Preprocessing and Mining Web Log Data for Web Personalization

M. Baglioni[1], U. Ferrara[2], A. Romei[1], S. Ruggieri[1], and F. Turini[1]

[1] Dipartimento di Informatica, Università di Pisa,
Via F. Buonarroti 2, 56125 Pisa Italy
{baglioni,romei,ruggieri,turini}@di.unipi.it

[2] KSolutions S.p.A.
Via Lenin 132/26, 56017 S. Martino Ulmiano (PI) Italy
ferrara@ksolutions.it

Abstract. We describe the web usage mining activities of an on-going project, called ClickWorld[3], that aims at extracting models of the navigational behaviour of a web site users. The models are inferred from the access logs of a web server by means of data and web mining techniques. The extracted knowledge is deployed to the purpose of offering a personalized and proactive view of the web services to users. We first describe the preprocessing steps on access logs necessary to clean, select and prepare data for knowledge extraction. Then we show two sets of experiments: the first one tries to predict the sex of a user based on the visited web pages, and the second one tries to predict whether a user might be interested in visiting a section of the site.

Keywords: knowledge discovery, web mining, classification.

1 Introduction

According to [10], *Web Mining* is *the use of data mining techniques to automatically discover and extract information from web documents and services.* A common taxonomy of web mining defines three main research lines: *content mining*, *structure mining* and *usage mining*. The distinction between those categories is not a clear cut, and very often approaches use combination of techniques from different categories.

Content mining covers data mining techniques to extract models from web object contents including plain text, semi-structured documents (e.g., HTML or XML), structured documents (digital libraries), dynamic documents, multimedia documents. The extracted models are used to classify web objects, to extract keywords for use in information retrieval, to infer structure of semi-structured or unstructured objects.

Structure Mining aims at finding the underlying topology of the interconnections between web objects. The model built can be used to categorize and to rank web sites, and also to find out similarity between them.

[3] Progetto di ricerca industriale *ClickWorld - Tecniche innovative di Web Proactive Personalization a fini di Internet marketing, commercio elettronico, editoria on-line*, Codice Progetto: S585-P, Ente finanziatore: MIUR Legge 488/92.

Usage mining is the application of data mining techniques to discover usage patterns from web data. Data is usually collected from user's interaction with the web, e.g. web/proxy server logs, user queries, registration data. Usage mining tools [3,4,9,15] discover and predict user behavior, in order to help the designer to improve the web site, to attract visitors, or to give regular users a personalized and adaptive service.

In this paper, we describe the web usage mining activities of an on-going project, called `ClickWorld`, that aims at extracting models of the navigational behavior of users for the purpose of web site personalization [6]. We have collected and preprocessed access logs from a medium-large national web portal, `vivacity.it`, over a period of five months. The portal includes a national area (`www.vivacity.it`) with news, forums, jokes, etc., and more than 30 local areas (e.g., `www.roma.vivacity.it`) with city-specific information, such as local news, restaurant addresses, theatre programming, bus timetable, ecc.

The preprocessing steps include data selection, cleaning and transformation and the identification of users and of *user sessions* [2]. The result of preprocessing is a data mart of web accesses and registration information. Starting from preprocessed data, web mining aims at pattern discovery by adapting methods from statistics, data mining, machine learning and pattern recognition. Among the basic data mining techniques [7], we mention *association rules*, discovering groups of objects that are frequently requested together by users; *clustering*, grouping users with similar browsing patterns, or grouping objects with similar content or access patterns; *classification*, where a profile is built for users belonging to a given class or category; and *sequential patterns*, namely sequences of requests which are common for many users.

In the `ClickWorld` project, several of the mentioned methods are currently being used to extract useful information for proactive personalization of web sites. In this paper, we describe two sets of classification experiments. The first one aims at extracting a classification model able to discriminate the sex of a user based on the set of web pages visited. The second experiment aims at extracting a classification model able to discriminate those users that visit pages regarding e.g. sport or finance from those that typically do not.

2 Preprocessing for Web Personalization

We have developed a data mart of web logs specifically to support web personalization analysis. The data mart is populated starting from a web log data warehouse (such as those described in [8,16]) or, more simply, from raw web/proxy server log files. In this section, we describe a number of preprocessing and coding steps performed for data selection, comprehension, cleaning and transformation. While some of them are general data preparation steps for web usage mining [2,16], it is worth noting that in many of them a form of domain knowledge must necessarily be included in order to clean, correct and complete the input data according to the web personalization requirements.

2.1 User registration data

In addition to web access logs, our given input includes personal data on a subset of users, namely those who are registered to the `vivacity.it` website (registration is not mandatory). For a registered user, the system records the following information: sex, city, province, civil status, born date. This information is provided by the user in a web form at the time of registration and, as one could expect, the quality of data is up to the user fairness. As preprocessing steps, improbable data are detected and removed, such as born data in the future or in the remote past. Also, some additional input fields were not imported in the data mart since almost all values were left as the default choice in the web form. In other words, the fields were considered not to be useful in discriminating user choices and preferences.

In order to avoid users to digit their login and password at each visit, the `vivacity.it` web site adopts cookies. If a cookie is provided by the user browser, then authentication is not required. Otherwise, after authentication, a new cookie is sent to the user browser. With this mechanism, it is possible to track any user as long as she deletes the cookies on her system. In addition, if the user is registered, the association login-cookie is available in the input data, and then it is possible to track the user also after she deletes the cookies.

This mechanism allows for detecting non-human users, such as system diagnosis and monitoring programs. By checking the number of cookies assigned to each user, we discovered that the user login 'test009' was assigned more than 24.000 distinct cookies. This is possible only if the user is some program that automatically deletes assigned cookies, e.g. a system diagnosis program.

2.2 Web URL

Resources in the World Wide Web are uniformly identified by means of URLs (Uniform Resource Locators). The syntax of an http URL is:

```
'http://' host.domain [':'port] [ abs_path ['?' query]]
```

where

- `host.domain[:port]` is the name of the server site. The TCP/IP `port` is optional (the default port is 80),
- `abs_path` is the absolute path of the requested resource in the server filesystem. We further consider `abs_path` of the form `path '/' filename ['.' extension]`, i.e. consisting of the filesystem path, filename and file extension.
- `query` is an optional collection of parameters, to be passed as an input to a resource that is actually an executable program, e.g. a CGI script.

On the one side, there are a number of normalizations that must be performed on URLs, in order to remove irrelevant syntactic differences (e.g., the host can be in IP format or host format – `131.114.2.91` is the same host as `kdd.di.unipi.it`). On the other side, there are some web server programs that adopt non-standard formats for passing parameters. The `vivacity.it` web server program is one of them. For instance, in the following URL:

`http://roma.vivacity.it/speciali/EditColonnaSpeciale/1,3478,|DX,00.html`

the file name `1,3478,|DX,00` contains a code for the local web site (`1` stands for `roma.vivacity.it`), a web page id (`3478`) and its specific parameters (`DX`).

The form above has been designed for efficient machine processing. For instance, the web page id is a key for a database table where the page template is found, while the parameters allow for retrieving the web page content in some other table. Unfortunately, this is a nightmare when mining clickstream of URLs. Syntactic features of URLs are of little help: we need some semantic information, or ontology [5,13], assigned to URLs.

At the best, we can expect that an *application-level* log is available, i.e. a log of accesses to semantic-relevant objects. An example of application-level log is one recording that the user entered the site from the home page, then visited a sport page with news on a soccer team, and so on. This would require a system module monitoring user steps at a semantic level of granularity. In the `ClickWorld` project such a module is called `ClickObserve`. Unfortunately, however, the module is a deliverable of the project, and it was not available for collecting data at the beginning of the project.

Therefore, we decided to extract both syntactic and semantic information from URLs via a semi-automatic approach. The adopted approach consists in *reverse-engineering* URLs, starting from the web site designer description of the meaning of each URL path, web page id and web page parameters. Using a PERL script, starting from the designer description we extracted from original URLs the following information:

- local web server (i.e., `vivacity.it` or `roma.vivacity.it` etc.), which provides us with some spatial information about user interests;
- a first-level classification of URLs into 24 types, some of which are: home, news, finance, photogalleries, jokes, shopping, forum, pubs;
- a second-level classification of URLs depending on the first-level one, e.g. URLs classified as *shopping* may be further classified as *book shopping* or *pc shopping* and so on;
- a third-level classification of URLs depending on the second-level one, e.g. URLs classified as *book shopping* may be further classified as *programming book shopping* or *narrative book shopping* and so on;
- a parameter information, further detailing the three level classification, e.g. URLs classified as *programming book shopping* may have the ISBN book code as parameter;
- the depth of the classification, i.e. 1 if the URL has only a first-level classification, 2 if the URL has first and second-level classification, and so on.

Of course, the adopted approach was mainly an heuristics one, with the hierarchical ontology designed at posteriori. Also, the designed ontology does not exploit any content-based classification, i.e. the description of an elementary object such as *sport news with id 12345* is its code (i.e., first-level is *news*, second-level is *sport*, parameter information *12345*), with no reference to the content of the news (was the news reporting about any specific player?).

2.3 Web Logs

Log files produced by web/proxy servers are text files with a row for each HTTP transaction. A typical row contains the following information:

```
...  213.213.31.41 [15/Apr/2000:04:00:04 +0200]
  "GET http://www.unipi.it/images/h/h_home.gif HTTP/1.1" 200 1267
      MmTaUg00pdA00001fvkwsM4000 http://www.unipi.it MSIE+6.0 ...
```

Here, `213.213.31.41` is the IP address of the client (the computer originating the request), `15/Apr/2000:04:00:04` is the date/time of the transaction, `GET` is the *method* of the transaction (we consider requests concerning `GET` and `POST`), `http://www.unipi.it/images/h/h_home.gif` is the URL requested by the client, `HTTP/1.1` is the HTTP protocol, `200` is the HTTP return code (`200` means OK), `1267` is the size in bytes of the response sent to the client, `MmTaUg00pdA00001fvkwsM4000` is the cookie at the client browser, `http://www.unipi.it` is the referrer, i.e. the URL referring to the requested one, and `MSIE+6.0` is the client environment specification provided by the client browser. The number of HTTP transactions per day of a medium-large web server is in the order of millions of requests. However, we are only interested in requests of `.html` files, which can be assigned a semantic meaning, as described in the last section. The `vivacity.it` dataset records requests for a period of 5 months for a total of more than 7 millions of requests. Peaks are distributed over working days.

HTTP transactions are preprocessed by hash coding URLs, IPs, client environment and cookies, removing web robots requests and system requests (i.e., "pings" from monitoring modules, batch requests and administration requests), loading them into a relational database, and computing the following derived attributes:

- `Last` and `Next` count how long in the past and in the future the same URL is requested. The count is absolute, i.e. it considers the number of requests. `Last` (resp., `Next`) is set to a large constant ∞ if there is no previous (resp., further) request.
- `PageDelay` is the distance in milliseconds of a request from the previous request of the same user. `PageDelay` is set to a large constant ∞ in case of the very first request of an user.
- `SessionID` is a unique identifier of the user session the request belongs to.

It is worth discussing on the notions of *user* and *user session*.

We cannot use the login name as an identifier for users. In fact, the login name is known only for registered users. As a first heuristics, we could use cookies as user identifiers. On the one hand, this is an approximation: once a user deletes her cookies, she will appear as a distinct user. On the other hand, web browsers can be set not to accept cookies. The former approximation cannot be refined. For the latter, one could consider the pair (cookie, IP) in case the cookie is not present. Considering only the cookie leads to have a single user ("those who do no accept cookies") with more than 20% of all requests. Nevertheless, the pair

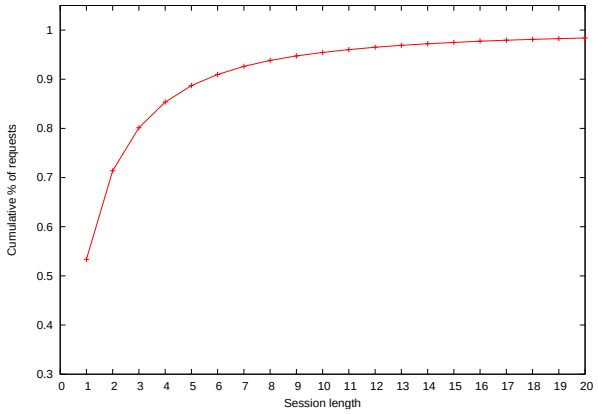

Fig. 1. Distribution of requests over session length

(cookie, IP) can only track a single "visit" (what we are going to call a *user session*) of a user and cannot be used to track users along two or more visits (IP can be reassigned, users can use different computers at home and at work, etc.).

Consider now the concept of user session. From a semantical point of view, a user session could be defined as the set of URLs accessed by a user for a particular purpose. We do not try to infer (or guess) the "purpose" in mind of the user, but rather we consider some heuristics [2]. Here, we adopt the *reference length* approach, which is based on the assumption that the amount of time a user spends examining an object is related to the interest of the user for the object contents. On this basis, a model for user sessions is obtained by distinguishing the *navigational* objects (i.e., containing only links interesting to the user) from the *content* objects (i.e., containing the information the user was looking for). The distinction between navigational and content accesses is related to the distance (in time) between a request and the next one. If between two accesses `A` and `B` there is a time delay greater than a given threshold, then `A` can be considered as a *content* URL; otherwise, it is a *navigational* URL.

On this basis, a user session is a sequence of navigational URLs followed by one content URL. Starting from `PageDelay`, we assign a `SessionID` to each URL request by grouping requests whose distance is below a given threshold. Figure 1 reports the distribution of requests over session length, where a threshold of 30 seconds is considered in computing sessions.

As a final consideration, it is well-known that the distribution of web accesses over URL typically follows the Zipf's law, which states that $f = r^{-\alpha}$ where f is the number of requests to an URL, r is the URL rank (i.e., 1 for the most accessed URL, 2 for the second most accessed one, and so on) and α is a parameter. Figure 2 (left) shows on the X axis the URLs rank and on the Y axis the number of requests to that URL. The plot is exponential on both

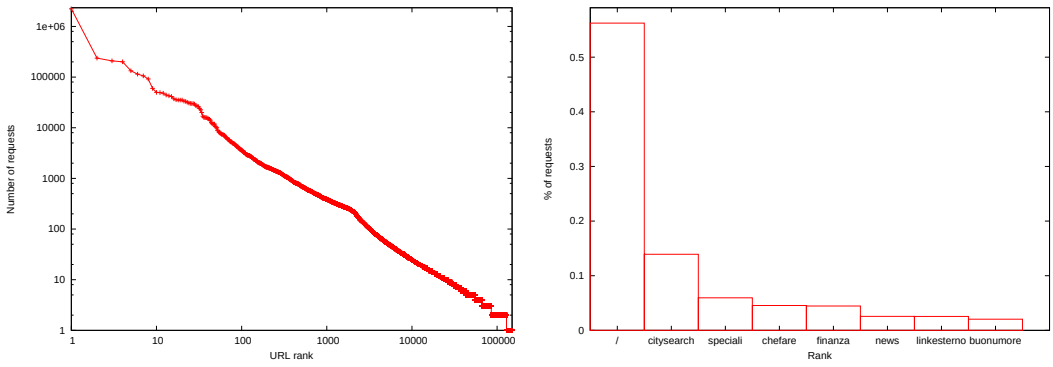

Fig. 2. Distribution of requests over URL rank and over first-level semantic rank

the X and Y axis. The straight line confirms the Zipf's law, since (by passing to logarithms) $log\ f = -\alpha log\ r$. A law similar to Zipf's can be expected at *semantic level*. Figure 2 (right) shows the distribution of requests over first-level semantic of accessed URLs.

3 Classification Experiments

In the `ClickWorld` project, we are conducting experiments to infer knowledge from access logs in order to provide web site users a proactive and personalized view. In this section, we describe two sets of classification experiments.

The first one tries to predict the sex of a user based on the web pages visited. We aim at extracting a classification model able to discriminate sex based on the navigational behavior of users. The knowledge of the sex allows to target banners, promotions, news, or to give relevance to specific sections of the web site. Similarly, the age of a user could also be relevant to be predicted, but the limited results on sex inference prevented us from exploring age inference.

The second set of experiments tries to predict whether a user might be interested in visiting a section of the web site based on the sections the user has already visited. The knowledge of the sections that could be of interest to the user allows for on-the-fly menu or page reorganization, thus giving a personalized view of the web site.

3.1 Predicting Sex from Access Logs

Only a small subset (about 15%) of the users in our dataset has registration information. It is then appealing to be able to build from those users a classification model that predicts sex of non-registered users. In this way, the web site could provide personalization features without forcing registration (for instance, this would attract users who would give up if asked for registration).

Model Extraction Process A classification algorithm takes as input a *training set*, which is a set of *cases*, or *tuples* in the database terminology. Each case specifies values for a collection of *attributes* and for a *class*. The output of the classification algorithm is a model that describes/predicts the class value of a case on the basis of the values of the attributes of the case. The predictive accuracy of the extracted model is evaluated on a *test set*, for which the actual class is known.

In our context, the set of registered users is split into training and test set (67% and 33% respectively).The attributes of a case consist of the site pages or sections visited by the user and the class consists of the user sex. Actually, assuming an attribute for each site page is not feasible (medium-large web site have hundreds of thousands of pages). As a first alternative, we consider the top most accessed pages. As a second alternative, we consider sections, i.e. we aggregate site pages according to their first-level ontology (or at deeper levels). This way, we have 24 attributes and a class. Each attribute value may be binary (i.e., sport pages have been visited or not) or numeric (i.e., sport pages have been visited n times) or discrete (i.e., sport pages have been visited zero, a few or many times).

Also, a case may be seen at two levels of granularities. The first level is to have a case for each user session, i.e. each case considers the sections visited in a single user session. The second level is to consider a case for recent past sessions of a user, i.e. each case considers the sections visited in the recent past by a user. Data for the second level can be obtained for registered users only, since for non-registered ones we cannot link two sessions of a same user (this happens when the user deletes her cookies). Since we would like to apply the model to predict the sex of non-registered users, we restrict to consider the first level of granularity.

To sum up, we have conducted a large number of experiments by varying all the factors above in the definition of the data set. Also, we have used a few classification models, including decision trees [14], instance-based models [1], classification by association[11] and bagging and boosting meta-classifiers [7]. Decision trees built with the C4.5 algorithm performed the best over almost all the experiments.

Experimental results The term of comparison in evaluating the experimental results is the *majority* classifier, i.e. the one that always predicts the most frequent class (which in our data is *man*). The error of the majority classifier over the 11614 user sessions of the test set is 45.2%, or, otherwise, its classification accuracy is 54.8%.

As a first experiment, we consider an attribute for each first-level semantics. Attribute values are *yes* or *no* depending whether at least one page belonging to the semantics has been visited in a user session. The C4.5 decision tree classifier yields a misclassification error of 40.7%, while the IBk instance based classifier yields 44.1%. Such numbers cannot be considered satisfactory.

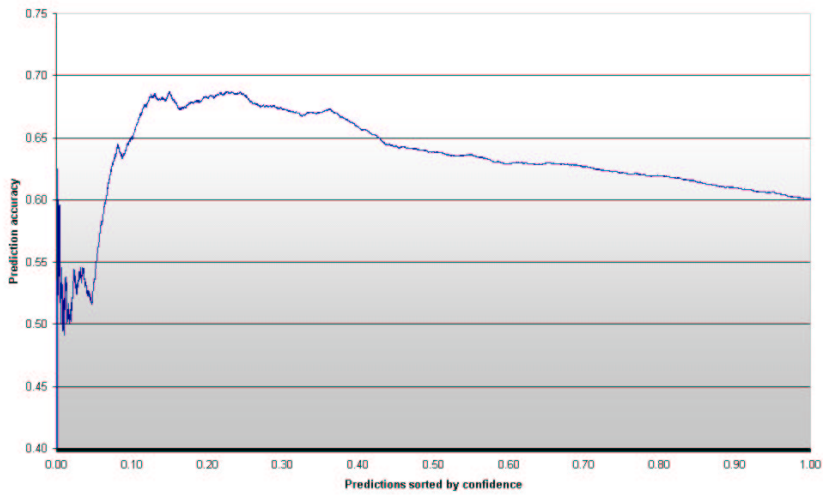

Fig. 3. Accuracy of predictions ordered by prediction confidence

In order to improve accuracy, we tried to provide more informative data, either by adding more attributes (i.e., considering first and second-level semantics) or by considering numeric attribute values (i.e., the number of pages of a given semantics visited in a user session). The latter yielded a higher misclassification rate, while the former slightly reduced it (39.8% for C4.5, and 41.4% for IBk). We also tried to control classifier induction parameters, such as pruning, bagging, boosting and n-fold cross validation, but with no significative improvement.

However, rather than simply looking at the misclassification error, let us consider a kind of lift report, which is shown in Figure 3. The plot is obtained as follows. Consider the predictions of the C4.5 decision tree ordered by prediction confidence. This is the X axis. Consider now for the first n predictions, the percentage of these that are correct. This is the Y axis. For any n, the majority classifier yields 54.8% in the Y axis! (assuming uniform distribution of cases over the X axis) On the contrary, our classifier yield as much as 65% for the top 10% predictions and as much as 68% for the top 20% predictions. As an application, the classifier yields a lift of 10.2% (resp., 13.2%) over majority classifier when selecting 10% (resp., 20%) of users to whom offer a sex-targeted service, banner or champaign.

To sum up, the small improvement over the majority classifier can be attributed to several reasons. The first one concerns the discriminative power of adopted ontology, i.e. it is too generic for distinguishing pages typically accessed by men from those typically accessed by women. The second one concerns the web site structure and contents. Typical user sessions consists of very few clicks (see Figure 1), and, perhaps, contents of the web site do not provide distinguished

access patterns based on the sex of users. Finally, we do not exploit *content mining* techniques, i.e. we only consider an *apriori* ontology, while looking into the contents of accessed pages may reveal topics that discriminate between demographic attributes. This is the approach taken in [12], where, however, a lift of 8% over the majority classifier in the top 10% predictions is reported.

3.2 Predicting Interests of Users

The `vivacity.it` web site includes sections (or *channels*) about news, sports, shopping, finance, jokes, restaurants and so on. Channels correspond approximatively to the second-level semantics attached to an URL. In this section, we consider the problem of classifying the user sessions that include an access to a channel based on the accesses to the other channels. Such a model can be used to predict potential interest of a new user in a channel she has not visited yet, based on the channels she visited in last or current session. As a consequence, the web site could highlight the channel suggested by the model thus offering a proactive presentation of contents to new users. The same scenario applies when we are going to introduce a *new* channel. Based on the user sessions that autonomously access the new channel, the model could suggest which (old and new) users may be interested in the new channel as well.

Model Extraction Process Since we do not need to restrict to registered users, cases now cover a generic user session and are available in a larger number (more than 215.000 sessions). Fixed a channel as the class, attributes of a case consist of the remaining channels. As in Section 3.1, each attribute value may be binary, numeric or discrete. The class value may be binary (channel accessed or not) or discrete (channel accessed zero, a few or many times). To sum up, we have conducted a large number of experiments by varying all the factors above in the definition of the data set.

It is worth noting that the misclassification error of extracted models cannot be considered as a good quality measure for our purposes. Let $C = \{yes, no\}$ the binary class value. Misclassification error considers those *yes* predictions that are actually *no* and those *no* predictions that are actually *yes*. However, rather than not making errors in *no* predictions, we are more concerned with correct *yes* predictions, i.e. in capturing all interested users and only them! Let RET be the set of cases (from here on, cases in the test set) that have been predicted as *yes*, REL the set of cases that have class *yes*. By borrowing the terminology of information retrieval, we define:

- *recall* as $\#(RET \cap REL)/\#REL$, i.e. the rate of retrieved (by the classification model!) interested users over the total number of interested ones,
- *precision* as $\#(RET \cap REL)/\#RET$, i.e. the rate of retrieved interested users over the total number of retrieved ones.

Recall measures how many users interested in a channel can be predict as such by the classification model. Precision measures how many predictions are cor-

rect. As one could expect, maximizing both recall and precision are contrasting objectives.

Experimental Results Since user sessions are short, most of the channels appear in a small fraction of the sessions (typically $< 10\%$). Therefore, the majority classifier would always answer *no*. This means a recall of 0%. In other words, the majority classifier, yet having a good misclassification error, is of little help in capturing potential visitors of a channel.

In Table 1, we report for some of the channels of the web site the recall, the precision and the misclassification error of our best C4.5 decision tree model. Also, we report the misclassification error of the majority classifier. The recall rate is satisfactory, with most of the time nearly the 50%, i.e. in making our predictions we reach 50% of potentially interested users. The precision rate is excellent, i.e. we make very few wrong predictions. Also, the overall misclassification error is typically a half of the misclassification error of the majority classifier.

Channel	**Recall**	**Precision**	**Misc. error**	**Maj. misc. error**
Finance	44.3%	98.27%	3.31%	5.87%
Health	52.3%	89.66%	2.06%	3.69%
Market	49.1%	83.34%	2.20%	4.05%
News	44.1%	89.27%	10.57%	17.28%
Shopping	31.5%	91.31%	5.33%	7.41%
Specials	60.2%	92.86%	11.56%	26.01%
Sport	50.0%	91.93%	1.65%	2.85%
Surveys	21.9%	92.66%	3.92%	4.91%
Theatre	54.8%	94.63%	1.20%	2.48%

Table 1. Performance of a C4.5 decision tree on the test set

4 Conclusions

We described the web usage mining activities of an on-going project, `ClickWorld`, that aims at extracting models of the navigational behavior of a web site users. Preprocessing is a critical step, where domain knowledge must be necessarily deployed, e.g. in providing an ontology of web page semantics. We have conducted experiments that build classification models for inferring sex and interests of users based on their navigational behavior. The former models yields a small improvement over random choice, probably due to scarce sex discrimination of access paths in our dataset. Nevertheless, we slightly improve a result reported in the literature. We plan to investigate whether a content mining approach could improve predictive accuracy. The latter models show an excellent precison and a satisfactory recall. All the models do not consider the order of accesses, i.e. user sessions are *sets* and not *sequences*. We are currently considering the issue by investigation sequential patterns, which take into account the order of accesses.

References

1. D.W. Aha, D. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.
2. B. Berendt, B. Mobasher, M. Spiliopoulou, and M. Nakagawa. A framework for the evaluation of session reconstruction heuristics in web usage analysis. *INFORMS Journal of Computing*, 15(2), 2003.
3. B. Berendt and M. Spiliopolou. Analysis of navigation behaviour in web sites integrating multiple information systems. *VLDB Journal*, 9(1):56–75, 2000.
4. R. Cooley, M. Deshpande, J. Srivastava, and P.N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations*, 1(2), January 2000.
5. H. Dai and B. Mobasher. A road map to more effective web personalization: Integrating domain knowledge with web usage mining. In *Proceedings of the International Conference on Internet Computing 2003 (IC03)*, 2003.
6. M. Eirinaki and M. Vazirgiannis. Web mining for web personalization. *ACM Transactions on Internet Technology (TOIT)*, 3(1):1–27, 2003.
7. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Mateo, CA, 2000.
8. K.P. Joshi, A. Joshi, Y. Yesha, and R. Krishnapuram. Warehousing and mining web logs. In *In Proc. of ACM CIKM Workshop on Web Information and Data Management (WIDM'99)*, pages 63–68. ACM, 1999.
9. KDnuggets. Software for web mining. `http://www.kdnuggets.com/-software/web.html`.
10. R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD Esplorations*, 2(1):1–15, 2000.
11. W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *IEEE International Conference on Data Mining*, pages 369–376, 2001.
12. D. Murray and K. Durrell. Inferring demographic attributes of anonymous internet users. In *Web Usage Analysis and User Profiling Workshop*, volume 1836 of *Lecture Notes in Computer Science*, pages 7–20. Springer, 2000.
13. C. Pohle and M. Spiliopoulou. Building and exploiting ad hoc concept hierarchies for web log analysis. In *Proceedings of DaWaK 2002*, volume 2454 of *LNCS*, pages 83–93, 2002.
14. S. Ruggieri. Efficient C4.5. *IEEE Transactions on Knowledge and Data Engineering*, 14:438–444, 2002.
15. M. Spiliopoulou and L.C. Faulstich. WUM: a Web Utilization Miner. In *Proceedings of the EDBT Workshop WebDB98*, volume 1590 of *LNCS*, pages 109–115, 1998.
16. M. Sweiger, M.R. Madsen, J. Langston, and H. Lombard. *Clickstream Data Warehousing*. John Wiley & Sons, 2002.