Information Sciences 373 (2016) 42–56

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# Finding informative code metrics under uncertainty for predicting the pass rate of online courses

José Otero[a], Luis Junco[a], Rosario Suárez[a], Ana Palacios[b], Inés Couso[c], Luciano Sánchez[a,*]

[a] Department of Computer Science, University of Oviedo, 33204 Gijón, Spain
[b] Department of Computer Science and Artificial Intelligence, CITIC-UGR (Research Center on Information and Communications Technology), University of Granada, 18071 Granada, Spain
[c] Department of Statistics, University of Oviedo, 33204 Gijón, Spain

ARTICLE INFO

*Article history:*
Received 7 September 2015
Revised 17 March 2016
Accepted 26 August 2016
Available online 31 August 2016

*Keywords:*
Low quality data
Vague data
Genetic fuzzy systems
Feature selection
Automatic grading

ABSTRACT

A method is proposed for predicting the pass rate of a Computer Science course. Input data comprises different software metrics that are evaluated on a set of programs, comprising students' answers to a list of computing challenges proposed by the course instructor. Different kinds of uncertainty are accepted, including missing answers and multiple responses to the same challenge. The most informative metrics are selected according to an extension to vague data of the observed Fisher information. The proposed method was tested on experimental data collected during two years at Oviedo University. Yearly changes in the pass rate of two groups were accurately predicted on the basis of 7 software metrics. 73 volunteer students and 1500 source files were involved in the experimentation.



## 1. Introduction

Automated Grading (AG) systems are becoming popular since the advent of Learning Management Systems and Content Management Systems, that allow very large groups of students and teachers to interact via lectures, assignments, exams or gradings. Massive Open Online Courses (MOOC) are a paradigmatic case. Notwithstanding that most MOOCs are ungraded, AG techniques are sought as analytical tools that, for instance, help to detect groups of students with a low accomplishment.

The first use of computers for automating the educational assessment was in 1966, with the Project Essay Grade program (PEG) [17]. This program was the first example of the so called Automatic Essay Scoring (AES) techniques [22]. AES comprises a set of procedures where a training set of essays are hand-scored and different features of the text are measured (total number of words, subordinate clauses, etc.). Regression analysis or other machine learning techniques are used to predict the human-assigned score [12].

In the context of Computer Science online courses, AES techniques are closely related to software metrics-based AG systems [2]. Early works in AG were semi-automated combinations of the task submission system and the grading [18]. *WebToTeach* [4] was the first system that was able to check the submitted source code automatically. Also focused on programming, in [6] and [11] AGs were proposed that compared the output of each student program with the output of a

* Corresponding author. Fax: +34 985 181986.
*E-mail addresses:* jotero@uniovi.es (J. Otero), lajunco@uniovi.es (L. Junco), mrsuarez@uniovi.es (R. Suárez), palacios@decsai.ugr.es (A. Palacios), couso@uniovi.es (I. Couso), luciano@uniovi.es (L. Sánchez).

http://dx.doi.org/10.1016/j.ins.2016.08.090
0020-0255/

correct program, without further measurements of the internals of the source code. The *AutoLEP* system [24] could also compare any implementation of an algorithm against a single model. Furthermore, in [23] a methodology was presented that accomplishes AG by testing the program results against a predefined set of inputs, and also by formally verifying the source code or by measuring the similarities between the control flow graph and the teacher's solution: a linear model was searched that averaged the influence of the three techniques in order to match teacher's and automatic grading in a corpus of manually graded exercises. In the most recent works, see for instance [10], software metrics are used to measure the properties of the students'programs and Artificial Intelligence (AI) methods are used to determine how close the programs submitted by students and the solutions provided by the teacher are. Lastly, in [14] another software metrics-based AG model was defined, where students picked out their tasks from a list of problems proposed by the course instructor and the final marks were predicted with a rule-based model. A feature selection algorithm was also provided that found out the most relevant metrics and supported the different types of uncertainty involved in that particular setup.

The present contribution is derived from this last reference [14] and it addresses a new kind of problem for ungraded courses, where predicting individual grades is of secondary importance but knowing in advance certain measurements of central tendency is needed. For instance, as mentioned before, in ungraded courses and MOOCs the instructor may want to estimate the hypothetical "pass rate", that is the fraction of people who would pass an exam related to the concepts taught in the course. This problem is a case of point estimation, that could be solved with the same AG techniques seen before, i.e. predicting the grade of each student and counting how many of them would pass the threshold. But the question now arises of whether a direct prediction of the pass rate is possible without the need of an intermediate AG stage. Furthermore, one may speculate that the best sets of features for each type of problem (AG and pass rate estimation or, alternatively, regression and point estimation) are different.

Because of these reasons, in this paper a feature selection algorithm designed for the new point estimation problem is proposed, and an interval-valued estimator of the pass rate is also defined. The feature selection algorithm makes use of a fuzzy extension of the observed Fisher information, and will be defined in Section 2. The paper also contains an experimental validation in Section 3, where a case study is described with actual data collected in classroom lectures in 2014 and 2015 at Oviedo University, Spain. Concluding remarks and future work are discussed in Section 4.

## 2. Feature selection for the direct estimation of the pass rate

As mentioned, the grading process is intended to determine the level of achievement of each student. A set of programming challenges or "assignments" is considered. Each assignment is related to a single matter or "programming concept". Each student answers zero or more times every assignment. In order to learn the AG model, it will also be assumed that each student is assigned a numerical grade based on his/her performance. The same set of software metrics is applied to all answers.

For instance, suppose that a student answered once to assignment 1, proposed two different solutions to assignment 3 and did not solve assignment 2. Assignments 1, 2 and 3 are related to the programming concepts 'A','B'and'C', respectively. Let the measured values of three software metrics SM1, SM2 and SM3 at these three solutions be:

| Assignment | Answer number | SM1 | SM2 | SM3 | Programming concept |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 3 | A |
| 3 | 1 | 7 | 8 | 9 | C |
| 3 | 2 | 6 | 4 | 7 | C |

The vector of input features for this student will be the cartesian product of the sets of software metrics (SM) and programming concepts (PC), which in this case is as follows:

| SM1-PCA | SM2-PCA | SM3-PCA | SM1-PCB | SM2-PCB | SM3-PCB | SM1-PCC | SM2-PCC | SM3-PCC |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | $\emptyset$ | $\emptyset$ | $\emptyset$ | {6,7} | {4,8} | {7,9} |

These vectors of features are joined to form a set-valued training dataset, which is a matrix whose rows are the students in the course and whose columns are numerical or set-valued input features, as shown in the preceding example. Each of the cells of this matrix is a random rample of the distribution of every SM, conditioned to a student (row) and programming concept (column). A possibilistic view of the uncertainty is assumed, for which there is not a need for introducing arbitrary hypotesis about the probability distributions of these SMs, and fuzzy sets are used for describing partial knowledge about the data. Consequently, each student will be assigned a vector of fuzzy-valued metrics, whose length is the product of SMs and PCs.

Following references [21] and [14], in this work bootstrap techniques are used for estimating a finite number of confidence intervals for the mean value of each SM at different confidence levels $1-\alpha$. In addition to this (see [9] and [8]), the membership function of the fuzzy SM is regarded the contour function of the possibility measure that upper-bounds the set of probability measures satisfying the restrictions indicated by this set of confidence intervals and their associated confidence levels. That is to say, fuzzy SMs are produced such that their $\alpha$-cuts are the mentioned confidence intervals with degrees $1-\alpha$. These fuzzy SMs model the available knowledge about the true expected value of the set of SMs and also