

Decision Trees - A New Paradigm in the Educational Research

Lavinia Aurelian Bădulescu, Roxana Tîmplaru

Abstract — Data Mining defines the process of knowledge discovery from data. We propose in this paper a new paradigm of educational research based on a Data Mining technique: Decision Trees. Decision Trees algorithms are used for two Data Mining tasks: classification and prediction, tasks frequently met in educational research. Experiments presented in our paper exemplify the way Decision Trees can be used in educational research.

Keywords — Data Mining, Decision Trees, Educational Research.

I. INTRODUCTION

DATA Mining (DM) defines the process of knowledge discovery from data [1], as an automatically extraction of „hidden” predictive information from the databases, or as the process of automatically extraction of valid, useful information, previously unknown and finally comprehensible, from huge databases, to be used in crucial decisions [2]. DM represents in certain views an extension of statistics to which there were added elements of artificial intelligence and machine learning.

Aristotle (384-322 B.C.) and Francis Bacon (1561-1626) sustained a conception of scientific methodology which was used during two millenniums. They suggested the gathering of huge amounts of data, the search of models in those data and then the formulation of several hypotheses of those models. Galileo Galilei (1564-1642) continued to sustain this direction, but he affirmed that scientists must also perform experiments to check the veracity of the hypotheses. The Galilean conception upon scientifically methodology was widely accepted by the scientific community for about 300 years; it was still used during the whole 19th century.

Although in 20th century an important change has been made in the manner it was put into practice scientific methodology. Something was inversed in Galilean scientific method; it was sustained that first must be postulated the theory, then must be collected the experimental data for the sustaining of that theory. This conception of the scientific method was named confirmatory science. And the educational research uses exclusively this methodology of the confirmatory science.

L. A. Bădulescu is with the Colegiul Național „Elena Cuza”, Craiova, Romania (phone: 40/351/430352; e-mail: laviniu_aurelian_badulescu@yahoo.com).

R. Tîmplaru, is with the Inspectoratul Școlar Județean Dolj, Craiova, Romania (e-mail: roxana_t2004@yahoo.com).

All the papers about educational researches stipulate the same way to follow: a pedagogic hypothesis it is formulated and then data are collected for confirmation or improving it.

DM comes back to the scientific method of Aristotle, Bacon and Galilei in that data generates the theory. This conception of the scientific methodology which is also used by DM, it is named exploratory science [3]. It is that we propose in this paper: a new paradigm of educational research based on a very interesting DM technique, Decision Trees (DT).

II. OPENSOURCE, GNU AND GPL

For solving DM task we can choose OpenSource software from net. Some of the available DM software under the GNU/GPL license is presented below:

- Weka is a suite of machine learning / DM software. It contains Java implementation for various mining algorithms, data preprocessing filters, and experimentation capabilities (OpenSource/GNU/GPL);
- R is both a programming language and an environment for statistical computing and graphics. It provides a powerful environment for DM, with a comprehensive collection of analytic tools (GNU);
- DMTools - written in Python provides a number of routines for basic DM tasks (GNU/GPL);
- C. Borgelt [4], University of Magdeburg, Germany, makes available a collection of DM algorithms. (GNU/GPL);
- The PNC2 Rule Induction System - Windows software tool that induces rules using the PNC2 cluster algorithm (GNU/GPL);
- QuickMiner - OpenSource creation of a data mining C++ procedure library; - ARMiner is a free client server application for discovering association rules in datasets. ARMiner currently implements two algorithms: Apriori and Closure (GNU/GPL).

III. DECISION TREES

Besides other DM techniques such as: neural networks, clustering, Bayesian networks, genetic algorithms, association rules, regression analysis, fuzzy sets etc., on datasets exploring there can be used DT [5] –[6]. DT algorithms are used for two DM tasks: classification and prediction, tasks frequently met in educational research.

In classification / prediction we are given a dataset named input data, known as training dataset in which each record consists of several attributes or features. An

attribute can either be numerical (continuous) or categorical (discrete, nominal). If the values of an attribute belong to an ordered domain, the attribute is named numerical (e.g. marks, age). On the other hand, a categorical attribute can take different values in an unordered domain (e.g. subject matter, specialization, gender). One of the categorical attributes is considered as a classification attribute; its values are called class labels. Class label indicates to which class each record belongs to. The classification's objective is to analyze input data and to create an exact description or a model for each class based on existing characteristics in the data. Once created this kind of model, the future records which do not exist in the training dataset and whose class labels are unknown, can be classified based on the model [7].

Inductive generalization given in the form of DT acts through data patterns identification. This approach is valid if there can be recognized, without the risk of coincidence, a sufficient number of patterns. As this differentiation usually depends on statistical tests, there must be a sufficient number of records to allow these tests to be effective.

The DT algorithms are supervised learning methods – in which the classes to which the cases must be assigned are established in advance – which build DT from an input-output set of cases. The DT is represented by tree-shaped structures which represents sets of decisions. These decisions generate rules for datasets classification. In order to build the tree-shaped model, the original variables which haven't been transformed or normalized variables may be used. The DT model will generate rules upon data in order to estimate the target variable or class. Each subtree from the DT represents an answer to a classifying question; its leaves are partitions or segmentations of dataset in accordance to the classification and nodes present statistical information.

A DT for classification builds a model through the recursive splitting of the training dataset in partitions so that all or almost all records in a partition have the same class label. Most DT classifiers realize the classification in two phases: growing (building, induction) and pruning (see Figure 1).

In growing phase the algorithm begins with the entire set of data allocated for the first node. The set of data is partitioned in subsets accordingly with the splitting criteria. This operation is recursively repeated for each subset until each subset contains only cases belonging to the same class, or the subset is small enough. The algorithm build on the DT base generates classification or estimation models using machine learning or statistical elements. The main idea of the algorithm is to use a splitting criteria to determine the most predictive factor in order to place it on the root as the first point of decision in the tree and furthermore, to search the predictive factors so that it builds the sub-trees, until the point in which there is no data left to be processed.

BuildTree (dataset S)

if all records in S belong to the same class

return

for each attribute A_i

evaluate splits on attribute A_i

use best split found to partition S into S_1 and S_2

BuildTree (S_1)

BuildTree (S_2)

PruneTree (node t)

if t is leaf

return $C(S) + 1$ /* $C(S)$ is the cost of encoding the classes for the records in set S */

$minCost_1 :=$ PruneTree (t_1) /* t_1, t_2 are */

$minCost_2 :=$ PruneTree (t_2) /* t 's children*/

$minCost_t := \min(C(S)+1, C_{split}(t)+1+minCost_1+ minCost_2)$

return $minCost_t$ /* C_{split} : cost of encoding a split */

Figure 1: DT algorithms scheme

In the pruning phase, the entire tree, grown in the previous phase, is shortened to prevent the overfitting and to improve the tree's accuracy [8]. During the growing phase the splitting criteria is determined by choosing the attribute which will best separate the remaining cases in individual classes. This attribute becomes the decision attribute of the node. By using this attribute the splitting criteria is being defined for data division, which is either in the form of $A < v$ ($v \in dom(A)$) for numeric attributes or $A \in V$ ($V \subseteq dom(A)$) for categorical attributes.

The tree pruning increases the accuracy at noise-data and can be made when the tree is built (pre-pruning), or after the induction of the tree (post-pruning). The algorithm is generally used in classification issues which demand the representation of the model in an expressive manner [9]. Especially, at the induction of the tree, the algorithm tries to create a tree which would function as perfect as possible on all available data. The greatest obstacle in this phase is rendered by the noise-data. In the development of the tree, the most important thing is finding the best question to ask at every ramification of the tree. The difference between a good and a bad question is given by the power of the question in organizing data (i.e. decreasing the disorder from the data, as we drift away from the root). The questions have to break the original dataset into segments as homogenous as possible concerning the classes that must be estimated. Certain DT algorithms use heuristic methods in order to choose the questions or they choose them randomly.

In order to select the best data split point, several measures have been proposed (e.g. the algorithms ID3 and C4.5 select the split that minimize the informational entropy of partitions, while the algorithm SLIQ uses gini index). For a data set S containing n records, the informational entropy is being defined as:

$$E(S) = -\sum p_i \log_2 p_i \quad (1)$$

where p_i is the relative frequency of i class. For a split that divides S in to S_1 (n_1 records) and S_2 (n_2 records) subsets, the entropy is:

$$E(S_1, S_2) = \frac{n_1}{n} E(S_1) + \frac{n_2}{n} E(S_2). \quad (2)$$

Gini index for S dataset is being defined as:

$$gini(S) = 1 - \sum p_i^2 \quad (3)$$

and for a split

$$gini_{split}(S) = \frac{n_1}{n} gini(S_1) + \frac{n_2}{n} gini(S_2). \quad (4)$$

Once an attribute is associated with a node, it can't be taken into consideration in the node's children [10]. The most time-consuming part when building a DT is obviously the selection of the split point. For each active node, there must be built up the partition satisfying all the splitting conditions of the node and its predecessors and all the possible divisions of the node must be evaluated for each left-out attribute.

IV. EDUCATIONAL EXPERIMENTS USING DECISION TREES

We'll take an example that will help us exemplify the way DT can be used in educational research: "The Computer Science teacher is at the beginning of 9th grade put in the situation of taking a decision, realizing a classification, a prediction, for a math-computer science class. This way he wants to be able to decide, without having to wait for the end of the first semester, which of the pupils from the class are capable of performing at Computer Science, and which can't. He has older data bases, in shape of a table, which he uses as a training dataset of a DT."

The training dataset contains 12 records comprising personal data of some students: "Birthplace" (BP), "Average secondary school marks" (ASSM) and "Passion for Mathematics" (PM) together with their capacity of making "Performance" concerning Computer Science coded with "Yes" (the student is capable of attaining performance at Computer Science) and "No" (the student is not capable of attaining performance at Computer Science). Obviously, for the construction of the DT the Computer Science teacher makes use of a dataset comprising much more records and much more attributes. We have chosen 12 cases of 4 attributes each in order to explain the working of the mechanism (see Table 1).

With the training dataset we induce the DT. Once built, DT can be pruned, if necessary, and used to classify a new dataset, to which the last attribute is unknown: classification attribute, the one that in forms us if the student, who corresponds to the case, is capable or not of performance. In this manner, the teacher, using the information he owns, can make predictions (regarding some values he doesn't know and needs) and classifications.

Table 1: Training dataset.

BP	ASSM	PM	Performance
urban	7.43	high	Yes
rural	6.00	moderate	No
rural	8.72	high	Yes
rural	7.58	moderate	No

urban	9.02	small	No
rural	9.25	moderate	Yes
rural	7.89	high	Yes
urban	9.32	moderate	Yes
urban	8.34	small	No
urban	10.00	moderate	Yes
urban	6.91	moderate	No
rural	9.55	small	No

A. Building DT

In the construction of the DT it can be noticed that the attribute BP is not used for classification, indicating the fact that its value is irrelevant for making a decision with regard to the possibility of performance of that student. DT has the graphic representation in Figure 2.

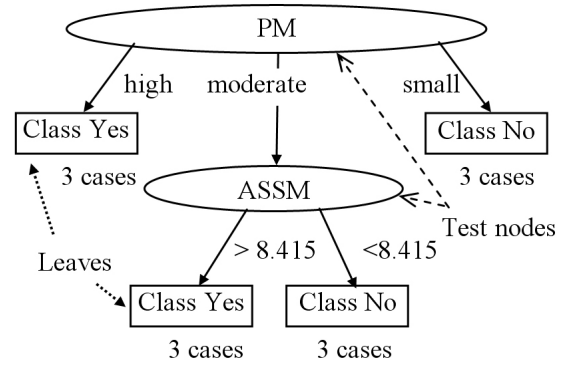


Figure 2. DT induced from training dataset.

Table 2: Test dataset.

BP	ASSM	PM	Performance
rural	8.00	moderate	Yes (error, correct: No)
rural	9.90	moderate	Yes
urban	6.72	high	Yes
rural	8.85	small	No
urban	6.89	high	Yes
rural	7.90	moderate	No
rural	8.89	moderate	Yes
urban	7.90	small	No
urban	8.99	moderate	Yes
rural	9.50	moderate	Yes
urban	7.85	small	No
urban	7.25	high	Yes

B. DT pruning

This is not the case, DT being very small.

C. DT execution

DT execution is use for:

- testing the classification accuracy of the previously built and maybe pruned DT, when we have test data for which we already know the classification attribute values;
- classifying new datasets for which the value of the classification attribute is unknown.

C.1. DT execution on test data – DT verification

Let a new dataset used for testing the accuracy of the previously built DT, see Table 2.

Testing classification accuracy we obtain a classification error (see Table 2) and a classification error rate of 8.33%. Classification error is produced due to the first case assigned by the DT to class No, but yet, in test data this case it is assigned in class Yes.

Based on the DT execution output we can compute the confusion matrix (see Table 3).

Table 3: Confusion matrix.

value	Yes	No	errors
Yes	7	1	1
No	0	4	0
errors	0	1	1

Table 4: A new dataset.

BP	ASSM	PM	Performance
urban	9.43	high	?
rural	9.00	small	?
rural	6.72	high	?
urban	8.58	moderate	?
urban	7.02	small	?
rural	9.05	moderate	?
rural	8.89	high	?
urban	8.32	moderate	?
urban	8.40	high	?
urban	6.30	small	?
urban	9.30	small	?
rural	6.80	high	?
rural	9.70	high	?
urban	10.00	high	?
urban	7.95	moderate	?
rural	8.00	small	?
rural	8.55	small	?
urban	6.98	moderate	?
rural	8.58	moderate	?
urban	6.92	small	?
rural	8.25	moderate	?
rural	7.90	high	?
urban	9.83	small	?
urban	8.49	high	?
rural	6.10	moderate	?
rural	9.75	high	?
rural	8.75	moderate	?
urban	9.82	small	?

The first line of the confusion matrix tells us that from the 8 cases of test data assigned to class Yes, there have been 7 cases classified in class Yes by the DT and one (the error) classified in class No. The second line of the confusion matrix tells us that all the 4 cases of test data assigned to class No have been classified in class No by the DT (no error).

C.2. DT execution on new data

As entries for the DT classifier can be supplied sets of new data for which we only know the values of the first

three attributes BP, ASSM, PM and we want to compute the value of the target attribute, class Performance.

Let the dataset form Table 4 for 28 students from the beginning of the 9th grade, Math-Computer Science class, the value of the class attribute Performance must be predicted for all of them. After the DT execution we find the output presented in Table 5.

Table 5: Classified dataset.

BP	ASSM	PM	Performance
urban	9.43	high	Yes
rural	9.00	small	No
rural	6.72	high	Yes
urban	8.58	moderate	Yes
urban	7.02	small	No
rural	9.05	moderate	Yes
rural	8.89	high	Yes
urban	8.32	moderate	No
urban	8.40	high	Yes
urban	6.30	small	No
urban	9.30	small	No
rural	6.80	high	Yes
rural	9.70	high	Yes
urban	10.00	high	Yes
urban	7.95	moderate	No
rural	8.00	small	No
rural	8.55	small	No
urban	6.98	moderate	No
rural	8.58	moderate	Yes
urban	6.92	small	No
rural	8.25	moderate	No
rural	7.90	high	Yes
urban	9.83	small	No
urban	8.49	high	Yes
rural	6.10	moderate	No
rural	9.75	high	Yes
rural	8.75	moderate	Yes
urban	9.82	small	No

D. Decision Rules extraction from the DT

Many times the DT seems difficult to understand and it is preferred the representation of the classifier through the Decision Rules. In this representation, at the end of each rule, between straight brackets, there are:

- *the support* for each rule (i.e. the number of cases form the dataset to which the rule is applied to) and

- *the confidence* for each rule (i.e. the percent of cases in which a rule is correct related to the number of cases in which the rule is applicable).

For the dataset mentioned above (see Table 1), the Decision Rule set will look like this:

Performance=Yes<-ASSM>= 8.415 & PM = moderate [3/100.0%];
 Performance=Yes<- PM=high [3/100.0%];

Performance=No<-ASSM<=8.415 & PM = moderate [3/100.0%];

Performance=No<-PM=small [3/100.0%].

We should note that there is a rule for each leaf of the DT (see Figure 2).

V. CONCLUSION

We consider that the data with didactic characteristics stored on electronic devices, that have been piling up lately, can represent real “mines” for the extraction of “nuggets” of knowledge in educational research.

We consider that the data with didactic characteristics stored on electronic devices, that have been piling up lately, can represent real “mines” for the extraction of “nuggets” of knowledge in educational research.

REFERENCES

- [1] A. Al-Attar, “Data Mining - Beyond Algorithms”, White Paper, Attar Software, 2004.
- [2] M. S. Almeida, M. Ishikawa, J. Reinschmidt and T. Roeber, *Getting Started with Data Warehouse and Business Intelligence*, International Technical Support Organization, International Business Machines Corporation, 1999, p. 45.
- [3] H. Bozdogan, *Statistical Data Mining & Knowledge Discovery*, Chapman & Hall/CRC, Boca Raton London New York Washington D.C., ISBN 1-58488-344-8, 2004, cap. 2.1.
- [4] C. Borgelt, „A decision tree plug-in for DataEngine”, in *Proc. European Congress on Intelligent Techniques and Soft Computing (EUFIT)*, vol. 2, 1998, pp. 1299-1303.
- [5] K. Thearling, „An Overview of Data Mining Techniques”, White Paper, 2004, Available: <http://www.thearling.com>.
- [6] M. Kamber, L. Winstone, W. Gong, S. Cheng and J. Han, „Generalization and Decision Tree Induction: Efficient Classification in Data Mining”, in *Proc. of 1997 Int'l Workshop on Research Issues on Data Engineering (RIDE'97)*, Birmingham, England, 1997, p. 111.
- [7] W. Du and Z. Zhan, „Building Decision Tree Classifier on Private Data”, in *IEEE International Conference on Data Mining Workshop on Privacy, Security, and Data Mining*, Maebashi City, Japan, Conferences in Research and Practice in Information Technology, vol. 14, 2002.
- [8] K. Uwe and S. O. Dunemann, „SQL Database Primitives for Decision Tree Classifiers”, in *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM'01)*, Atlanta, ACM, GA USA, 2001, pp. 379-386.
- [9] A. Nepomnjashiy, „Data Mining Algorithms: Microsoft SQL Server 2000 vs. "Yukon" SQL Server”, in *DatabaseJournal.com*, February 3, 2004, Available: <http://www.databasejournal.com/>.
- [10] L. A. Bădulescu, „Data Mining Algorithms Based On Decision Trees”, in *Annals of the Oradea University. Fascicle of Management and Technological Engineering*, Publishing House of Oradea University, vol. V (XV), ISSN 1583 – 0691, 2006, pp. 1621-1628.