1 AN APPLICATION OF MACHINE LEARNING FOR INTERNET USERS

Machová Kristína
Technical University of Košice, Department of Cybernetics and Artificial Intelligence
Kristina.Machova@tuke.sk

This paper presents new trends in machine learning. It contains a short survey of classic methods of machine learning. Meta-learning, Boosting and Bagging are characterized in the paper as well. The paper focuses on solving the problem of Internet users' cognitive load decrease based on machine learning methods. It presents the AWS system designed to suggest Internet pages to a user on the base of his/her model. The system also offers information about visitor models for the purpose of the server content management. The AWS system is an advisory system with off-line learning capabilities, individual adaptation, and the support of global server content adaptation.

1. INTRODUCTION

Machine learning methods can be divided into supervised (with teacher) and unsupervised (without teacher) methods. A further division can be based on the type of learning task: classification or sequential task. A great deal of these methods can be found in (Mitchell, 1997) and (Machová, 2002). We will assume, that the input of a cognitive algorithm has the form of a set of training examples. The algorithm produces a concept as its output – we expect to obtain a definition of that concept as well. This definition can be of different sorts. It means, that learning methods can use various representations of knowledge to be learned. In case of the classification task, the definition can often be in the form of a classification rule. "Then" part of this rule usually represents a class (concept), to which a new example will be classified. "If" part of the rule contains a concept definition in the form of attributes with appropriate values – conditions whose meeting will be sufficient for the classification of a new example into a given class (concept). The concept definition can be of various kinds: logical conjunction, production rule, decision tree, decision list, threshold concept, criterion table, probabilistic concept and so on. Moreover, various representations of learned concepts have in common basic learning principles, for example ordered version space, hill-climbing principle, division of example space into subspaces, control with exceptions, competitive principle, score function and reduction of the number of concept versions (Machová-Paralič, 2003).

An extension of the machine learning task is represented by the field of meta-learning. Its aim is to obtain a set of rules for determining how to select the best

cognitive algorithm for a given cognitive task. These rules could be incorporated into a knowledge-based system dedicated to meta-learning. (Bensuan, 2000) presents a method which is able to obtain these rules using the supervised learning approach (e.g. using an Instance Based Learning method). A task description is based on landmarks which can be of different types, for instance decision node, arbitrary selected node, the worst node, naïve Bayes, 1NN, elitist 1NN, and linear discriminator. A value of a landmark can be determined as an average error over example space. This given method employs also meta-attributes based on information theory (class entropy, average attribute entropy, mutation information, equivalent attribute number, etc.). A disadvantage of this approach is that it is based on a high number of experiments which enables to measure classification error for different combinations of learning algorithms and databases. Further research activities could be focused on uncovering general relationships between cognitive algorithms and cognitive tasks.

2. BAGGING AND BOOSTING

Bagging and boosting represent general methods for improving results of a selected classification machine learning algorithm. Both of them modify a set of training examples in order to obtain a sequence of classifiers which can be subsequently combined into a final classifier.

Bagging (Breiman, 1994) performs random selections of data from the training set. Based on these random selections, a learning algorithm produces a sequence of results – classifiers. It is possible to obtain a final classifier by selecting from this sequence of classifiers.

On the other hand, boosting (Schapire, 1999), (Quinlan, 1996) modifies the set of training data using a distribution of weights assigned to particular examples. When the first classifier is being induced, the weight distribution is uniform. For every subsequent iteration, the weights are modified. The weights of those examples which are not correctly classified by the classifier induced in the previous iteration step are increased. And the weights of correctly classified examples are decreased. The prediction of a final classifier is given as a weighted combination of predictions of particular basic classifiers. One of the surprising and recurring phenomena observed in experiments with boosting is that the test error of the generated classifiers usually does not increase as its size becomes very large, and often it is observed to decrease even after the training error reaches zero. This phenomenon is related to the distribution of margins of the training examples with respect to the generated voting classification rule, where the margin of an example is simply the difference between the number of correct votes and the maximum number of votes received by any incorrect label. The most known boosting algorithm is AdaBoost, which significantly reduces the error of any learning algorithm that consistently generates classifiers whose performance is a little better than random guessing. (Freud-Schapire, 1996) presents two sets of experiments. The first set compared boosting to Breiman‘s bagging method when used to aggregate various classifiers including decision trees. The performance of boosting using a nearest-neighbour classifier is studied in the second set of experiments.

Bagging and boosting can be used in any domain in which machine learning methods can be employed, for example in extracting knowledge for knowledge bases, data mining, etc. (Schapire-Singer, 2000) presents algorithms which learn from examples to perform multi-class text and speech categorisation tasks. The presented approach is based on an implementation of the boosting algorithm for text categorisation tasks, called BoosTexter. This system is applied to automatic call-type identification from unconstrained spoken customer responses.

3. AN APLICATION OF MACHINE LEARNING

3.1 Problem definition

At present, the Web represents one of the most used Internet based services. The number of accesses of various users is almost unbelievable. The Web consists of a vast number of web pages. It is not uncommon a case when a user stops its browsing through pages which seem to be uninteresting (or unattractive) for him/her although the searched information is present on these pages. Another issue is, that a user searching the Web can be currently interested in some other information than during his/her previous visits. Moreover, a huge number of links were accumulated among web pages. The basic feature of the Web – hyper-textual links representing relationships among pages – can be a source of difficulties (turning to a real nightmare) when browsing the Web. These problems related to information search and retrieval can be measured by a user cognitive load.

The problem is addressed by the AWS system striving for decreasing the cognitive load of Internet users. The focus of the system is on supporting an adaptive web. The adaptive web is able to adapt itself to its visitors – the adaptation is based on an observation of users' activities (the behaviour of users) during users' visits of the Web.

The AWS system focuses on the development of user models from users' requirements. Such model type can be used to customise the response to a user requirement – the user is provided only with those documents which are relevant to his/her profile (i.e. his/her model). User models are constructed using heuristic machine learning methods. The learning is based on logs of web servers. The AWS system represents an advisory system with off-line learning, individual adaptation (customisation for each particular user based on his/her individual model), and the support for global server adaptation (transforming pages into the form suitable for majority of visitors).

3.2 Used methods

The system employs two methods for heuristic search of concept space (namely HGS and HSG) which belong to supervised methods of machine learning and are applicable for solving the classification task. In addition, a clustering method (CLUSTER/2) belonging to unsupervised learning methods was used by this system.

Machine learning is generally based on a set of training examples and achieved results are tested using a set of test examples. Training and test examples constitute

a set of typical examples. The typical examples are represented as a set of *n* attributes with their values. The last attribute can represent (in case of supervised learning) a class to which the given example belongs.

A set of typical examples is the most often given in the form of a table. An example is given in Table 1 presenting typical examples in the form used by the AWS system. This table contains typical examples characterised by an attribute *A* and belonging to a class *T*. The examples represent accesses to server pages. The attribute *A* (url) characterises those pages which were accessed (each page is stored on the server together with a set of key words which characterise the content of the page). The attribute *T* (user ID) identifies users who accessed the given pages and in this way it specifies the class to which the given accesses belong. It is quite common, that several users visit the same page – and the same typical example is classified into more than one class at the same time.

Table 1. A set of typical examples obtained from a log of a www server

Number	A (url)	T (userID)
1	/som.php	USER3eafc6cd8c98a
2	/maxnet.php	USER3eafc6cd8c98a
3	/ns_top.php	USER3eafc6cd8c98a
4	/cobweb.php	USER3eafc71274ad9
5	/id3.php	USER3eafc71274ad9
6	/c45.php	USER3eafc71274ad9
7	/pid1.php	USER3eafc7413857e
8	/psd1.php	USER3eafc7413857e
9	/plc.php	USER3eafc7413857e

Classification represents the decision on a class of a new example (with unknown class) based on definitions of available classes which were constructed using some machine learning method.

The AWS system relies on using HGS (Heuristic General to Specific) and HSG (Heuristic Specific to General) methods (Michalski, 1980) and (Machová, 2002). Both methods differ from exhaustive search of concept space – they do not search all concept space but the most promising hypotheses only. How promising particular hypotheses are can be calculated by a score heuristic function. Each algorithm iteration considers only a limited number of hypotheses (with the highest score) – this number is defined as Beam Size (BS).

Both methods (HGS and HSG) use the principle of limiting the concept space to be searched. They differ in the used direction of search. The HGS algorithm searches the concept space from more general concept descriptions to more specific (GS search direction). On the other hand, the HSG algorithm searches the concept space from more specific concept descriptions to more general ones (SG search direction).

Clustering methods can be applied when training examples do not contain any information about the class they belong to. In this case they can be grouped into natural groups or clusters using techniques of unsupervised learning (there is no feedback in the form of a class defined in advance). The clustering process starts with a set of objects – training examples. The aim is to create a set of clusters and all

available training examples to distribute over the set of clusters. In general, it is possible to distinguish several different approaches to clustering: iterative, conceptual, hierarchical, and probabilistic. The AWS system employs the CLUSTER/2 clustering method (Michalski, 1983).

3.3 The AWS system

The AWS system (Machová-Klimko, 2004) was designed with the aim to enable suggestions of pages to a user based on his/her model and to carry out an individual adaptation of the content of a server. The user model is generated as a result of the heuristic search of concept space using the HGS and HSG algorithms.

At the same time, the system provides information about models of server visitors and their interests in order to support server content management. In this way it contributes to customising the server to users – it supports a global server adaptation. This feature of the system is backed up by the CLUSTER/2 clustering technique.

As depicted in Figure 1, the system consists of two parts: on-line and off-line. The on-line part is responsible for the identification of visitors and subsequent generation of suggestions based on visitors' models. The off-line part of the system is responsible for development of user models and providing information vital for the adaptation of the server content. The learning itself is performed utilising information about the content of a server and a log of the given server. The application requires the server log in the NCSA Combined Log format.

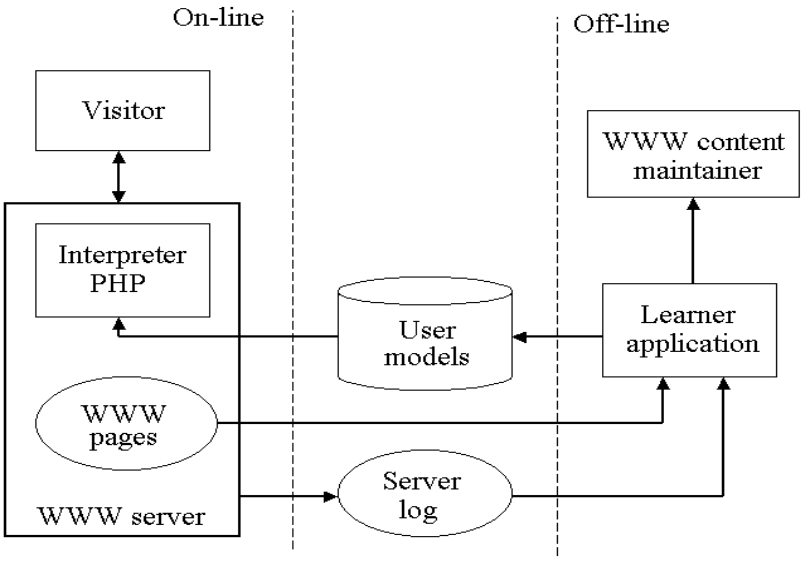

Figure 1: Structure of the AWS system

The shared part of the system is represented by a database storing user models and server logs with information about processed user requirements. The AWS system enables to identify a visitor using his/her IP address or using cookies. The identification using cookies seems to be more suitable.

The on-line part consists of a www server, PHP interpreter, and a database of user models. A visitor sends his/her requirements on the www server. The server delivers these requirements to the PHP interpreter. The interpreter identifies the visitor and generates a query into the database. Using this query the interpreter retrieves addresses and names of pages to be suggested (based on the model of the identified visitor) and sends this suggestion to the visitor together with the page which was required by the visitor. If cookies are used to identify visitors, then in case that the visitor cannot be identified (e.g. because the user accesses the server for the first time or he/she deleted cookies in his/her web browser) a new unique identifier is generated. This identifier is sent to the client and stored on his/her disc in the form of a cookie for a subsequent identification.

The off-line part of the system represents a system mainstay. It consists of the ASW/Learner application. This application is responsible for learning (creation of user models using heuristic algorithms), populating the developed user models with relevant web pages, and for the support of the server content management based on clustering of the user models using a clustering algorithm.

3.5 A business application

The AWS system can be used in the field of advertisement as well. The system generates models of users from accesses of these users to particular web pages. These models reflect (not only) professional interests of the users. Based on this information, it is possible to recommend some product to those users who can be potentially interested in it.

For example, if a user frequently visits pages devoted to a particular software product, the AWS system can recommend him additional software packages or new versions of the product. Another example is represented by the case when the user(s) represent(s) a company. The model of this user/these users generated by the AWS system can play the role of a company profile. In consequence, new apparatus, products, or technological methods and procedures can be offered to the company representatives.

3.6 Conclusion

The paper focuses on machine learning and some of it's new trends, specifically meta-learning, bagging and boosting. This contribution presents also an application possibility of mentioned fields, namely solving the problem of cognitive load of Internet users by means of machine learning. A description of the AWS system is presented – the system which was designed as an advisory system with off-line learning capabilities, possibility of an individual adaptation and with the support for a global content adaptation.

The presented approach can be further extended using a method for automatic extraction of key words from documents in order to replace the manual web page

description/annotation, for example using the method presented in (Paralič-Bednár, 2003).

The work presented in the paper was supported by the Slovak Grant Agency of Ministry of Education and Academy of Science of the Slovak Republic within the 1/1060/04 project ”Document classification and annotation for the Semantic web”.

4. REFERENCES

1. Bensusan, H., Giraud-Carrier, C.: Casa Batlo is in Passeig de Gracia or how landmark performances can describe tasks. http://www.metal-kdd.org/, 2000.
2. Breiman, L.: Bagging predictors. Technical Report 421, Department of Statistics, University of California at Berkeley, 1994.
3. Freund, Y., Schapire, R.E.: Experiments with a New Boosting Algorithm. Machine Learning: Proc. of the Thirteenth International Conference, 1996.
4. Machová, K.: *Machine learning. Principles and algorithms* (in Slovak), ELFA s.r.o., 2002, Košice. ISBN 80-89066-51-8.
5. Machová, K., Klimko, I.: Support of the adaptive WEB by means of machine learning. Proc. of the conf. Znalosti 2004, Brno, Czech republic, 2004, VSB-Technical University Ostrava, 218-225, ISBN 80-248-0456-5.
6. Machová, K., Paralič,J.: Basic Principles of Cognitive Algorithms Design. Proc. of the IEEE International Conference Computational Cybernetics, Siófok, Hungary, 2003, 245-247 ISBN 963 7154 175.
7. Michalski, R.S.: Pattern Recognition as Rule-guided Inductive Inference. *IEEE Trans. on Pattern Analysis and Machine Intelligence, 2*, 1980, 349-361.
8. Michalski, R.S., Stepp, R.: Automated Construction of Classification: Conceptual Clustering versus Numerical Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence, No.5,* 1983, 219-243.
9. Mitchell, T.M.: Machine Learning. The McGraw-Hill Companies, Inc. New York, 1997, 414.
10. Paralič, J., Bednár, P.: Text Mining for Documents Annotation and Ontology Support. A book chapter in: “Intelligent Systems at the Service of Mankind”, Volume 1, Ubooks, 2003, 237-247, ISBN 3-935798-25-3.
11. Quinlan, J.: Bagging, boosting, and C4.5. In Proc. AAAI 96, Portland, OR, 1996, 725-730.
12. Schapire, R.E., Singer, Y.: Improved Boosting Algorithms Using Confidence-rated Predictions. Machine Learning, 37(3), 1999, 297-336.
13. Schapire, R.E., Singer, Y.: BoostTexter: A Boosting-based System for Text Categorization. Machine Learning, 39(2/3), 2000, 135-168.