Meta-learning approach for automatic parameter tuning: A case study with educational datasets

M.M. Molina, J.M. Luna, C.Romero, S.Ventura
Department of Computer Science
University of Cordoba, Spain
i02mamom@uco.es, i32luarj@uco.es, cromero@uco.es, sventura@uco.es

ABSTRACT

This paper proposes to the use of a meta-learning approach for automatic parameter tuning of a well-known decision tree algorithm by using past information about algorithm executions. Fourteen educational datasets were analysed using various combinations of parameter values to examine the effects of the parameter values on accuracy classification. Then, the new meta-dataset was used to predict the classification accuracy on the basis of the value parameters and some characteristics of the dataset. The obtained classification models can help us decide how the default parameters should be tuned in order to increase the accuracy of the classifier when using different types of educational datasets.

Keywords

parameter tuning, classification, J48 algorithm

1. INTRODUCTION

One of the objectives of Educational Data Mining (EDM) [10] must be to design easy-to-use tools and algorithms for educators and non-expert users of data mining. Traditional data mining tools, such as Weka, Rapid-Miner, Clementine, DB-Miner, etc., are normally designed more for power and flexibility than for simplicity. Therefore, these tools can be complex, with features well beyond the scope of an educator's needs. Most current data mining algorithms used by these tools need to be configured before they are executed. In other words, users have to provide appropriate values for the parameters in advance in order to obtain good results or models; therefore, the user must possess a certain amount of expertise in order to find the right settings. To resolve this problem, data mining can be used to learn from past executions of the algorithms in order to improve the future selection of parameters according to the past behaviour of the algorithm.

In this paper, we propose a meta-learning approach for tuning parameters. Meta-learning is the study of principled methods that exploit meta-knowledge to obtain efficient models and solutions by adapting machine learning and the data mining process [1]. In our case study, we used a meta-learning approach to support the user in tuning the parameter values of a decision tree classification model when using different types of educational datasets. The decision tree model has some parameters that influence the amount of pruning. By trimming trees, the computational efficiency and classification accuracy of the model can be optimised. As a case study, we used a set of educational datasets and the J48 [9] (improved version of the C4.5 classification algorithm) to predict a discrete variable or class (accuracy variations) based on the values of the parameters and some features of the datasets. We executed some combinations of parameter values to examine their effects on a classification quality metric.

This paper is organised as follows: Section 2 provides background information from related works on applying data mining for parameter tuning; Section 3 describes the methodology used in this work; Section 4 includes the list of educational datasets used as a case study; Section 5 describes the experiments, results, and model obtained; and finally, conclusions and future works are outlined in Section 6.

2. BACKGROUND

In data mining, it is generally necessary to set the parameters used by the algorithm in order to achieve the best possible model and results [7]. Experiments show a substantial increase in accuracy when the right parameters are used. However, there is an associated problem in adjusting the parameters of most data mining algorithms. This task may involve a high computational cost for finding the optimal parameters or else risk relying on assumptions that may bias the results. Achieving optimal parameters automatically is not an easy task, therefore, and it often requires help from an expert. Some possible solutions include providing default values to the user (the most simple and common solution), reducing the number of parameters, tuning parameters automatically (the chosen option in this paper), and developing parameter-free data mining [6] algorithms (the ideal but most difficult solution).

The area of automatic parameter tuning research has gained much interest in recent years [13]. The definition of automatic parameter tuning used in this paper is to automatically find parameter settings that are better than the defaults. Different methods and techniques have been proposed for automatic parameter tuning [2], such as optimisation techniques (racing algorithms, local search, experimental design, etc.), machine learning and/or data mining. In fact, classifiers have been used to learn the values of parameters needed to set the configuration. Maimon, Rockach, and Edel [7] describe a classification model for meta-based parameter tuning. Srivastava and Mediratta [11] suggest the use of decision trees for automatic tuning of search algorithms. Pavon, Diaz, Laza, and Luzon [8], have automated the parameter tuning process through classification of previous runs of the algorithms. Dakovski and Shevked [3] consider an algorithm for learning from examples from the view point of improving classification accuracy by determining influencing parameters and optimal values.

This paper focuses on automatic parameter tuning by supporting the selection of the parameter values of a J48 classifier. The obtained model can help us make decisions about how we can tune the default parameters to increase the accuracy of the classification when using different types of educational datasets.

3. METHODOLOGY

We propose a methodology that uses a meta-learning approach to support the selection of parameter values for the algorithms (see Figure 1).

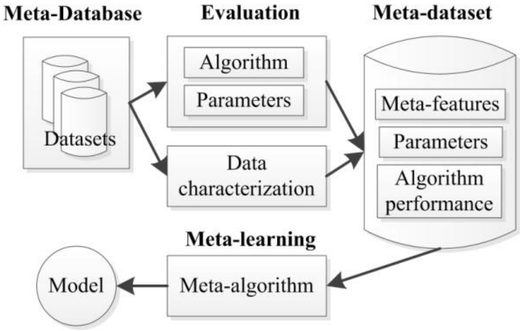


Figure 1. Meta-learning approach

In our meta-learning approach (see Figure 1), the meta-database consists of educational datasets. Then, we defined properties that are important for characterising datasets and developing meta-features (the number of instances, attributes, and classes). We selected a base algorithm, and parameters, to evaluate its performance. In this case, we selected the J48 algorithm and two parameters (*confidenceFactor* and *minNumObj*) to obtain the meta-dataset with meta-features, parameters, and performance (classification accuracy). Finally, meta-learning (a meta-algorithm) was applied to the previous meta-dataset in order to obtain a classification model for predicting whether an increase or decrease in estimated accuracy is to be expected for a given record. Each record of the meta-dataset represents a type of dataset and a certain parameter setting.

4. DATASETS

We used a set of 14 educational datasets based on the traditional classification problem for predicting students' final performance [10]. These datasets (see Table 1) contain as input attributes a variety of information about students and as classes (the output attribute to predict) the categorical final marks obtained by students in different types of courses:

- Moodle 1 to 7: Data about first, second, and third–year students for a degree in computer science at Cordoba University during the years 2007–2010, obtained from Moodle (accesses, assignments, and activities in questionnaires, forums, etc.)
- Higher 1 and 2: Data about first–year Cordoba students for a degree in computer science during 2010, obtained from several sources (admission and progress in subjects, Moodle, and a survey)
- Secondary 1 to 5: Data about students of secondary education in Zacatecas, Mexico, during 2010, obtained from several sources (admission information, scores in subjects, and a specific survey)

Table 1 shows the list of educational datasets and three features of these datasets: the number of attributes (Nattributes), the number of instances (Ninstances) and the number of classes (Nclasses). Clearly, there is a wide range of values in the features of each dataset. In fact, there are datasets with a low, medium, or high number of attributes, instances, or classes.

Dataset	Nattributes	Ninstances	Nclasses
Moodle1	4	1000	5
Moodle2	10	103	3
Moodle3	41	103	3
Moodle4	6	2708	3
Moodle5	6	9554	3
Moodle6	10	438	4
Moodle7	10	438	2
Higher1	24	88	6
Higher2	24	88	2
Secondary1	77	670	2
Secondary2	14	670	2
Secondary3	60	419	2
Secondary4	17	386	2
Secondary5	53	419	3

Table 1. Features of the educational datasets

5. EXPERIMENTS

Experiments were conducted to predict how to increase or decrease the accuracy of a well-known classification algorithm, depending on the parameters used and the features of the educational datasets used, using past information about algorithm executions. The decision tree learner selected was J48, which has several parameters but only two of which influence the amount of pruning [12]:

- ***confidenceFactor*** is the confidence factor for pruning, and it influences the size and predictability of the tree constructed. For each pruning operation, it defines the probability of error in the hypothesis that deterioration due to this operation is significant. The default value is 0.25. The lower this value, the more pruning operations allowed.
- ***minNumObj*** is the minimum number of instances per leaf. The default value is 2.

We executed the algorithms using different settings and stored the accuracy obtained in each execution as part of the meta-database. In fact, J48 was executed several times for each dataset by modifying these parameters into a range (in a similar way that an optimiser works). Each setting was evaluated using 10-fold cross-validation, and the accuracy (rate of correctly classified instances) obtained from test data was stored. The settings used were: *confidenceFactor* (0.1, 0.25, and 0.5) and *minNumObj* (1, 2, and 10), that is, a total of nine different combinations of parameters for each dataset. Next, in order to have a classification problem (that is, a class), we transformed the continuous value (float) of the obtained accuracy to a discrete or categorical value (label) in the following way:

- The accuracy value obtained when using the two default parameters together (0.25 and 2) was used as a control value; therefore, it was not discretised and was not used later for predicting (only the remaining eight executions).
- All the other accuracy values obtained were used as experimental values and transformed to the labels Equal, Increase, Decrease, Increase+, and Decrease− depending on

the variation of accuracy with respect to the control accuracy. In other words, each value was compared with the accuracy obtained using the default settings, and the label describes the difference: no difference (Equal), a higher or lower accuracy (Increase or Decrease, respectively), a much higher or lower accuracy (Increase+ or Decrease− respectively).

Finally, all the previous information was stored in a meta-dataset with 112 instances/examples and six attributes (five numerical attributes (three meta-features and two parameters) and one class (accuracy variation)). However, in order to create a different version of the same meta-dataset, we discretised all the numerical values. The labels used by *ConfidenceFactor* are LOWER to 0.1, DEFAULT to 0.25, and HIGHER to 0.5. The labels used by *MinNumObj* are LOWER to 1, DEFAULT to 2, and HIGHER to 10. The labels used by Nattributes, Ninstances, and Nclasses are shown in Table 2.

	LOW	**MEDIUM**	**HIGH**
Nattributes	≤10	>10 AND ≤30	≥ 30
Ninstances	≤ 100	>100 AND ≤1000	> 1000
Nclasses	= 2	>2 AND ≤ 4	> 4

Table 2. Discretisation of the meta-features

Based on the two previous meta-datasets, meta-learning (discrete and numerical classification) was used to predict the variation of the accuracy depending on the meta-features of the dataset and the values of the parameters. We used different types of classification algorithms provided by Weka [12]:

- Bayes-based algorithms: BayesNet, NaiveBayes
- Functions-based algorithms: Logistic, RBFNetwork, and MultilayerPerceptron
- Rules-based algorithms: JRip, NNge, PART, and Ridor
- Trees-based algorithms: LADTree, SimpleCART, REPTree,and J48

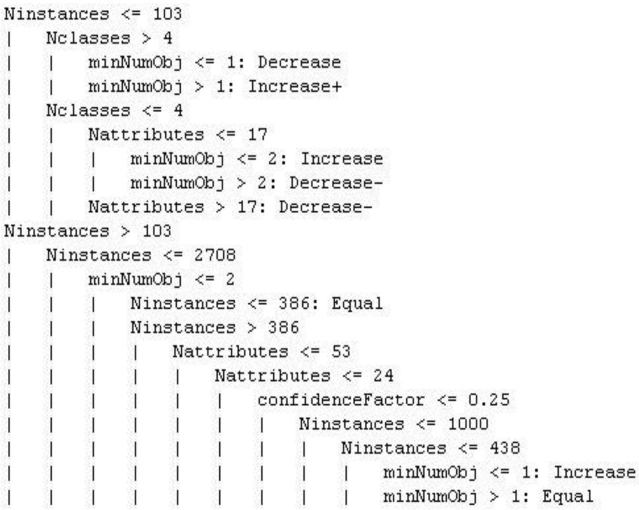
All these algorithms were executed using default parameters and 10-fold cross-validation, and their accuracy when using the original numerical attributes (A) was compared with their accuracy when using the categorical attributes (B) (see Table 5).

In general, none of the meta-learning classification algorithms obtained a very high accuracy, with values varying between 50% and 75% of correctly classified instances (see Table 3). From the results using original numerical attributes (column A) and those using categorical attributes (column B), it is apparent that all the algorithms obtained better results when using the original numerical attributes. Finally, the algorithm that obtained the highest accuracy in both cases (A and B) was the J48 classifier.

Algorithm	**(A)**	**(B)**
BayesNet	0.573	0.492
NaiveBayes	0.573	0.492
Logistic	0.617	0.573
RBFNetwork	0.617	0.537
MultilayerPerceptron	0.537	0.519
JRIP	0.573	0.528
NNge	0.671	0.492
PART	0.671	0.600
RIDOR	0.600	0.591
LADTree	0.671	0.582
SimpleCart	0.689	0.564
REPTree	0.635	0.573
J48	**0.751**	**0.698**

Table 3: Accuracy of classification algorithms

Next, we describe the two classification models obtained by the J48 algorithm. These decision trees can easily be interpreted by a human and can help in making decisions about how to tune parameter values in order to increase the accuracy of the classification when using different types of datasets. Figure 2 shows part of the J48 pruned tree obtained when using the meta-datasets with numerical attributes.

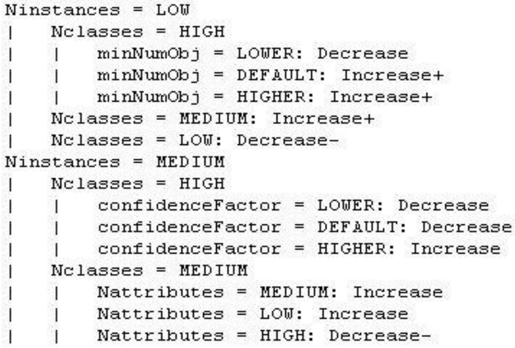


```
Ninstances <= 103
|   Nclasses > 4
|   |   minNumObj <= 1: Decrease
|   |   minNumObj > 1: Increase+
|   Nclasses <= 4
|   |   Nattributes <= 17
|   |   |   minNumObj <= 2: Increase
|   |   |   minNumObj > 2: Decrease-
|   |   Nattributes > 17: Decrease-
Ninstances > 103
|   Ninstances <= 2708
|   |   minNumObj <= 2
|   |   |   Ninstances <= 386: Equal
|   |   |   Ninstances > 386
|   |   |   |   Nattributes <= 53
|   |   |   |   |   Nattributes <= 24
|   |   |   |   |   |   confidenceFactor <= 0.25
|   |   |   |   |   |   |   Ninstances <= 1000
|   |   |   |   |   |   |   |   Ninstances <= 438
|   |   |   |   |   |   |   |   |   minNumObj <= 1: Increase
|   |   |   |   |   |   |   |   |   minNumObj > 1: Equal
```

Figure 2. Part of the decision tree using numerical attributes

As we can see, all the input attributes (the three meta-features and the two parameters) appear in the decision tree; therefore, all show a relationship with the variations of accuracy. For example, the first two rules of the tree show that if the number of instances is less than 103 and the number of classes is greater than 4, then the value of the *minNumObj* parameter can decrease the accuracy a little (for a value less than or equal to 1) or can increases it quite a lot (for a value greater than 1).

Figure 3 shows part of the J48 pruned tree obtained using the meta-datasets with discrete attributes.



```
Ninstances = LOW
|   Nclasses = HIGH
|   |   minNumObj = LOWER: Decrease
|   |   minNumObj = DEFAULT: Increase+
|   |   minNumObj = HIGHER: Increase+
|   Nclasses = MEDIUM: Increase+
|   Nclasses = LOW: Decrease-
Ninstances = MEDIUM
|   Nclasses = HIGH
|   |   confidenceFactor = LOWER: Decrease
|   |   confidenceFactor = DEFAULT: Decrease
|   |   confidenceFactor = HIGHER: Increase
|   Nclasses = MEDIUM
|   |   Nattributes = MEDIUM: Increase
|   |   Nattributes = LOW: Increase
|   |   Nattributes = HIGH: Decrease-
```

Figure 3. Part of the decision tree using categorical attributes

As we can see in Figure 3, very similar rules are obtained and, again, all the input attributes appear in the decision tree. The three first rules of the tree show that if the number of instances is low and the number of classes is high, then the value of the *minNumObj* parameter can decrease the accuracy a little (for a value lower than the default value) or can increase it quite a lot (for a value equal to or higher than the default value). In our opinion, this second decision tree is a little more comprehensible to a human for two main reasons:

1. The tree is much smaller. The first decision tree (Figure 2) has 47 nodes and 24 leaves (rules), and the second decision tree (Figure 3) has 28 nodes and 19 leaves (rules). We maintain that a small decision tree with fewer and shorter rules is more comprehensible.

2. Although the accuracy of the classification is lower when discretising (see Table 3), the use of labels instead of numbers and operators (equal, greater than, less than, etc.) provides more simple rules. We maintain that a decision tree with labels or linguistic variables is more comprehensible.

6. CONCLUSIONS

In this paper, we have shown that a meta-learning approach can be used for parameter tuning of decision tree algorithms. We used 14 educational datasets because there are no more datasets on classification tasks in education available. Although there are some public and well-known data repositories, such as the UCI machine learning repository [4] and the PSLC DataShop [5], there are no educational datasets available in UCI and the PSLC datasets are oriented to predicting student step-level performances and not to the classification problem/task of predicting final marks. The ideal would be to use a great number of educational classification datasets from different types of education systems, such as primary, secondary, higher, special education, and so on, both in traditional face-to-face and in on-line education (learning management systems, adaptive educational hypermedia systems, intelligent tutoring systems, etc.). We selected the J48 algorithm and only two of its parameters, but in the future, other well-known algorithms and a great number of parameters may be used to broaden the research on the relationship between parameters and performance (accuracy). Finally, we used only three basic characteristics of the datasets (number of instances, number of attributes, and number of classes). However, future research may use other characteristics, such as level of missing data, level of imbalance in data, level of complexity, and so on.

7. ACKNOWLEDGMENTS

This work was supported by the Regional Government of Andalusia and the Spanish Ministry of Science and Technology projects, P08-TIC-3720 and TIN-2011-22408, respectively, FEDER funds, and the Spanish Ministry of Education under the FPU grant AP2010-0041.

8. REFERENCES

[1] Brazdil, P., Giraud-Carrier, C., Soares, C. and Vilalta, R. Metalearning: Applications to Data Mining. *Series: Cognitive Technologies*. Springer, 2009.

[2] Cayci, A., Eibe, S., Menasalvas, E. and Saygin, Y. Bayesian networks to predict data mining algorithm behavior in ubiquitous computing environments. *International Workshops MSM*, 2010.

[3] Dakovski, L. and Shevked, Z. Tuning classification for prime implicants based learner. *International Conference on Computer Systems and Technologies*, 2006.

[4] Frank, A. and Asuncion, A. UCI machine learning repository. Irvine, CA: University of California, School of Information and Computer Science, 2010.

[5] Koedinger, K., Cunningham, K., Skogsholm, A. and Leber, B. An open repository and analysis tools for fine-grained, longitudinal learner data. *1st International Conference on Educational Data Mining*, 157–166, 2008.

[6] Keogh, E., Lonardi, S. and Ratanamahatana, C.A. Towards parameter-free data mining. *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 206–215, 2004.

[7] Maimon, O., Rokach, L. and Edel, I. Parameter tuning for classification algorithms in data mining using meta learning. *13th Israeli Conference of Industrial Engineering and Management*, 2004.

[8] Pavon, R., Diaz, F., Laza, R. and Luzon, V. Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study. *Expert Systems with Applications*, 36(2), 3407–3420, 2009.

[9] Quinlan, J. R. *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

[10] Romero, C. and Ventura, S. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics*, 40(6), 601–618, 2010.

[11] Srivastava, B. and Mediratta, A. Domain-dependent parameter selection of search-based algorithms compatible with user performance criteria. *Proceedings of AAAI*, 3, 1386–1391, 2005.

[12] Witten, I. H., Eibe, F. and Hall, M.A. *Data Mining: Practical Machine Learning Tools and Techniques, Third Edition*. Morgan Kaufman Publishers, 2011.

[13] Konen, W., Koch, P., Flasch, O., Bartz-Beielstein, T., Friese, M. and Naujoks, B. Tuned data mining: a benchmark study on different tuners. *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, 1995–2002, 2011.