CLASSIFICATION PROCESS IN A TEXT DOCUMENT RECOMMENDER SYSTEM

Dan MUNTEANU, Severin BUMBARU

"Dunărea de Jos" University of Galatz
Faculty of Computer Science
Department of Computers and Applied Informatics
111 Domnească Street, 800201-Galatz, Romania
Phone/Fax: (+40) 236 460182; (+40) 236 461353
E-mail: dan.munteanu@ugal.ro severin.bumbaru@ugal.ro

Abstract: this paper presents the classification process in a recommender system used for textual documents taken especially from web. The system uses in the classification process a combination of content filters, event filters and collaborative filters and it uses implicit and explicit feedback for evaluating documents.

Keywords: information systems, information analysis, algorithms, machine learning, agents

1. INTRODUCTION. INFORMATION MANAGEMENT ON WEB

With the venue of the information society, knowledge is being leveraged from the individual stage to the community level at a pace never wondered before. Information, the precious raw material of the digital age, has never been so easy to obtain, process and disseminate through the Internet. Yet, with the avalanche of information at our doors, there is a rapidly increasing difficulty of finding what we want, when we need it, and in a way that better satisfies our requirements.

Recommender systems make a recommendation for a specific object by using evaluations for that object made by other users with similar interests. Examples of such systems are movie recommender systems like Moviefinder, MovieLens and Movie Critic, music recommender systems like CDNow's Album Advisor, Launch and book recommender systems like Amazon's Recommendation Center, Barnes and Noble's Recommended Reads. These systems ignore any information that can be extracted from the content.

This paper tries to present a recommender system that combine content filtering, collaborative filtering and agent technology. Every user has a personal agent which helps him to classify the information found on Internet and the information he had on his personal computer and also helps at recommending the documents to other users with similar interests. The agent suggests a classification of a document and extracts ratings for every document by analyzing user's actions (accept, reject, and modify agent's suggestion).

2. RECOMMENDER SYSTEMS

Recommender systems were introduced as a computer-based intelligent technique to deal with the problem of information and product overload.

The two basic entities which appear in any Recommender System are the user and the item. A user is a person who utilizes the Recommender System providing his opinion about various items and receives recommendations about new items from the system.

The input of a Recommender System depends on the type of the employed filtering algorithm. Generally, the input belongs to one of the following categories:

1. Ratings (also called votes), which express the opinion of users on items. Ratings are normally provided by the user and follow a specified numerical scale (example: 1-bad to 5-excellent). A common rating scheme is the binary rating scheme, which allows only ratings of either 0 or

This paper was recommended for publication by Cornelia NOVAC

1. Ratings can also be gathered implicitly from the web logs, hyperlink visits, browsing habits or other types of information access patterns.
2. Demographic data, which refer to information such as the age, the gender and the education of the users. This kind of data is usually difficult to obtain. It is normally collected explicitly from the user.
3. Content data, which are based on a textual analysis of documents related to the items rated by the user. The features extracted by this analysis are used as input to the filtering algorithm in order to infer a user profile.

The goal of Recommender Systems is to generate suggestions about new items or to predict the utility of a specific item for a particular user.

The output of a Recommender System can be either a Prediction or a Recommendation (Vozalis and Konstantinos, 2003). A Prediction is expressed as a numerical value, representing the anticipated opinion of active user for a specific item. This form of Recommender Systems output is also known as Individual Scoring. A Recommendation is expressed as a list of *N* items, where N is lower then the number of items, which the active user is expected to like the most. This form of Recommender Systems output is also known as Top-N Recommendation.

Relevance can be defined for a particular user and in the context of a particular subject.

A popular method to keep useful information from Internet is represented by using bookmark manager from web browser. These systems have some drawbacks:

- lack of immediate portability
- lack of visibility from different locations
- difficult management

It's reasonable to suppose that if anyone adds an URL to a bookmark manager, save a document from Internet, print a document is because he is interested in the information contained by that document.

Documents and user profiles are represented using keywords vectors for comparing and learning. For a specific user, processing a lot of relevant documents correctly classified and irrelevant documents from a domain can lead to identification of the relevant terms for that domain.

This system has two major components: one for classification and the other for recommendation. For classification it will use a text classification algorithm based on Rocchio's algorithm (Salton and Buckley, 1990). The difference is that the keywords used for representing the domain can be added and modified. The classifier uses relevance feedback (Douglas and Jinmook, 1998) when a document is added to the database by using implicit evaluation of the document. For updating the classifiers (that are used in the process of classification) the system uses the information gain measure to select the most informative keywords. The keywords will be words and roots of the words that are obtained using the Porter's stemming algorithm (Porter, 1980). A text classifier contains a number of keywords (128) that are manually selected (28) and the rest are extracted from the well classified documents.

This system has two major components: one for classification and the other for recommendation. For classification it will use a text classification algorithm based on Rocchio's algorithm (Salton and Buckley, 1990). The difference is that the keywords used for representing the domain can be added and modified. The classifier uses relevance feedback (Douglas and Jinmook, 1998) when a document is added to the database by using implicit evaluation of the document. For updating the classifiers (that are used in the process of classification) the system uses the information gain measure to select the most informative keywords. The keywords will be words and roots of the words that are obtained using the Porter's stemming algorithm (Porter, 1980). A text classifier contains a number of keywords (128) that are manually selected (28) and the rest are extracted from the well classified documents.

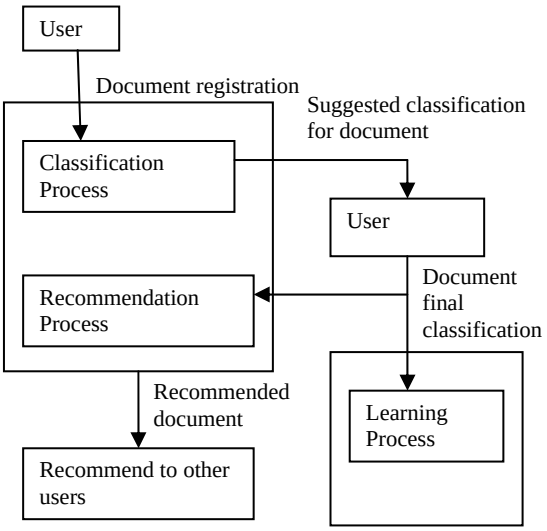

Fig. 1. System Architecture

The recommendation process uses a modified *Pearson-r* algorithm (Breese, 1998), computing the correlation between users and modifying by adding the correlation between categories. The Pearson correlation coefficient was first defined in the context of the GroupLens project (Resnick *et al.*, 1994) as the basis for the weights.

The goal of the system is to assist the user in the process of classifying web documents and to automatically recommend them to other user with similar interest.

The system has a number of *n* categories to classify a document. From here the term category is considered to be similar with class, topic. In the same way document will represent web page, web document and bookmark.

The system contains a database with bookmarks and references to local documents for each user and an agent that monitors the user's actions. When a document is registered, the agent suggests a classification in a category by analyzing the content of the document and user's profiles. The user can confirm the suggestion or choose another category which he considers to be better. In the meantime the agent checks to see if there are new bookmarks and recommends them to other users.

In the registration process the user has to select the areas of interest. With this information an initial profile is build for every category. The agent modifies the classifier of a category when a number of *k* documents have been correctly classified in it.

2.1 Classification

In general, classification (Pierre *et al*., 2003) consists of learning a mapping that can classify a set of measurements on an object, such as a *d*-dimensional vector of attributes, into one of a finite number *K* of classes or categories. This mapping is referred to as a *classifier* and it is typically learned from training data. Each training instance (or data point) consists of two parts: an input part *x* and an associated output class "target" *c*, where $c \in \{1,2,\ldots,K\}$. The classifier mapping can be thought of as a function of the inputs, *g(x)*, which maps any possible input x into an output class *c*.

$$(1)\ \mathrm{D} = \{[\mathrm{x}_1, \mathrm{c}_1], \ldots, [\mathrm{x}_\mathrm{n}, \mathrm{c}_\mathrm{n}]\}$$

The goal of classification learning is to take a training data set D and estimate the parameters of a classification function *g(x)*. Typically we seek the best function *g*, from some set of candidate functions, that minimizes an empirical loss function, namely

$$(2)\ \varepsilon = \sum_{i=1}^{n} l(c_i, g(x_i)),$$

where $l(c_i, g(x_i))$ is defined as the loss that is incurred when our predicted class label is $g(x_i)$, given input $g(x_i)$, and the true class label is c_i.

There are two broad approaches to classification, the probabilistic approach and the discriminative or decision-boundary approach.

In the probabilistic approach can be learned a probability distribution $P(x \mid c)$ for each of the *K* classes, as well as the marginal probability for each class $P(c)$. This can be done straightforwardly by dividing the training data *D* into *K* different subsets according to class labels, assuming some functional form for $P(x \mid c)$ for each class, and then using ML or Bayesian estimation techniques to estimate the parameters of $P(x \mid c)$ for each of the *K* classes. Once these are known we can then use Bayes' rule to calculate the posterior probability of each of the classes, given an input *x*:

$$(3)\ P(c = k \mid x) = \frac{P(x \mid c = k)P(c = k)}{\sum_{j=1}^{K} P(x \mid c = j)P(c = j)}$$
$$1 \le k \le K,$$

To make a class label prediction for a new input that is not in the training data *x*, $P(c = k \mid x)$ must be calculated for each of the *K* classes.

The other alternative to the probabilistic approach is to simply seek a function *f* that optimizes the relevant empirical loss function, with no direct reference to probability models. If *x* is a *d*-dimensional vector where each attribute is numerical, these models can often be interpreted as explicitly searching for (and defining) decision regions in the *d*-dimensional input space *x*. There are many non-probabilistic classification methods available, including perceptrons, support vector machines, kernel approaches, and classification trees.

It is important to note that the ultimate goal in building a classifier is to be able to do well on predicting the class labels of new items, not just the items in the training data.

2.2 Text classification (categorization)

Document classification may be seen (Sebastiani, 1999) as the task of determining an assignment of a value from $\{0,1\}$ to each entry of the *decision matrix*: where $C = \{c_1, \ldots c_m\}$ g is a set of predefine categories, and $D = \{d_1, \ldots d_n\}$ is a set of documents to be classified.

	d_1	…	d_j	…	d_n
c_1	$a_{1,1}$	…	$a_{1,j}$	…	$a_{1,n}$
…	…	…	…	…	…
c_i	$a_{i,1}$	…	$a_{i,j}$	…	$a_{i,n}$
…	…	…	…	…	…
c_m	$a_{m,1}$	…	$a_{m,j}$	…	$a_{m,n}$

Fig. 2. Decision Matrix:

A value of 1 for $a_{i,j}$ is interpreted as a decision to file d_j under c_i and a value of 0 is interpreted as a decision not to file d_j under c_i.

For understanding this task some observations can be made:

- the categories are just symbolic labels. No additional knowledge of their "meaning" is available to help in the process of building the classifier in particular, this means that the "text" constituting the label (e.g. Sports in a news classification task) cannot be used;
- the attribution of documents to categories should, in general, be attributed on the basis of the content of the documents, and not on the basis of metadata (e.g. publication date, document type, etc.) that may be available from an external source. This means that the notion of relevance of a document to a category is inherently subjective.

Different constraints may be enforced on the classification task, depending on the application:

1. $\{\leq 1 | 1 | \geq 1 | \ldots\}$ elements of *C* must be assigned to each element of *D*. When exactly one category is assigned to each document, this is often referred to as the non-overlapping categories case.
2. each element of *C* must be assigned to $\{\leq 1 | 1 | \geq 1 | \ldots\}$ elements of *D*.

2.3 Classification process for recommender system

Text classification is the automatic categorization of texts into topical categories. In this case, by using relevance feedback the topical categories are modified.

It is assumed (Sebastiani, 1999) that the categories are just symbolic labels, and no additional knowledge (of a procedural or declarative nature) of their meaning is available.

It is also assumed that no exogenous knowledge (i.e. data provided for classification purposes by an external source) is available; therefore, classification must be accomplished on the basis of endogenous knowledge only (i.e. knowledge extracted from the documents). In particular, this means that metadata such as e.g. publication date, document type, publication source, etc. is not assumed to be available.

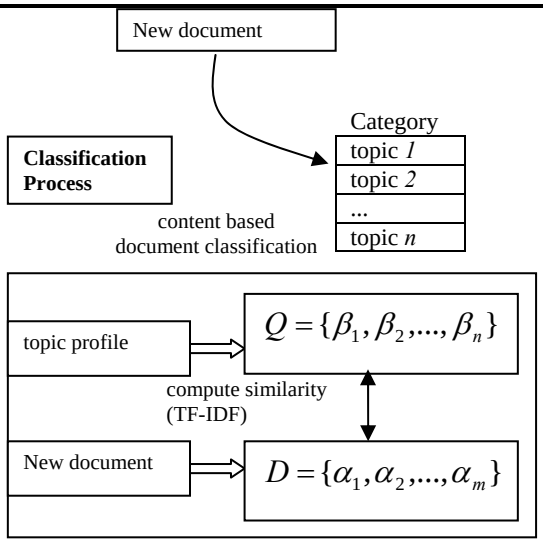

Fig. 3. Content text-based classification

2.4 The construction of text classifiers

The system is used by many users; each user registers documents of different types. A document belonging to user u_i is represented as a list of the most informative keywords from that document.

Positive examples for user u_i and class c_j are the documents explicitly registered and accepted by the user u_i in class c_j. Negative examples are deleted or misclassified bookmarks, or rejected recommendations, which are classified in category c_j.

Notations:

$C_{i,j}^{+}$ – documents classified as positive examples for user u_i and class c_j;

$C_{i,j}^{-}$ – documents classified as negative examples for user u_i and class c_j.

For each user and for each category a classifier *Q* it is build as a list of keywords (the system will contain *m* x *n* classifiers – where *m* is the numbers of users and *n* is the numbers of categories). The number of categories (domains, classes) is fixed, *n*. The scope is to apply the similarity measure:

$$(4)\ sim(Q, D) = \sum_{\tau \in Q} w(\tau, Q) w(\tau, D)$$

To compute the term weight the TF-IDF (Term Frequency – Inverse Document Frequency) algorithm it is used (Tokunaga and Iwayama 1994).

$$(5)\ w(\tau,d)=\frac{tf_{\tau,d}\left[\log_2\left(\frac{N}{df_\tau}\right)+1\right]}{\sqrt{\sum_i\left(tf_{\tau_i,d}\left[\log_2\left(\frac{N}{df_{\tau_i}}\right)+1\right]\right)^2}}$$

Where $tf_{\tau,d}$ is term frequency of τ in document *d*,

$$(6)\ idf_\tau = \log_2(N/dt_\tau)+1$$

is the inverse document frequency, *N* is the total number of documents, dt_τ is the number of appearances of term τ in collection. For each user a separate collection will be kept.

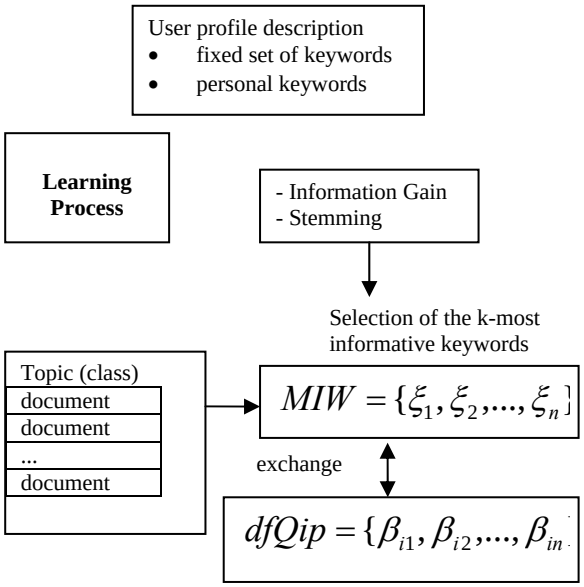

Fig. 4. Learning classifiers

2.5 Relevance Feedback

Let

$$(7)\ P=\{\tau_1,\tau_2,...,\tau_r\}$$

be the set of terms used for updating the classifiers.

The classifier and the document are represented by numeric vectors that contain the frequency of each term from P in them.

$$(8)\ Tf(Q)=<tf_{\tau_1,Q},f_{\tau_1,Q},...,f_{\tau_r,Q}>$$

and

$$(9)\ Tf(D)=<tf_{\tau_1,D},f_{\tau_1,D},...,f_{\tau_r,D}>$$

Relevance feedback can be described mathematically:

$$(10)\ Tf(Q^{i+1})=Tf(Q^i)+\alpha Tf(D)$$

In which $\alpha=1$, if $D\in C^+_{i,j}$ and $\alpha=-1$, if $D\in C^-_{i,j}$

Then Q^{i+1} is recomputed using values from $Tf(Q^{i+1})$.

The problem is that the dimension of vectors *Q* and *D* cannot be changed.

An algorithm will be used, to build the classifier *Q* step by step.

Notations:

$$(11)\ P^{(t)+}_{i,j}=\{\tau_1,\tau_2,...,\tau_r\}$$

is a set of unique terms, relevant for class c_i until the time moment *t*; $P^{(t)-}_{i,j}$ a subset of $P^{(t)+}_{i,j}$ in which every element is found in the set of negative examples for class c_i.

A possible algorithm to be used for constructing Q^+ is proposed by Ishii (Delgado and Ishii, 2000) and is presented next:

At the moment *t*+1

if ($P^{(t+1)+}_{i,j}=P^{(t)+}_{i,j}$)

then

$Tf(Q^{(t+1)+}_{i,j})=Tf(Q^{(t)+}_{i,j})+Tf(D^{(t)+}_{i,j})$

modify $Q^{(t+1)+}_{i,j}$ using

$Tf\ (Q^{(t+1)+}_{i,j})$

else

for each $\tau\in P^{(t+1)+}_{i,j}-P^{(t)+}_{i,j}$

compute $w(\tau,d)$ for the most recent *n* documents $\in C^+_{i,j}$ and modify these values in $Q^{(t+1)+}_{i,j}$

Where *n* is the number of documents used for updating.

The algorithm it is applied in the same way for the negative classifier Q^-.

The similarity between class c_j and document *D* is:

$$(12)\ sim_i^t(c_j, D) = (Q_{i,j}^{(t)+} \cdot D_{i,j}^{(t)}) - (Q_{i,j}^{(t)-} \cdot D_{i,j}^{(t)})$$

This equation tells that a document is similar to a class if it is similar with the positive classifier and is not similar with the negative classifier. The category with the highest score for similarity will be chosen.

2.6 Selection of terms for updating the classifier

Information Gain method is used to select the most informative terms from the documents collection.

$$(13)\ E_{i,j}(\tau, S) = I(S) - [P(\tau = prezent) I(S_{\tau=prezent}) + P(\tau = absent) I(S_{w=absent})]$$

$$(14)\ I_{i,j}(S) = \sum_{c \in \{C_{i,j}^+, C_{i,j}^-\}} -P(S_c) \log_2(p(S_c))$$

Where $P(\tau = prezent)$ is the probability that τ to be present in a document $S_{\tau = prezent}$ is the set of documents that contains at least one appearance of τ and S_c are the documents that belong to the class *c*.

The agent finds first *k* most informative terms from the set S of the last *n* classified documents. Pazzani in the Syskill & Webbert project (Pazzani *et al*., 1994) have proposed $k = 128$ and $n = 3$. The classifier contains 128 terms, from which 28 are fixed. For the rest of the terms the next method is used to add/delete terms in the positive classifier:

1. stemming algorithm is applied to extract stems(roots) of words.
2. the terms that are in the classifier and not in the list of the most informative words are replaced
3. the weights of added/replaced terms are updated (using *n* processed documents)

3. CONCLUSIONS AND FUTURE WORK

This paper has described a recommender system for textual documents (from web) and the accent was put on the classification process.

In the future the classification algorithm shall be modified for updating categories (user will be able to create a new category which belongs to him).

The efficiency of the system is (theoretically) better if the number of users that use the system is high and if each if the number of documents registered by each user is also high.

4. REFERENCES

Breese, J., D. Heckerman and C. Kadie, (1998). *Empirical Analysis of Predictive Algorithms for ollaborative Filtering*. In: *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*.

Delgado, J. and N. Ishii (2000), *Memory Based Weighted-Majority Prediction*, Department of Intelligence & C.S., Nagoya Institute of Technology, JAPAN

Douglas W. O. and K. Jinmook (1998). *Implicit Feedback for Recommender Systems*. Digital Library Research Group, College of Library and Information Services, University of Maryland

Pazzani M., J. Muramatsu and D. Billsus (1996). *Syskill & Webert: Identifying interesting websites*. In: *Proceedings of the American National Conference on Artificial Intelligence (AAAI'96), Portland, OR.*

Pierre B., F. Paolo and S. Padhraic (2003), *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*, JohnWiley & Sons Ltd, West Sussex, England

Porter, M.F. (1980). *An Algorithm For Suffix Stripping In*. In: *Program 14 (3),* pp. 130-137.

Resnick, P., N. Iacovou, M. Sushak, P. Bergstrom and J. Riedl (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*. In the *Proceedings of the CSCW 1994 conference*.

Salton, G. and C. Buckley (1990). *Improving retrieval performance by relevance feedback*. In *Journal of the American Society for Information Science* Vol. 41, pp. 288-297

Sebastiani, F. (1998) *A Tutorial on Automated Text Categorisation, Istituto di Elaborazione dell'Informazione*, Consiglio Nazionale delle Ricerche

Tokunaga, T. and M. Iwayama (1994). *Text categorization based on weighted inverse document frequency*. In *Technical Report 94-TR0001*, Department of Computer Science, Tokyo Institute of Technology

Vozalis E. and M. Konstantinos (2003). *Analysis of Recommender Systems' Algorithms*. Presented at the Sixth Hellenic-European Conference on Computer Mathematics and its Applications (HERCMA), Athens, Greece