilities in (10).

4 Experimental Sessions

The goal of the experiment has been the comparison of the above described methods in terms of classification accuracy when coping with the task of learning user profiles. The experiments have been carried out on a collection of textual book descriptions rated by real users according to their preferences. Eight book categories were selected at the Web site of a virtual bookshop. For each book category, a set of book descriptions was obtained by analyzing Web pages using an automated extractor and stored in a local database. Several users have been involved in the experiments: each user was requested to choose one or more categories of interest and to rate 40 or 80 books in each selected category, providing 1-10 discrete ratings. In this way, for each couple user-category, a dataset of 40 or 80 rated books was obtained (see Table 1). For each user we considered:

- *Rated books* - number of rated books with the indication of negative (rates in the range 1-5) and positive (rates in the range 6-10) ones;
- *Books with annotation* - number (and percentage) of books with a textual annotation (slot annotation not empty);
- *Avg. annotation length* - average length (in words) of the annotations;
- *Avg. rate μ / σ* - average rate provided by the user and standard deviation.

Table 1. Dataset information

User ID	Category	Rated books	Books with Annotations	Avg. Ann. Length	Avg. Rate μ / σ
37	SF, Horror & Fantasy	40 (22+, 18-)	40 (100%)	30.475	4.87 / 2.731
26	SF, Horror & Fantasy	80 (46+, 34-)	70 (87.5%)	19.512	5.49 / 3.453
30	Computer & Internet	80 (40+, 40-)	80 (100%)	56.425	5.31 / 2.462
35	Business	80 (30+, 50-)	78 (97.5%)	64.150	4.21 / 3.488
24c	Computer & Internet	80 (38+, 42-)	76 (95%)	49.100	5.71 / 3.174
36	Fiction & literature	40 (25+, 15-)	40 (100%)	40.225	5.87 / 1.805
24f	Fiction & literature	40 (27+, 13-)	38 (95%)	45.500	6.40 / 2.662
33	Sport & leisure	80 (35+, 45-)	49 (61.25%)	23.337	4.34 / 3.342
34	Fiction & literature	80 (42+, 38-)	70 (87.5%)	44.925	5.61 / 2.492
23	Fiction & literature	40 (39+, 1-)	36 (90%)	45.875	7.25 / 1.089

Table 2. Performance of the Rocchio algorithms on 10 different datasets. Notice that 9 users rated books in one category, while user 24 rated books in two categories.

Id	Precision		Recall		F1		Accuracy	
	$\beta = 16$	New	$\beta = 16$	New	$\beta = 16$	New	$\beta = 16$	New
	$\gamma = 4$	Rocchio	$\gamma = 4$	Rocchio	$\gamma = 4$	Rocchio	$\gamma = 4$	Rocchio
37	0.641	0.925	1	0.717	0.781	0.808	0.675	0.800
26	0.797	0.845	1	0.830	0.887	0.837	0.837	0.812
30	0.500	0.534	1	0.875	0.666	0.663	0.500	0.550
35	0.391	0.690	1	0.700	0.562	0.695	0.412	0.762
24c	0.471	0.675	0.966	0.583	0.633	0.626	0.475	0.687
36	0.616	0.767	0.916	0.700	0.737	0.732	0.600	0.675
24f	0.675	0.825	1	0.833	0.805	0.829	0.675	0.750
33	0.651	0.743	0.966	0.917	0.778	0.821	0.737	0.812
34	0.525	0.644	0.975	0.645	0.682	0.644	0.525	0.625
23	0.966	0.975	0.966	0.975	0.966	0.975	0.950	0.950
Mean	0.623	0.762	0.979	0.777	0.750	0.763	0.639	0.742

We can observe that the number of books rated as positive and negative for each user is balanced, except for the user 23 that has rated 39 books as positive and only 1 as negative. Then, almost the totality of books rated by the users contain annotations: the user 33 is the only one with a low percentage. As far as the average annotation length we can notice that only users 26 and 33 have values lower than the others. On each dataset a 10-fold cross-validation was run and several metrics were used in the testing phase. In the evaluation phase, the concept of *relevant book* is central. A book in a specific category is considered as relevant by a user if the rating is greater than 5. This corresponds in the Rocchio-based profiling algorithm to having similarity greater than 0; ITR classifies a book d_i as interesting if $P(c_+|d_i) \geq 0.5$, calculated as in equation (7). Symmetrically, INTHELEX considers as relevant books covered by the inferred theory. Classification effectiveness is measured in terms of the classical Information Retrieval notions of *precision*, *recall* and *accuracy*, adapted to the case of text categorization [8]. *Precision* is the proportion of items classified as relevant that are really relevant, and *recall* is the proportion of relevant items that are classified as relevant; *accuracy* is the proportion of items that are correctly classified as relevant or not.

4.1 Discussion

Table 2 shows the results of the experiments using the modified version of the Rocchio algorithm and the classical Rocchio one obtained by setting the values of the control parameters β and γ according to the literature [5]. We have carried out a pairwise comparison of the methods, using the nonparametric Wilcoxon signed rank test [9]. Requiring a significance level $p < 0.05$, the test revealed that there is a statistically significant difference in performance both for precision and accuracy in favor of the modified Rocchio and for recall in favor of

Table 3. Performance of the systems on 10 different datasets.

Id	Precision			Recall			F1			Accuracy		
	ITR	INTH.	New Rocchio	ITR	INTH.	New Rocchio	ITR	INTH.	New Rocchio	ITR	INTH.	New Rocchio
37	0.767	0.967	0.925	0.883	0.500	0.717	0.821	0.659	0.808	0.731	0.695	0.800
26	0.818	0.955	0.845	0.735	0.645	0.830	0.774	0.770	0.837	0.737	0.768	0.812
30	0.608	0.583	0.534	0.600	0.125	0.875	0.604	0.206	0.663	0.587	0.488	0.550
35	0.651	0.767	0.690	0.800	0.234	0.700	0.718	0.359	0.695	0.725	0.662	0.762
24c	0.586	0.597	0.675	0.867	0.383	0.583	0.699	0.467	0.626	0.699	0.599	0.687
36	0.783	0.900	0.767	0.783	0.300	0.700	0.783	0.450	0.732	0.700	0.513	0.675
24f	0.785	0.900	0.825	0.650	0.350	0.833	0.711	0.504	0.829	0.651	0.535	0.750
33	0.683	0.750	0.743	0.808	0.308	0.917	0.740	0.437	0.821	0.730	0.659	0.812
34	0.608	0.883	0.644	0.490	0.255	0.645	0.543	0.396	0.644	0.559	0.564	0.625
23	0.500	0.975	0.975	0.130	0.900	0.975	0.206	0.936	0.975	0.153	0.875	0.950
Mean	0.679	0.828	0.762	0.675	0.400	0.777	0.662	0.520	0.763	0.627	0.636	0.742

the classical Rocchio method ($\beta = 16$, $\gamma = 4$), but not as regards F1. These results led us to conclude that the new method is more effective than the traditional one in the e-commerce domain, due to the fact that *trust* is a key word in giving recommendations: the systems should minimize false positive errors. This means that in the e-commerce domain it is better to provide users with a few number of high quality recommendations than to overload users with many recommendations that they should manually filter.

Table 3 shows the results of the second experiment aimed at comparing the new Rocchio method with the ones implemented by INTHELEX and ITR in terms of average precision, recall, F1 and accuracy of the models learned in the 10 folds for each dataset. The last row of the table reports the mean values, averaged on all datasets. The results of INTHELEX and ITR are described in more details in [10]. The most important result is that the proposed method outperforms the other ones as regards accuracy. It is surprising to observe that the algorithm reaches high values of precision and recall for the users 26 and 37, even if the average annotation lengths of the documents rated by the users are among the shortest in the dataset. This means that the profiles contain few words for computing similarity on new documents, but these words are indicative of the users' preferences. In general, the new Rocchio algorithm outperforms ITR in precision, but not INTHELEX (requiring a significance level $p < 0.05$ the systems are equivalent, even if the test revealed a better ranking for INTHELEX values). Another remark worth noting is that theories learned by the symbolic system are very interesting from a human understandability viewpoint, in order to be able to explain and justify the recommendations provided by the system. From what said above, it seems that the approaches compared in this paper have complementary *pros and cons*, not only as regards the representation language, but also as concerns the predictive performances. This naturally leads to think that some cooperation could take place in order to reach higher effectiveness of the recommendations. For instance, since the new Rocchio method has a better

accuracy, it could be used for selecting which items are to be presented to the user. Then, some kind of filtering could be applied on them, in order to present to the user first those items that are considered positive by the symbolic theories, that are characterized by a slightly better precision.

5 Conclusions

The paper presented different approaches for learning user profiles. We proposed a new Rocchio-based method able to discover user preferences from the analysis of textual descriptions of items in online catalogues of e-commerce Web sites. In order to evaluate the effectiveness of the approach, we have performed an intensive experimental session involving real users. Results have been compared to the performance of an ILP approach and a probabilistic one. The comparison highlighted the usefulness and drawbacks of each method, suggesting possible ways of integrating the approaches in order to offer better support to users.

6 Acknowledgments

This research was partially funded by the Italian Ministry of Education (MIUR) under the PRIN-2003 National Project “Tecniche di intelligenza artificiale per il reperimento di informazione di qualità sul Web” and by the European Commission under the IST-2003-507173 Project VIKEF (<http://www.vikef.net>).

References

1. Rocchio, J.: Relevance feedback information retrieval. In Salton, G., ed.: The SMART retrieval system - experiments in automated document processing, Prentice-Hall, Englewood Cliffs, NJ (1971) 313–323
2. Mladenic, D.: Text-learning and related intelligent agents: a survey. *IEEE Intelligent Systems* **14** (1999) 44–54
3. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: Proceedings of the 5th ACM Conference on Digital Libraries, San Antonio, US, ACM Press, New York, US (2000) 195–204
4. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. *Machine Learning* **27** (1997) 313–331
5. Sebastiani, F.: Machine learning in automated text categorization. *ACM Computing Surveys* **34** (2002)
6. Semeraro, G., Esposito, F., Malerba, D., Fanizzi, N., Ferilli, S.: A logic framework for the incremental inductive synthesis of datalog theories. In Fuchs, N.E., ed.: *Logic Program Synthesis and Transformation*. Number 1463 in *Lecture Notes in Computer Science*, Springer-Verlag (1998) 300–321
7. Mitchell, T.: *Machine Learning*. McGraw-Hill, New York (1997)
8. Salton, G., McGill, M.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1983)
9. Orkin, M., Drogin, R.: *Vital Statistics*. McGraw-Hill, New York (1990)
10. Esposito, F., Semeraro, G., Ferilli, S., Degemmis, M., Mauro, N.D., Basile, T., Lops, P.: Evaluation and validation of two approaches to user profiling. In: Proc. of the ECML/PKDD-2003 First European Web Mining Forum. (2003) 51–63