APPLICATION OF DATA MINING METHODS FOR CLASSIFICATION OF STUDENT RESULTS FROM CONVEYING AN ECTS FACULTY COURSE

Emil Stankov
Institute of Informatics,
Faculty of Natural Sciences and Mathematics
Skopje, Macedonia

Ana Madevska Bogdanova
Institute of Informatics,
Faculty of Natural Sciences and Mathematics
Skopje, Macedonia

Mile Jovanov
Institute of Informatics,
Faculty of Natural Sciences and Mathematics
Skopje, Macedonia

ABSTRACT

In the large datasets (tables) containing grades gained by students according to many criteria (attributes, i.e., columns in the tables) for particular ECTS courses, it is a common situation every criterion to have its own sub criteria, and these sub criteria to be linearly combined to form the corresponding criterion. Therefore, practically, for each student we get a single record (single row in a table) with many attributes. The main goal of our research was to assess how well selected data mining methods are capable of detecting the linear dependency of the final course grade from the course criteria. To this purpose, we collected a dataset containing results from a particular course held at our institution, and we made appropriate experiments. We evaluated three different data mining methods on this dataset (in its raw form) in order to discover how well they would be able to model the criterion for forming the final grade, and to estimate the classification accuracy that they would achieve on independent test sets. In this paper we study the performance of these data mining methods on the dataset, analyze the results and point out further directions for research.

I. INTRODUCTION

According to ECTS, it is a good practice the final course grade to be derived based on many parameters. At our institution, the Institute of Informatics, Faculty of Natural Sciences and Mathematics – Skopje, we always try to follow these guidelines and to use many criteria on which the final grade is derived. Of course, the final grade is simply a linear combination of all individual criteria.

In the large datasets (tables) containing grades gained by students according to many criteria (attributes, i.e., columns in the tables) for particular courses, it is a common situation every criterion to have its own sub criteria, and these sub criteria to be linearly combined to form the corresponding criterion. Therefore, practically, for each student we get a single record (single row in a table) with many attributes. The main goal of our research was to assess how well selected data mining methods are capable of detecting the linear dependency of the final course grade from the course criteria, some of which are themselves linear combinations of their sub criteria. Hence, we needed to evaluate these methods on concrete datasets (in raw form) in order to discover how well they would be able to model the criterion for forming the final grade, and to estimate the classification accuracy that they would achieve on independent test sets. Of course, the final course grades were removed from the test sets.

In order to achieve our goal, we collected a dataset containing results from the course Basics of Programming, held in the winter semester of 2009 at our institution, and we made appropriate experiments. In data mining terminology, each instance in the dataset represents a result achieved by a single student: the grades he/she has gained on number of examinations held during the course (according to ECTS, there are several different forms of examination: colloquia, tests, activities, etc.). The class of each instance is the final course grade that the course teacher assigned to the student at the end of the semester for the scored result. Possible final grades according to the Macedonian academic educational system are 5 (indicating that the student has failed the class), 6 (lowest grade), 7, 8, 9, and 10 (highest grade).

In this paper we study the performance of different data mining methods on the dataset mentioned previously (and described more thoroughly in the following section), analyze the results and point out further directions for research.

II. DATASET DESCRIPTION

The dataset that we used in our experiments contains 30 attributes. As stated previously, every instance in the dataset consists of data regarding a single student.

The first three attributes are of type String and represent each student's personal data (*ID*, *Last Name*, *First Name*). The fourth and the 30th attributes are nominal. The fourth attribute (*Lab_ID*) gives the code of the laboratory group in which the student followed the course classes, while the 30th attribute (*Grade*) is the class attribute and represents the grade (5, 6, 7, 8, 9 or 10) that the course teacher proposed to the student at the end of the semester, i.e. after completing all the course obligations. All the other attributes are numeric and indicate the success of the student on a particular examination during the course.

In accordance with ECTS, continuous assessment of students was conveyed within the course. Three main criteria were used: practical colloquia, theoretical e-tests and activity shown on the lab classes. Each colloquium included solving practical (programming) tasks on a PC. The e-tests contained theoretical questions from the course material. Also, the students were obligated to attend laboratory classes, within which they were required to solve different types of programming exercises. The course teacher(s) recorded the activity of each student on each of the ten classes held during the semester, using integer values in the range from 0 (indicating that the student didn't solve any of the tasks given at the particular class) to 3 (indicating that the student solved all the tasks given at the particular class). Special value (100) was used to indicate that the student missed a class. Each of these criteria participated in the final grade with precisely determined factor. Namely, three e-tests were held, and they participated with 30%, the two colloquia – with 50%, and the lab activity participated with 20% in the final course grade.

Table 1 shows the distribution of the instances from the dataset by classes. As it can be seen, the class 6 contains only 0.9% of the instances, so we expected that none of the learning algorithms would be able to "learn" this class, i.e. to generate a model that would accurately classify instances belonging to this class.

Table 1: Distribution of the instances in the dataset by classes.

Class	Percentage of instances that belong to the class
5	72.5%
6	.9%
7	9%
8	8.1%
9	4.5%
10	5%

III. Data Preprocessing

In its source format, the dataset contained 3 attributes of type String, which inevitably had to be removed. The reason for this is that these attributes are irrelevant in terms of the classification of the instances – they cannot (and must not!) have any kind of influence on the class value. If we assume an existence of correlation of any kind between any one of these attributes and the class attribute, this means that we are assuming that the grade the student receives depends on his name/surname/ID.

In addition, it was easy to see that another irrelevant attribute would be *Lab_ID*. However, in order to see how it would affect the performance of the methods, we decided to make experiments on two datasets – one dataset that did and one dataset that did not contain this attribute.

As stated previously, we left the data as it was – in its raw form. Although we knew that we had attributes that were linearly dependent on other attributes, we didn't remove them intentionally – our goal was to discover this linear dependence.

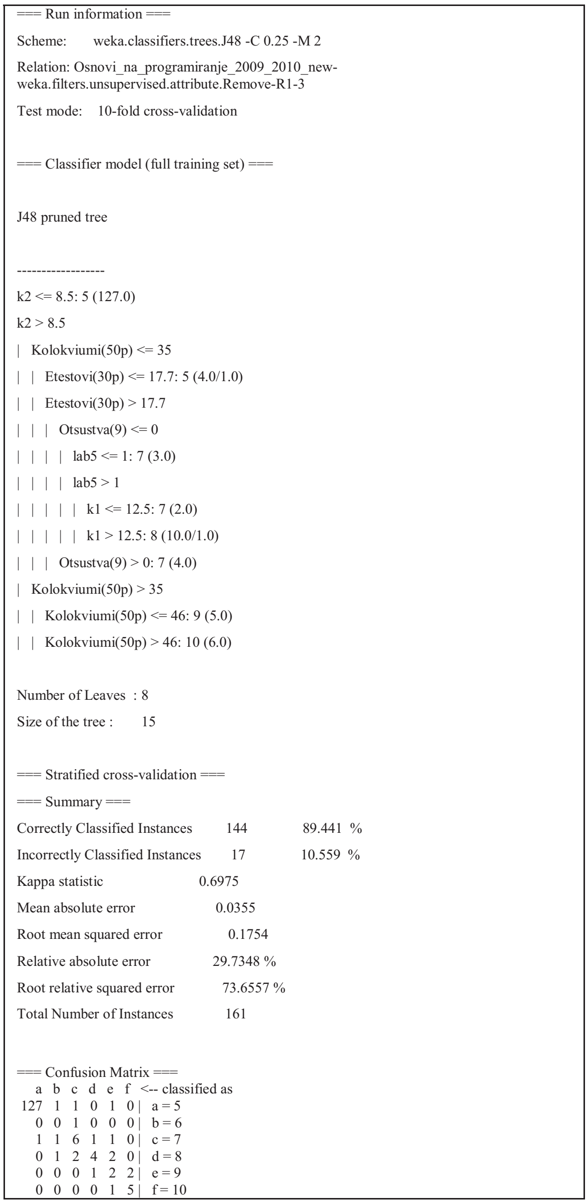
Another problem with the dataset under consideration (as with many real world datasets) were missing attribute values. Fortunately, it was easy to see that all the missing values were a consequence of unintentional mistakes of the course teacher(s) (for example, not entering 0s as results from colloquia or tests on which the student didn't participate). There were 529 missing values in the dataset, and they were all replaced with appropriate values.

IV. Experiments

All the experiments were conveyed using the WEKA data mining toolkit. A detailed description of the WEKA architecture and the wide set of implemented data mining schemes, as well as data preprocessing tools that it provides, can be found in [1].

We estimated the performance of three different data mining schemes on our dataset: C4.5, 1R and the perceptron learning method. Because the size of the training dataset was relatively small, we used 10-fold stratified cross-validation for evaluation of the methods on the dataset to ensure that the results would be representative of what independent test sets would yield.

A. *Classification using C4.5*



```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: Osnovi_na_programiranje_2009_2010_new-
weka.filters.unsupervised.attribute.Remove-R1-3

Test mode:    10-fold cross-validation


=== Classifier model (full training set) ===


J48 pruned tree


------------------

k2 <= 8.5: 5 (127.0)

k2 > 8.5

|   Kolokviumi(50p) <= 35

|   |   Etestovi(30p) <= 17.7: 5 (4.0/1.0)

|   |   Etestovi(30p) > 17.7

|   |   |   Otsustva(9) <= 0

|   |   |   |   lab5 <= 1: 7 (3.0)

|   |   |   |   lab5 > 1

|   |   |   |   |   k1 <= 12.5: 7 (2.0)

|   |   |   |   |   k1 > 12.5: 8 (10.0/1.0)

|   |   |   Otsustva(9) > 0: 7 (4.0)

|   Kolokviumi(50p) > 35

|   |   Kolokviumi(50p) <= 46: 9 (5.0)

|   |   Kolokviumi(50p) > 46: 10 (6.0)


Number of Leaves  : 8

Size of the tree :      15


=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances         144               89.441  %

Incorrectly Classified Instances        17               10.559  %

Kappa statistic                          0.6975

Mean absolute error                      0.0355

Root mean squared error                  0.1754

Relative absolute error                 29.7348 %

Root relative squared error             73.6557 %

Total Number of Instances              161


=== Confusion Matrix ===
   a   b   c   d   e   f   <-- classified as
 127   1   1   0   1   0 |   a = 5
   0   0   1   0   0   0 |   b = 6
   1   1   6   1   1   0 |   c = 7
   0   1   2   4   2   0 |   d = 8
   0   0   0   1   2   2 |   e = 9
   0   0   0   0   1   5 |   f = 10
```

Figure 1: Results from applying the J4.8 algorithm on our dataset.

WEKA's implementation of the C4.5 algorithm is called J4.8. Actually, according to [2], J4.8 implements a later and slightly improved version called C4.5 revision 8, which was the last public version of this family of algorithms before the commercial implementation C5.0 was released.

The output produced by WEKA when executing the J4.8 algorithm on the dataset under consideration is shown in Fig. 1. The parameters of the algorithm used in this final run (results of which are shown in Fig. 1) were selected after some experimentation, as the ones that proved to give the best performance on the dataset (the same applies to all the algorithms we analyze in this paper). The values were the following: confidence level *c* = 0.25; minimum number of instances at leaves: 2; subtree raising was used, and reduced-error pruning was not used.

After the basic information regarding the dataset on which the algorithm was executed, the middle part of Fig. 1 shows the pruned decision tree learned by it. As can be seen, *k2* was selected as the first attribute to split on (attribute placed at the root node of the decision tree). According to the induced tree, if the value of this attribute is less or equal to 8.5, the instance belongs to class 5. This means that for all students who have won 8.5 or less points (out of the maximal 25) on the second colloquium we can conclude that they failed the class.

Continuing the analysis of the output of the algorithm execution in Fig. 1, following the textual representation of the tree and the information about the number of leaves and the total number of tree nodes, is a list of statistical data. All these statistics, as mentioned previously, were calculated using 10-fold stratified cross-validation. First, an estimate of the tree's predictive performance is given: the percentage of correctly and incorrectly classified instances. As can be seen, only 10.559% of the instances were incorrectly classified during the cross-validation. Below, a few more statistics that can serve as an indicator of the tree's performance are shown: the Kappa statistic, the Mean absolute error, the Relative absolute error, etc.

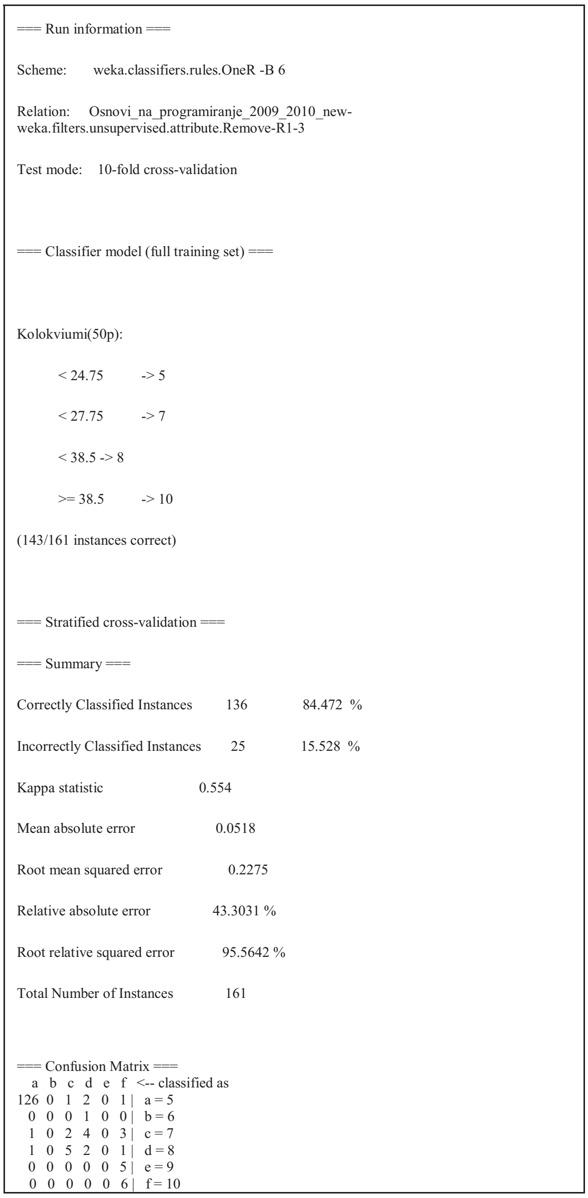
The final part of Fig. 1 shows the confusion matrix. This matrix reveals the distribution of correctly and incorrectly classified instances by classes (for example, for the class 5, 127 instances were correctly classified, while 3 instances were incorrectly classified: one in each of the classes 6, 7 and 9, respectively).

The execution of J4.8 on the dataset obtained by excluding the attribute *Lab_ID* led to identical results.

B. *Classification using 1R*

The output produced by WEKA when executing the 1R algorithm on the dataset under consideration is shown in Fig. 2.

1R generated a rule set that tests the attribute *Kolokviumi* (*Colloquia*). This was totally expected, considering the fact that the sum of the points gained on the 2 colloquia (i.e., the value of the attribute *Kolokviumi*) participates in the formation of the final course grade with the highest factor (50%). The induced rule set contains 4 rules – it does not include rules for the classes 6 and 9. The missing rule for the class 6 is (again) in accordance with our expectations given that the dataset contained only one instance belonging to this particular class. We emphasize again that the 10-fold cross-validation method was used as an evaluation technique.



```
=== Run information ===

Scheme:       weka.classifiers.rules.OneR -B 6

Relation:     Osnovi_na_programiranje_2009_2010_new-
weka.filters.unsupervised.attribute.Remove-R1-3

Test mode:    10-fold cross-validation



=== Classifier model (full training set) ===



Kolokviumi(50p):

        < 24.75        -> 5

        < 27.75        -> 7

        < 38.5 -> 8

        >= 38.5        -> 10

(143/161 instances correct)



=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances        136             84.472  %

Incorrectly Classified Instances       25             15.528  %

Kappa statistic                        0.554

Mean absolute error                    0.0518

Root mean squared error                0.2275

Relative absolute error               43.3031 %

Root relative squared error           95.5642 %

Total Number of Instances             161



=== Confusion Matrix ===
   a   b   c   d   e   f   <-- classified as
 126   0   1   2   0   1 |   a = 5
   0   0   0   1   0   0 |   b = 6
   1   0   2   4   0   3 |   c = 7
   1   0   5   2   0   1 |   d = 8
   0   0   0   0   0   5 |   e = 9
   0   0   0   0   0   6 |   f = 10
```

Figure 2: Results from applying the 1R algorithm on our dataset.

The predictive performance of the rules generated by 1R was estimated as weaker than the one of the decision tree induced by J4.8: during the cross-validation, 15.528% of the instances were incorrectly classified by the rules.

For comparison, when we executed this algorithm on the dataset without the attribute *Lab_ID* we obtained the same results.

C. *Classification using perceptron learning*

The output produced by WEKA when executing the perceptron learning algorithm on our dataset is shown in Fig. 3. Due to practical reasons, we have omitted the induced set of linear models, so Fig. 3 shows only the statistics obtained by evaluation of those models. The selected final values for the parameters of the algorithms were the following: learning rate: 0.3; number of passes through the data (epochs): 500; we used attribute normalization and automatic reset of the neural network (in case it started to diverge from the solution).

```
=== Run information ===

Scheme:     weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -
E 20 -H 0

Relation:     Osnovi_na_programiranje_2009_2010_new-
weka.filters.unsupervised.attribute.Remove-R1-3

Test mode:    10-fold cross-validation



=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances         138               85.7143 %

Incorrectly Classified Instances        23               14.2857 %

Kappa statistic                          0.5665

Mean absolute error                        0.047

Root mean squared error                    0.1798

Relative absolute error                  39.3065 %

Root relative squared error              75.5424 %

Total Number of Instances                161



=== Confusion Matrix ===
   a   b   c   d   e   f   <-- classified as
 126   0   2   1   1   0 |   a = 5
   1   0   0   0   0   0 |   b = 6
   4   0   3   2   1   0 |   c = 7
   1   1   3   4   0   0 |   d = 8
   0   0   0   0   2   3 |   e = 9
   0   0   0   1   2   3 |   f = 10
```

Figure 3: Results from applying the perceptron learning algorithm on our dataset.

The generated perceptron for the dataset under consideration contains 34 nodes in the input layer (one node corresponding to the additional attribute that always has the value 1, one node for each of the 25 numeric attributes and 8 nodes for the nominal attribute *Lab_ID*, which can accommodate exactly 8 values), and 6 nodes in the output layer (one node for each of the possible class values). Hence, 6 hyperplanes have been learned, one for each of the classes, and every hyperplane is defined with 34 weight values. We emphasize again that the 10-fold cross-validation method was used as an evaluation technique.

The predictive performance of the induced perceptron was estimated as weaker than that of the decision tree generated by J4.8, but stronger than the one of the rules generated by 1R: during the cross-validation, 14.286% of the instances were incorrectly classified by the perceptron.

For comparison, when we executed this algorithm on the dataset without the attribute *Lab_ID*, we obtained very similar results: the percentage of incorrectly classified instances was 14.907%.

D. *Comparison of the performance of the methods*

In order to compare the estimated performance of the data mining methods that we used in our experiments, we employed a statistical significance *t*-test of the performance of the first method (J4.8) versus that of the other two (1R and the perceptron learning algorithm). The result obtained when applying this test in WEKA is shown in Fig. 4.

```
Tester:     weka.experiment.PairedCorrectedTTester

Analysing:  Percent_correct

Datasets:   1

Resultsets: 3

Confidence: 0.05 (two tailed)



Dataset                          (1) trees.J4 | (2) rules (3) funct

----------------------------------------------------------------------

'Osnovi_na_programiranje_(100)   88.64 |   84.62     86.78

----------------------------------------------------------------------

                                      (v/ /*) |   (0/1/0)   (0/1/0)



Key:

(1) trees.J48 '-C 0.25 -M 2' -2.17733168393644448E17

(2) rules.OneR '-B 6' -2.4594270021478615E18

(3) functions.MultilayerPerceptron '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H 0' -
5.9906078170482104E18
```

Figure 4: Results from comparing the performance of the three data mining methods using a statistical significance *t*-test.

As it can be seen, a comparison was made on basis of the "percentage of correctly classified instances" statistic. The three methods are displayed horizontally, numbered *(1)*, *(2)* and *(3)*, as the heading of a little table. The labels for the columns are repeated at the bottom, but this time together with the all the values used as parameters for the methods. The value in parenthesis in the second row of the table (100) is the number of experimental runs: in this case, we used 10-fold cross-validation, repeated 10 times. This means that for each method, a single estimate (statistical value) was obtained from each cross-validation run; so 10 estimates were obtained from the 10 runs and the average value was calculated. Finally, the statistical significance *t*-test was performed on these average values.

The percentage of correctly classified instances achieved by each of the three methods is given in the second row of the table: 88.64% for method *(1)*, 84.62% for method *(2)*, and 86.78% for method *(3)*. A symbol placed beside a result would indicate that the corresponding method was statistically better (v) or worse (*) than the baseline method (in this case, it is J4.8) at the specified significance level (0.05). Since those symbols are not present in Fig. 4, we can conclude that none of the two methods is statistically better or worse than J4.8 at the specified significance level.

V. Conclusion and Further Work

In this paper we discussed the existence of linear dependence between the final grades that the teacher assigns to the students on a typical ECTS course and the separate course criteria. We showed how this linear dependence could be successfully revealed using suitable data mining methods. In particular, we collected a dataset containing results from an ECTS course held at our institution, and made appropriate experiments using three different schemes. Then, we evaluated the schemes on this dataset to obtain appropriate classification models (that model the criterion for forming the final grade). Finally, we estimated the classification accuracy of these models. Because the size of the dataset was relatively small, we used 10-fold stratified cross-validation for evaluation to ensure that the results would be representative of what independent test sets would yield. The estimated accuracy of the models generated by these three methods ranged from 84% to 89%, which means that they all managed to detect the dependence very well.

In the future, we plan to extend our research in two main directions. First, we would want to make some more experiments on the same dataset by trying out some other data mining classification schemes (rule-learning methods, multi-response linear regression, support vector machines, etc.) in order to estimate their classification accuracy. And second, we intend to collect more data from some other courses taught at our institution, so that we could try to obtain more accurate and reliable classification models.

On the basis of the results of the experiments, beside the estimation of the classification accuracy of the methods, as an additional gain we could determine the criteria that are relatively unnecessary. This means that a particular method could learn a classification model that would show relatively high performance (expressed in terms of average classification accuracy), without taking into consideration some criteria that were originally included (when forming the final grade).

References

[1] I. H. Witten and E. Frank, *Data Mining – Practical Machine Learning Tools and Techniques*, Second Edition, Morgan Kaufmann Publishers, 2005.

[2] *Weka Software*, The University of Waikato, http://www.cs.waikato.ac.nz/ml/weka/.