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	∅	∅	∅	{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 α-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

about its dispersion. Lastly, missing values are represented by intervals spanning the whole range of the metric, i.e. fuzzy sets with membership 1 to the whole domain of the variable.

2.1. Formal definition of the problem

Let Ω be the set of potential answers to the programming challenges for a certain time interval. A sample of solutions is available, and $\omega \in \Omega$ is one of these solutions. The student that wrote the solution ω is $S(\omega)$ and the PC of the corresponding assignment is $T(\omega)$. M is the number of SMs and P is the number of PCs.

The random vector $X(\omega) = (X_1(\omega), \ldots, X_M(\omega))$ comprises the measured values of the SMs for the solution ω. For the s-th student and t-th PC, $\Omega_{st} = \{\omega \in \Omega : S(\omega) = s, T(\omega) = t\}$ and the mean value of the m-th SM is

$$\mu_{stm} = E(X_s||\Omega_{st}) = \frac{\sum_{\Omega_{st}} X_m(\omega)}{\sum_{\Omega_{st}} 1} \tag{1}$$

Lastly, let $\tilde{\mu}_{stm}$ be fuzzy random variables whose cuts $[\tilde{\mu}_{stm}]_\alpha$ are nested bootstrap estimations of confidence intervals with level $1-\alpha$ of the values μ_{stm} [21]. Each student will be described by a vector comprising $P \cdot M$ fuzzy features, and the training dataset mentioned before is the fuzzy-valued matrix

$$[\tilde{\mu}_s] = [\tilde{\mu}_{s11}, \ldots, \tilde{\mu}_{s1M}, \ldots, \tilde{\mu}_{sP1}, \ldots, \tilde{\mu}_{sPM}]. \tag{2}$$

$\tilde{\mu}_{stm}$ represents our incomplete information about the value of the mth SM when applied to the tth PC and the sth student, having into account that we have observed a limited number of solutions submitted by the student.

2.2. Mutual information-based ranking for AG

In reference [14], the AG problem was regarded as a regression problem with fuzzy inputs and crisp output. The purpose of the regression is to obtain a fuzzy restriction of the performance y_s of the sth student, which is assumed known, given the fuzzy vector $\tilde{\mu}_s$ (see Eq. 2).

To solve this regression problem, an intermediate crisp function $f : \mathbb{R}^m \to \mathbb{R}$ was introduced first, and the fuzzy grade $\widetilde{F_s}$ of the sth student is defined through this function f as follows:

$$[\widetilde{F_s}]_\alpha = \{ f(x) \mid x \in [\tilde{\mu}_s]_\alpha \} \tag{3}$$

where $[\widetilde{F_s}]_\alpha$ and $[\tilde{\mu}_s]_\alpha$ are the α-cuts of the fuzzy grade and the input vector, respectively. In turn, the α-cuts of the residual of the fuzzy approximation can be expressed as the set of residuals of the model f for all the selections of each $[\tilde{\mu}_s]_\alpha$:

$$[\widetilde{R}]_\alpha = \left\{ \sum_s (y_s - f(x_s))^2 \mid x_s \in [\tilde{\mu}_s]_\alpha \text{ for all } s \right\} \tag{4}$$

This way, the appropriateness of a function f to the dataset is assessed with a fuzzy number. For each $\alpha \in (0, 1]$, $[\tilde{R}]_\alpha$ represents a $1-\alpha$-confidence interval for the actual value of the SSE (sums of squares of errors) associated to f: it is known, with confidence greater than or equal to $1-\alpha$, that the f-estimate of the grade of the student belongs to $[\tilde{F_s}]_\alpha$. As mentioned before, such a confidence-level representation agrees with the possibilistic interpretation of fuzzy sets. Therefore, for an arbitrary positive value $e \in \mathbb{R}^+$, the membership value $\tilde{R}(e)$ denotes the degree of possibility that the actual value of the SSE is equal to e. A fuzzy ranking [5] was used in [14,15] to introduce an order among these residuals and consequently among the different alternatives for f.

For finding the combinations of SM and PC that are relevant for the AG problem, in the same reference [14] a method was proposed for extending crisp feature selection algorithms to fuzzy datasets. Let be supposed that a feature selection algorithm inputs a crisp matrix of features $[z_s]$ with the same dimensions as $[\tilde{\mu}_s]$ and a column vector of performances $[y_s]$. If there are N students in the course, the input data is a matrix or dataset D,

$$D = \begin{bmatrix} z_1 & y_1 \\ \ldots & \ldots \\ z_N & y_N \end{bmatrix} \tag{5}$$

and the feature selection algorithm outputs a ranking of the features, which is a permutation of the indexes $1, \ldots, PM$:

$$\mathrm{FS}(D) = (r_1, \ldots, r_{PM}) \tag{6}$$

where $\mathrm{FS}_1 = r_1$ is the number of the column of the most important variable, $\mathrm{FS}_2 = r_2$ is the position of the next variable in importance, etc.

If this algorithm is applied to each of the selections of a particular cut $[\tilde{\mu}_s]_\alpha$ of the fuzzy features, the set of results can be aggregated to form an interval-valued set of ranks. For instance, if a feature is assigned the interval-valued rank [3, 5], it means that a selection was found for which this feature was the third most important one, and another selection was found where the same variable was the fifth in order of importance. Also, no selections were found where the variable was ranked better than 3 or worse than 5.

If the process is repeated for different cuts, the interval-valued ranks can be stacked to form a fuzzy-valued rank $\widetilde{\mathrm{FS}}$, where each feature is assigned a fuzzy set defining how relevant the feature is:

$$[\widetilde{\mathrm{FS}}_k]_\alpha = \{\, \mathrm{FS}_k([(z_1, y_1), \ldots, (z_N, y_N)] \mid z_s \in [\tilde{\mu}_s]_\alpha \text{ for } s = 1 \ldots N \},\ \text{for } k = 1 \ldots PM \tag{7}$$

The same procedure is also suitable for randomized feature selection algorithms, where successive launches of the feature selection algorithm over the same crisp dataset would produce different permutations (for instance, random forest feature importance measures [19]). The output of a randomized feature selection algorithm can be regarded as a matrix $\mathrm{RFS}(D) = [p_{ij}]$ where p_{ij} is the probability that there are $j-1$ variables more relevant than the i-th feature of the dataset (observe that deterministic algorithms, mentioned before, would produce permutation matrices $[p_{ij}]$). The fuzzy rank for these randomized feature selection algorithms is defined as follows:

$$[\widetilde{\mathrm{FS}}_k]_\alpha = \{\, r \mid p_{kr} \geq \alpha \text{ and } [p_{kr}] = \mathrm{RFS}(\{(z_1, y_1), \ldots, (z_N, y_N)\}) \text{ and } z_s \in [\tilde{\mu}_s]_\alpha \text{ for } s = 1 \ldots N \},\ \text{for } k = 1 \ldots PM \tag{8}$$

2.3. Anonymization of the marks and pass rate prediction

Given a certain grade threshold, a classification problem can be defined where a binary mark '1' is assigned to a student if his/her grade is higher than the threshold, and '0' otherwise. The pass rate of the course can be readily predicted as the fraction of individuals of class '1' guessed for the test dataset. We will refer to the latter procedure as the "indirect" or "paired" approach, in contrast with the "direct" or "unpaired" procedure that will be be introduced in this section. It is remarked that the indirect approach makes use of an intermediate AG stage, and the direct estimation maps the data to a point estimation of the expectation of the distribution of the output given the input, without mediation of an AG stage.

The rationale under the direct estimation model is that there is not a need to accurately predict the mark of each student in order to estimate the pass rate of the whole course. Suppose, for instance, that the data is anonymized, and each mark is replaced by a random binary value, following the principle of the minimum privacy information loss ratio [3]. The fraction of predicted '1's in the anonymized sample must be the same as the pass rate in the training set (in other words, this kind of anonymization is a permutation of the marks of the students). The question can be raised whether a permutation exists for which a better AG model can be found. This will be illustrated in the following example.

Example 2.1. The following training dataset is considered. A single software metric is used to predict the grades of the students, and a linear regression model is used. The grade threshold is 5.

Software metric	Grade	Anonymized grade	Best Linear Model	Best Linear Model for Anonymized grades
2	3	7	5.9	6.3
1	8	8	6.7	7.9
4	4	4	4.3	3.1
3	7	3	5.1	4.7
	True pass rate: 50%		Estimation #1: 75%	Estimation #2: 50%

Observe that a model was found that predicts the anonymized grades better than the best model for the actual grades. The pass rate estimated from this model also improves the indirect ("Best Linear Model") estimation.

The anonymization stage can also be regarded as a different criteria for assessing the fitness of an AG model. In the preceding subsection, the error function of a (crisp) model was the sum of the squared differences between data and predictions. This kind of deviation between model and data could also have been measured with a statistical goodness of fit test for *paired* data. In this section it is being proposed that, in case the model is only intended for predicting the fraction of students that would pass the exam, the deviation between model and data might as well be measured by means of a statistical test for *unpaired* data. The model minimizing the unpaired deviation could possibly be better than the paired one. Hence, at this point, the following two questions can be raised:

1. Are direct (unpaired) and indirect (paired) estimations equally accurate?
2. Are the most relevant features the same for both cases? In other words, could a software metric deemed irrelevant for the paired problem be significant for the unpaired problem?

Counterexamples for both questions will be given in the empirical study included in the next section. Actual data from students at Oviedo University in the year 2014 was used to build both the indirect and the direct models. In turn, these models were used to predict the pass rate of the students of 2015, and the results compared to the actual statistics of the courses.

2.4. Feature ranking through fisher information

Measurements such as correlation or mutual information assess the dependence between features and output variables and cannot be used to quantify how much a given feature contributes to the accuracy in the estimation of the pass rate,

which is not a random variable but a parameter of the probability distribution of a variable. Notwithstanding this, the information contained in a sample about an unknown parameter of its probability distribution can be guessed through the observed Fisher information. Hitherto, Fisher information has not been used to perform feature selection. In this section a method is described that makes use of this metric for comparing the relevance of different features in the direct model. The formal definition of this procedure and their related definitions are given below.

Recall that ω is a solution to an assignment written by certain student $S(\omega)$ and $X(\omega) = (X_1(\omega), \ldots, X_M(\omega))$ is a vector comprising M different measurements (software metrics) taken on the solution ω. y_s is the grade of the student s, as defined in the preceding section. μ_{stm} is the mean value of the mth SM for the sth student and tth PC. The available knowledge about μ_s is given by a fuzzy vector $\tilde{\mu}_s$ and the vector $\tilde{\mu}_s = [\tilde{\mu}_{s11}, \ldots, \tilde{\mu}_{s1M}, \ldots, \tilde{\mu}_{sP1}, \ldots, \tilde{\mu}_{sPM}]$ comprises the features describing the sth student. The input dataset comprises N students, as before, whose grades are $[y_1, \ldots, y_N]$.

Let be assumed for the time being that the features $\tilde{\mu}_s$ are crisp, hence the input data is the same dataset D seen before:

$$D = \begin{bmatrix} z_1 & y_1 \\ \ldots & \ldots \\ z_N & y_N \end{bmatrix} \tag{9}$$

Let $\theta \in \mathbb{R}^q$ be a parameter vector that indexes the probability distribution of the grades in a certain family: $F_\theta(z) = P(\omega \in \Omega \mid y_{S(\omega)} \leq z)$. In the particular case studied in this paper, θ is the pass rate. Let $\hat{\theta}$ be the estimator of the unknown parameter vector θ. The corresponding log-likelihood function L satisfies

$$L(y|\theta) = \log F'_\theta(y). \tag{10}$$

The score vector of the log-likelihood function is

$$u(y|\theta) = \frac{\partial L(y|\theta)}{\partial \theta}. \tag{11}$$

The Fisher information that is carried about the unknown parameter θ is the variance of $u(y|\theta)$, which in turn is the negative of the second derivative of $L(\theta)$

$$I(\theta) = \mathrm{Var}(u(y|\theta)) = -E\left\{\left(\frac{\partial^2 L(y|\theta)}{\partial\theta_j\ \partial\theta_k}\right)\right\} \quad \text{for } j,k = 1\ldots,q. \tag{12}$$

Its sample-based version is the observed Fisher information. For the matrix D defined before, this information is another matrix $I_{\mathrm{Obs}} \in \mathbb{R}^{q\times q}$,

$$I_{\mathrm{Obs}}(\hat{\theta}) = \left[-\frac{1}{N}\sum\left(\frac{\partial^2 L(y|\theta)}{\partial\theta_j\ \partial\theta_k}\middle|\ D\right)(\hat{\theta})\right] \quad \text{for } j,k = 1\ldots,q. \tag{13}$$

It is proposed that the estimator $\hat{\theta}$ of the unknown parameter vector θ is defined through two auxiliary functions $T : \mathbb{R}^N \to \mathbb{R}$ and $g : \mathbb{R}^{MP} \to \mathbb{R}$ as follows:

$$\hat{\theta} = T(g(z_1), \ldots, g(z_N)). \tag{14}$$

The intuitive meaning of this definition is clearer if the function g is compared to the predictor f introduced in Section 2.2. Observe that both f and g are functions of the input vector z_s. $f(z_s)$ is intended to approximate y_s. On the contrary, $g(z_s)$ does not need to approximate y_s: it is enough that the empirical probability distributions of the values $g(z_s)$ and y_s are similar, thus the statistic T produces similar results when applied to the actual population $\{y_1, \ldots, y_N\}$ and when applied to the surrogated population $\{g(z_1), \ldots, g(z_N)\}$.

Suppose, for instance, that y_s does not denote the raw grade but directly determines whether the student has passed the exam ($y_s = 1$) or not ($y_s = 0$). The indirect estimator would be a binary function such that most of times $f(z_s) = y_s$ and then counting how many times $f(z_s) = 1$. An alternative estimation, following the direct method proposed here, consists in deciding that the operator T is the mean value, and finding a binary function g such that the number of times that $g(z_s)$ is 1 is the same as the number of times that $y_s = 1$. It is clear that the solution $g = f$ fulfills these premises, but many other different solutions for g exist that are equally accurate.

In this paper, T is a design parameter and will be preset (to the average or any other suitable aggregation operator) and g will be obtained by machine learning techniques, i.e. a learning algorithm $\mathcal{L}$ exists that inputs a dataset D and outputs the function $\mathcal{L}(D) = g_D$ such that $T(g_D(M(\,\cdot\,)))$ is the maximum likelihood estimator of θ given D.

Observe also that the value of $I_{\mathrm{Obs}}(\hat{\theta})$ is not always enough for ranking the features, as different features can share the same observed Fisher information at the dataset D. Given that an estimator that can predict the pass rate of the whole course is searched for, but at the same time this estimator must also be capable of predicting the pass rate of certain subsets of the students, a partition $P = \{P_1, \ldots, P_r\}$ of $\{1, \ldots, N\}$ is defined first, comprising the subsets of students of interest. The loss of information that happens when the estimator $\hat{\theta} = T(g_D(\cdot))$ is replaced by the r estimators $\hat{\theta}_u = T(g_{P_u}(\cdot))$, $u = 1, \ldots, r$ is

$$\text{Information Loss}(P) = ||I_{\mathrm{Obs}}(\hat{\theta})|| + \left|\left|\left[\frac{1}{N}\sum_{u=1}^{r}\sum\left(\frac{\partial^2 L}{\partial\theta_j\ \partial\theta_k}\middle|\ P_u\right)(\hat{\theta}_u)\right]\right|\right| \tag{15}$$

and the function g with the highest loss of information will be preferred, i.e. the estimator whose sensitivity is higher.

Finally, the uncertainty in the determination of the values of the metrics is managed as follows. Given the fuzzy dataset

$$\widetilde{\mathbf{D}} = [(\tilde{\mu}_1, y_1) \dots (\tilde{\mu}_n, y_n)] \tag{16}$$

whose meaning was explained in the preceding section, it is proposed to use the following generalization of the observed Fisher information, defined through its α-cuts:

$$[\widetilde{I}_{\text{Obs}}(\hat{\theta})]_\alpha = \{I_{\text{Obs}}(\hat{\eta}) \mid \hat{\eta} = T(g(m_1), \dots, g(m_n)),\ m_1 \in [\tilde{\mu}_1]_\alpha, \dots, m_n \in [\tilde{\mu}_n]_\alpha\}. \tag{17}$$

The feature selection process is illustrated by means of the following example:

Example 2.2. An hypothetical course comprising 50 students and three metrics SM_1, SM_2 and SM_3 is considered. A single programming concept is analyzed. Grades are numbers between 0 and 10. An student passes the course when scores 5 or better. A computer-generated dataset is defined where

- The inputs (values of the software metrics) are generated at random, with independent uniform distributions in [0, 1].
- The outputs are functions of the first and the second inputs. If $\sqrt{(SM_1^2 + SM_2^2)} \geq \frac{1}{2}$ then the student is given a mark between 5 and 10, chosen at random. Otherwise, the mark is an also random number between 0 and 5.

Therefore, the first two metrics convey information about the mark, but the third one is completely irrelevant. The procedure described in the text is applied to this synthetic data to detect the least useful metric among SM_1, SM_2 and SM_3, which should be SM_3.

Let y_s take binary values: $y_s = 1$ if the student s scores 5 or higher, $y_s = 0$ otherwise. The grades y_s follow a Bernouilli distribution, that depend on a parameter θ: $p(y_s = 1 \mid \theta) = \theta,\ p(y_s = 0 \mid \theta) = 1 - \theta$. Furthermore,

$$L(y|\theta) = \begin{cases} \log\theta & y = 1 \\ \log(1-\theta) & y = 0 \end{cases} \tag{18}$$

and

$$-\frac{\partial^2 \log L}{\partial \theta^2} = \begin{cases} \dfrac{1}{\theta^2} & y = 1 \\ \dfrac{1}{(1-\theta)^2} & y = 0 \end{cases} \tag{19}$$

The function T that computes the estimation of θ through the values of $g(z)$, on the basis of the kth metric is [13]

$$\hat{\theta}(k) = \frac{1 + \sum_{s=1}^{50} g(z_{ks}, \tau)}{50 + 2}. \tag{20}$$

The auxiliary function g in the preceding definition was chosen to be a threshold that depends on a parameter τ:

$$g(z, \tau) = \begin{cases} 0 & z < \tau \\ 1 & z \geq \tau \end{cases} \tag{21}$$

thus determining the maximum likelihood estimator amounts to finding numerically the value of τ that maximices the likelihood.

The maximum likelihood estimations $\hat{\theta}$ are shown in the following table, along with their related likelihoods and observed Fisher informations. Observe that the estimation of the average number of students that pass the exam on the basis of SM_3, which is completely independent of the grades, has the same likelihood as those arising from SM_1 and SM_2, which are not:

Metric	$\hat{\theta}$	$L(\hat{\theta})$	$I_{\text{Obs}}(\hat{\theta})$
SM_1	0.115	−0.00734	10.13
SM_2	0.115	−0.00734	10.13
SM_3	0.115	−0.00734	10.13

A partition is introduced to check the sensitivity of the estimator through the information loss defined in Eq. (15). The partition comprises two subsets, defined by the values of the corresponding metric being lower or higher than a given splitting point. The information losses for each possible splitting point are shown in Fig. 1 and the results for the splitting for which the estimator losses the most information are displayed in the following table. Observe that there are partitions for both SM_1 and SM_2 for which the information loss is higher that 8 units, while the highest loss is 3.71 for the variable SM_3 (see table below) and therefore we can conclude that SM_3 is the less interesting metric for predicting the percentage of students that failed the exam, as desired.

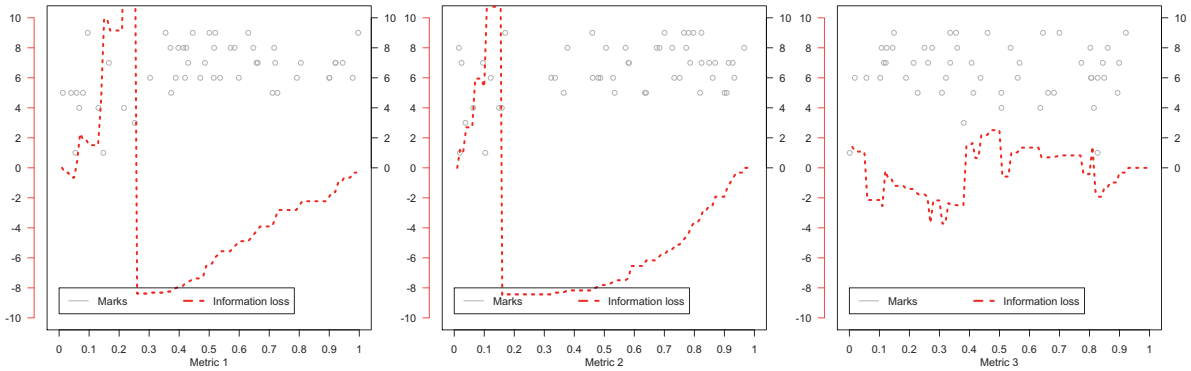

Fig. 1. Information losses for each splitting point. For metrics 1 and 2 the information loss changes when the estimator is evaluated for each element of the partition, but this does not happen for metric 3, where the information loss is roughly constant for all the partitions considered. This is consistent with the synthetic data used in this example: marks should depend on SM_1 and SM_2 but not on SM_3.

Metric	Split	$\hat{\theta}_1$	$L_1(\hat{\theta}_1)$	$I_{Obs,1}$	$\hat{\theta}_2$	$L_2(\hat{\theta}_2)$	$I_{Obs,2}$	Information loss
SM_1	0.29	0.50	−0.0139	4.00	0.02	−0.0005	1.05	8.38
SM_2	0.16	0.58	−0.0134	4.06	0.02	−0.0004	1.05	8.48
SM_3	0.35	0.10	−0.0047	6.48	0.17	−0.0085	6.39	**3.71**

Example 2.3. An interval-valued problem is generated where the values of SM_1, SM_2 and SM_3 have been replaced by intervals of random length (uniform in [0,0.01]) enclosing the actual value of the metric. All the definitions in the preceding example apply.

The maximum likelihood estimates ("maximum" being understood in the total order defined by the interval ranking defined in [21]) are:

Metric	$\hat{\theta}$	$L(\hat{\theta})$	$I_{Obs}(\hat{\theta})$
SM_1	[0.115,0.115]	[−0.00734, −0.00734]	[10.13,10.13]
SM_2	[0.115,0.115]	[−0.00734, −0.00734]	[10.13,10.13]
SM_3	[0.096,0.115]	[−0.00740, −0.00734]	[10.13,14.06]

Again, there are not relevant differences between the likelihoods of the estimations for the three software metrics. However, the sensitivity analysis is coherent with the preceding example, and SM_3 is detected again:

Metric	Split	$\hat{\theta}_1$	$L_1(\hat{\theta}_1)$	$I_{Obs,1}$	$\hat{\theta}_2$	$L_2(\hat{\theta}_2)$	$I_{Obs,2}$	Inf. loss
SM_1	0.3	[0.50,0.57]	[−0.0139, −0.0139]	[4.00,4.25]	[0.025,0.025]	[−0.0005, −0.0005]	[1.05,1.05]	[8.31,8.38]
SM_2	0.16	[0.58,0.66]	[−0.0137, −0.0135]	[4.06,4.95]	[0.025,0.025]	[−0.0005, −0.0005]	[1.05,1.05]	[8.31,8.48]
SM_3	0.27	[0.12,0.12]	[−0.0052, −0.0052]	[6.02,6.02]	[0.16,0.16]	[−0.0082, −0.0082]	[6.46,6.46]	**[3.67,7.59]**

In the upper part of Fig. 2 the propagation of the uncertainty to the observed Fisher information is displayed. The same dataset used in the preceding example was used, with added interval-valued epistemic uncertainty whose average width was of 1% of the variable range. Observe that the difference between the graphs in Figs. 1 and 2 is much higher for SM_3. The amount of uncertainty introduced by the uncertain inputs is remarkable in the latter case. This variability is much reduced if the sample size is higher. In the lower part of the same Fig. 2 a new sample of size 1000 was used. Observe again that the information loss for SM_3 clearly shows that the variable is insensitive with respect to the partition of the set of students, but there are partitions for which SM_1 and SM_2 have a large loss in the observed Fisher information. This means that an estimator $\hat{\theta}$ built from SM_3 would be almost independent of the value of the software metric, and estimators of $\hat{\theta}$ that depend on SM_1 or SM_2 would produce different pass rates for different subsets of the students. It is expected that a model built on SM_1 and/or SM_2 has a higher predictive accuracy than a model built on SM_3, because the insensivity of $\hat{\theta}$ to changes in SM_3 means that a model which is based on this metric will predict the same pass rate for future curses even though the values of the software metrics are different from those in the training data.

2.5. Estimation from multiple metrics

It is proposed that the estimation of the pass rate from multiple metrics is solved with an ensemble of predictors, each one depending on a subset of the variables. The members of the ensemble depend on the choice of anonymization function

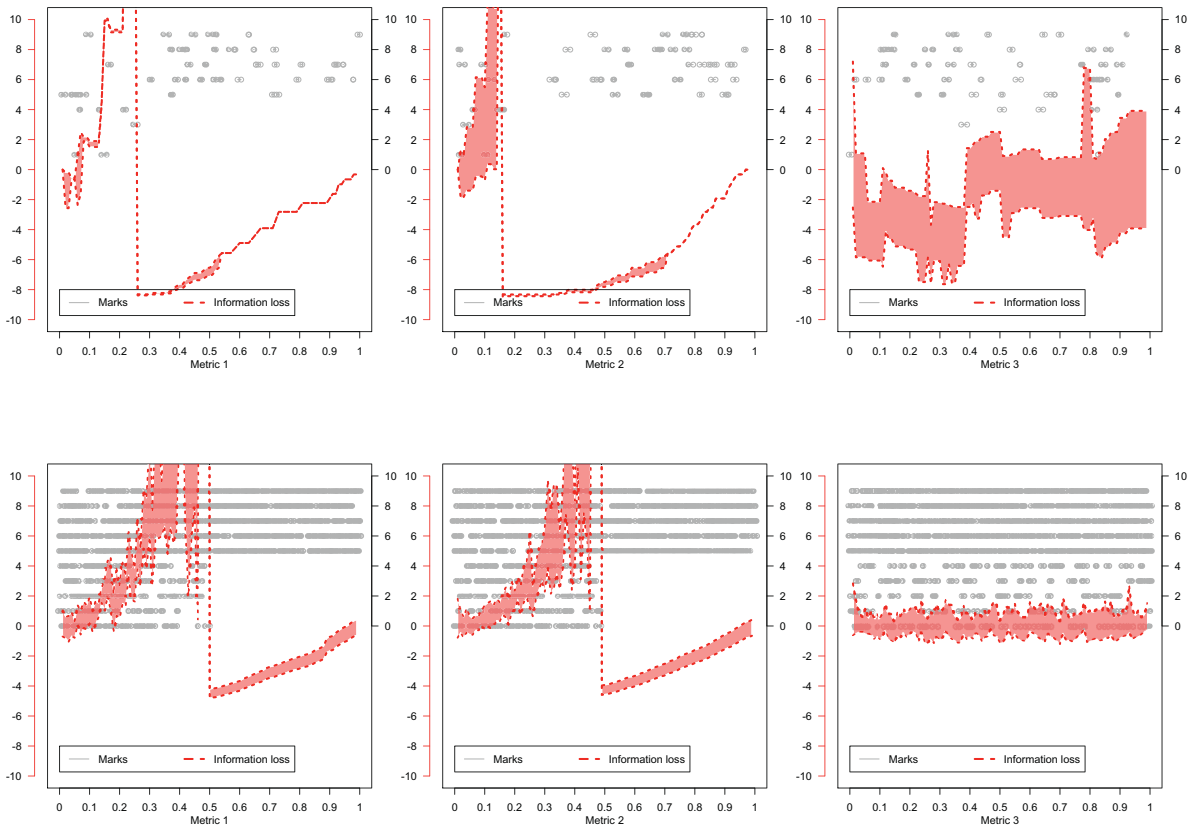

Fig. 2. Upper part: Information losses for each splitting point when 1% of interval uncertainty is added to the data. The same conclusions seen for Fig. 1 apply. Lower part: Same analysis when the sample size is 1000.

g. Suppose for instance that the ensemble comprises F members and the fth member depends on a group of m metrics $(z_{f1}, \ldots, z_{fm})$. Suppose also that g is a threshold of a linear combination of the software metrics,

$$g(z_{f1}, \ldots, z_{fm} \mid w_1, \ldots, w_m, \tau) = \begin{cases} 1 & w_{f1} \cdot z_{f1} + \ldots + w_{fm} \cdot z_{fm} > \tau \\ 0 & w_{f1} \cdot z_{f1} + \ldots + w_{fm} \cdot z_{fm} \leq \tau. \end{cases} \tag{22}$$

Let T be defined as in Example 2.2 (Eqs. (14) and (20)). In this case, the estimation of the pass rate produced by the fth member of the ensemble is

$$\hat{\theta}(z_{f1}, \ldots, z_{fm}) = \frac{1 + \sum_{s=1}^{N} g(z_{f1s}, \ldots, z_{fms} \mid w_{f1}, \ldots, w_{fm}, \tau_f)}{N+2}. \tag{23}$$

The likelihood of $\hat{\theta}(z_{f1}, \ldots, z_{fm})$ on a dataset D is

$$L(\hat{\theta}(z_{f1}, \ldots, z_{fm})|D) = \prod_{s=1}^{N} L(y_s|\hat{\theta}(z_{f1}, \ldots, z_{fm})) \tag{24}$$

where y_s is 1 if the sth student passed the course (0 else) and

$$L(y|\theta) = \begin{cases} \theta & y = 1 \\ 1 - \theta & y = 0. \end{cases} \tag{25}$$

The coefficients $(w_{f1}, \ldots, w_{fm})$ and the threshold τ_f that maximize Eq. (24) must be searched with a suitable optimization algorithm. Lastly, the ensemble of predictors is a weighted average, where the fth member has a number v_f of votes (with $\sum v_f = 1$)

$$\hat{\theta}(z_1, \ldots, z_{PM}) = \sum_{f=1}^{F} v_f \cdot \hat{\theta}(z_{f1}, \ldots, z_{fm}) \tag{26}$$

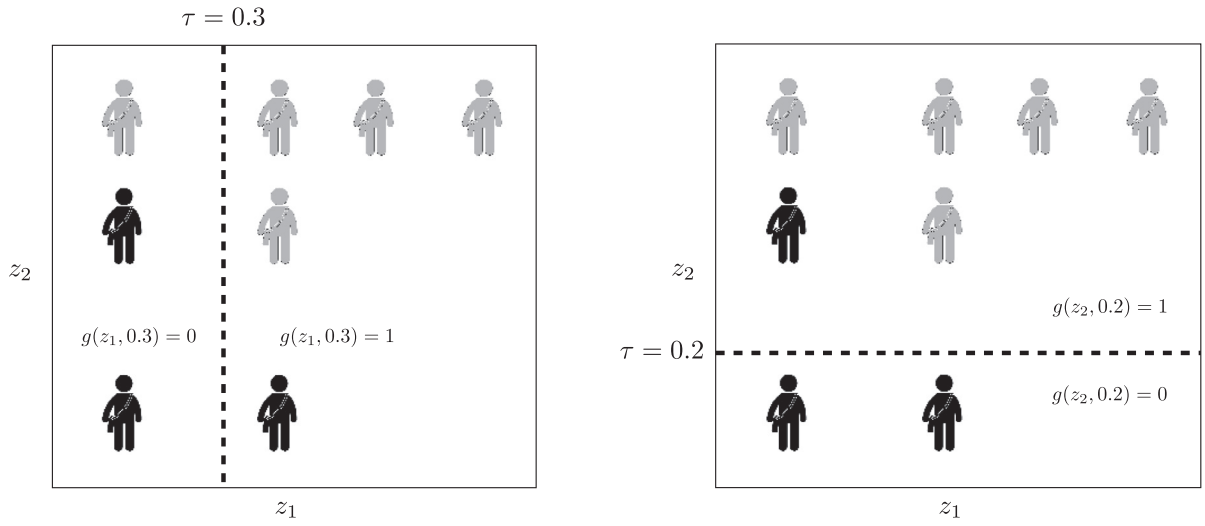

Fig. 3. Data for Example 2.4. Students colored in black have failed, grey students have passed the exam. Two metrics z_1 and z_2 are used. Left: maximum likelihood estimation of τ and corresponding values of $g(z, \tau)$ for the first feature z_1. Right: τ and $g(z, \tau)$ for the second feature z_2.

Given that the likelihoods of all the members of the ensemble have been computed, the Likelihood Weighted Average (LWA) ensemble of estimators is a natural choice for the votes v_f:

$$v_f = \frac{L(\hat{\theta}(z_{f1}, \ldots, z_{fm})|D)}{\sum_{g=1}^{F} L(\hat{\theta}(z_{g1}, \ldots, z_{gm})|D)} \tag{27}$$

It is remarked that this kind of ensemble it is easily extended to fuzzy data:

$$[\hat{\theta}(\tilde{\mu}_1, \ldots, \tilde{\mu}_{PM})]_\alpha = \left\{ \hat{\theta}(z_1, \ldots, z_{PM}) \,\middle|\, (z_1, \ldots, z_{PM}) \in [(\tilde{\mu}_1, \ldots, \tilde{\mu}_{PM})]_\alpha \right\}. \tag{28}$$

In the particular case that there are as many members in the ensemble as software metrics, and each of the ensemble members depends on a single software metric, anonymization functions depend on a single threshold:

$$g(z, \tau) = \begin{cases} 1 & z > \tau \\ 0 & z \leq \tau \end{cases} \tag{29}$$

thus the estimate of the pass rate produced by the kth member of the ensemble is

$$\hat{\theta}(z_k) = \frac{1 + \sum_{s=1}^{N} g(z_{ks}, \tau_k)}{N + 2} \tag{30}$$

and τ_k is chosen to maximize the likelihood $\hat{\theta}(z_k)$ on D, which is

$$L(\hat{\theta}(z_k)|D) = \prod_{s=1}^{N} L(y_s|\hat{\theta}(z_k)). \tag{31}$$

The LWA ensemble is

$$\hat{\theta}(z_1, \ldots, z_{PM}) = \frac{\sum_{k=1}^{PM} \hat{\theta}(z_k) L(\hat{\theta}(z_k)|D)}{\sum_{k=1}^{PM} L(\hat{\theta}(z_k)|D)} \tag{32}$$

The calculations needed to obtain a model in this case (so many members in the ensemble as software metrics) are detailed in the example that follows.

Example 2.4. A group of $N = 8$ students is considered. Five of them passed the exam, the other three failed: the true pass rate is $5/8 = 0.625$. A model of the pass rate of this course is built on the basis of two software metrics z_1 and z_2.

In Fig. 3 some properties of this group of students are illustrated. Grey students passed the course, and black students failed. The position of each student is given by the values of the two software metrics, i.e. each student is plotted at coordinates (z_{1s}, z_{2s}) for $s = 1, \ldots, 8$.

Let the first software metric be considered. The anonymization function $g(z_1, \tau)$ divides the plane into two different regions ($g = 0$ and $g = 1$) for each value of τ, and each division produces a different estimate $\hat{\theta}(z_1)$ according to Eq. (30). The value of τ that produces the maximum likelihood estimation of θ for the first software metric is $\tau = 0.3$, that leaves

5 students in the region where $g(z_1, 0.3) = 1$ and three students in the region where $g(z_1, 0.3) = 0$. The corresponding estimation of the pass rate is

$$\hat{\theta}(z_1) = \frac{1+5}{8+2} = 0.6 \tag{33}$$

and the likelihood of this estimate is

$$L(\hat{\theta}(z_1)) = 0.6^5 \cdot (1-0.6)^3 = 0.00497. \tag{34}$$

The case with the second software metric is plotted in the right part of the same figure. In this case the maximum likelihood estimate of θ is reached at $\tau = 0.2$, that is 6 students for which $g(z_2, 0.2) = 1$ and 2 students where $g(z_2, 0.2) = 0$. The corresponding estimation of the pass rate for the second metric and $\tau = 0.2$ is

$$\hat{\theta}(z_2) = \frac{1+6}{8+2} = 0.7 \tag{35}$$

and the likelihood of this estimate is

$$L(\hat{\theta}(z_2)) = 0.7^6 \cdot (1-0.7)^2 = 0.01058. \tag{36}$$

The LWA ensemble of $\hat{\theta}(z_1)$ and $\hat{\theta}(z_2)$ is a weighted mean of these values, where the most likely estimates have a higher weight. The learned model of the pass rate is

$$\hat{\theta}(z_1, z_2) = \frac{0.00497 \cdot \frac{1+\sum_{s=1}^{N} g(z_{1s}, 0.3)}{N+2} + 0.01058 \cdot \frac{1+\sum_{s=1}^{N} g(z_{2s}, 0.2)}{N+2}}{0.00497 + 0.01058} \tag{37}$$

whose evaluation on the training data is

$$\frac{0.00497 \cdot 0.6 + 0.01058 \cdot 0.7}{0.00497 + 0.01058} = 0.668. \tag{38}$$

3. Case study

Seventy three engineering students enrolled in a course of Computer Science Foundations (including the Python programming language) volunteered in this study. Their grades are computed as the weighted average of theoretical (one exam) and practice skills (two exams, at the beginning and end of each term). Programming assignments, multiple choice tests and surveys were also conducted but these were not taken into account in the final grading. The purpose of this empirical study is to predict the pass rate (that only depends on the mentioned exams) on the basis of the different software metrics that are measured on the programming assignments.

The students uploaded their assignments to a *Moodle* task [7]. Each assignment is the solution to one of the proposed exercises. Each student could upload an arbitrary number of source code files (some of them are different versions of earlier solutions to the same problem by the same student) or none at all. One separate task was available for each of the topic's chapters: Standard I/O and expressions, Conditionals, While loop, For loop, Functions and Lists. This work data has been collected during the first terms of 2014 and 2015. The experimental setup is as follows:

Experimental data		
	2014	2015
Total students	110	49
Volunteer students	44	29
Total source files	727	827
File I/O	148	76
Conditionals	212	123
While loop	108	248
For loop	114	98
Functions	107	175
Lists	38	107

23 software metrics were measured for each source file [1], thus the feature selection stage has to select the most relevant inputs among 138 different combinations of programming concept and software metric.

The study had two stages: first, in 2014 a pure AG problem was proposed, whose conclusions were published in [14]. Second, the work in this paper data from both years, 2014 and 2015, is used. On the one hand, the AG model developed in 2014 is used to perform an indirect estimation of the pass rates at 2015: the grades of the new students are predicted, and the fraction of predictions that are higher than 5 computed. On the other hand, a direct model is built with the same data from 2014 and their results are validated in 2015. The most relevant features and programming concepts of both the direct and the indirect approach are discussed, and the respective accuracies in the prediction of the pass rates of either approach compared.

Table 1
Relevant metrics for the indirect model (taken from [14]).

Programming concept	Description of the metric	Rank 99%	Rank 80%
Conditional	COCOMO SLOC	1 ± 0	11 ± 10
Conditional	Number of tokens	2 ± 0	8.5 ± 7.5
Conditional	Code ratio	4 ± 1	25 ± 24
File I/O	Number of characters	4 ± 1	11 ± 8
Conditional	Number of lines	7 ± 1	21 ± 19
Functions	Number of characters	4 ± 1	43 ± 37
Conditional	Number of keywords	7.5 ± 1.5	36 ± 33
Conditional	Number of comments	17 ± 6	48 ± 44
File I/O	Ratio of comments	17 ± 6	48 ± 44
File I/O	McCabe complexity	17 ± 9	30 ± 24
File I/O	Number of blocks	17 ± 9	31 ± 25

Table 2
Relevant metrics for the direct model (Fisher information).

Programming concept	Description of the metric
For loop	McCabe total complexity
For loop	McCabe complexity
Conditional	Number of logical lines of code
File I/O	Pylint invalid name
Functions	Number of logical lines of code
While loop	Halstead volume
While loop	Halstead estimated number of delivered bugs
Functions	Pylint unused argument

The 8 most relevant metrics for the indirect model (this is the subset for which the best model attained a minimum training error), and two α-cuts of their fuzzy ranks, are shown in Table 1 (taken from [14]; notice that there is a quadruple tie at position 8 hence the table contains 11 lines). The most relevant metrics for the direct model (the eight combinations of software metric/programming concept with a higher loss of Fisher information for any partition of size 2) are shown in Table 2. Only two programming concepts (Conditional and File I/O) made into the list of features for the indirect model. The list of features obtained by the Fisher information method involves 5 programming concepts, and there is a certain agreement about the most useful metrics (number of logical lines/tokens, complexity measures), but the importance of the assignments of type "Conditional" is reduced, while the assignments of "Functions" and "Loops" gain importance.

3.1. Features for AG and pass rate estimation

The AG problem was solved with Linear (LIN), Support Vector Machines (SVM), Neural Networks (NN), Regression Trees (RT), Random Forest (RF) and NMIC models [20] for two sets of features. The first set is obtained with the fuzzy generalization of the Random Forest proposed in [14]. The second set is attained by means of the Fisher information-based method proposed in this paper.

Actual data was used for validating the solutions to the AG problem: models were learnt with the grades obtained by the students at 2014, and tested with the grades obtained by a different group of students of the same subject at 2015. The main idea is to test whether the pass rate in 2015 can be approximated by a model learnt from data in 2014. In addition to this practical validation, a sensitivity analysis is performed to measure the influence of the dimension of the input space in the accuracy of the models.

The results of the validation of the AG problem with actual data are shown in the first and second columns of Table 3. The Medium Squared Errors (MSE) of the different models are shown for the best subsets of 8 features that were found by both the Fuzzy Random Forest-based selection and the Fisher selection. These MSE values were found to be too high for the automatic assessment to be useful for practical purposes, i.e. the accuracy of the best predictor is not much different than that of a trivial predictor.

The sensitivity analysis has three different aspects:

1. The third column of Table 3 contains models that have been learnt without feature selection techniques. It serves as a reference to measure the amount of information (if any) that is lost when the number of features is reduced from 138 features to 8.
2. A bootstrap estimate of the standard deviation of each method has been computed. To obtain these values, each of the models was trained on 100 resamples with replacement of the training datasets (year 2014). Numbers in parentheses on Table 3 are the mentioned estimates. These numbers measure the statistical relevance of the differences among the MSE values (the larger the standard deviation, the less relevant is the difference between the MSEs).

Table 3
Test results of the indirect models for the sets of features in [14] and the Fisher information-based selection. The last column comprises models learned from the complete set of features (without selection). Numbers in parentheses are bootstrap estimates of the standard deviation of the results.

	MSE regression 2015		
Method	Fuzzy random forest	Fisher	Without selection
LIN	6.93 (2.09)	4.28 (1.30)	33.20 (13.97)
SVM	3.07 (0.20)	2.96 (0.66)	3.13 (0.37)
NN	3.67 (0.66)	3.25 (0.93)	6.44 (1.97)
RT	5.11 (1.10)	4.65 (1.01)	6.21 (2.29)
RF	3.07 (0.27)	3.75 (0.81)	3.36 (0.46)
NMIC	3.06 (0.24)	3.20 (0.20)	2.89 (0.43)

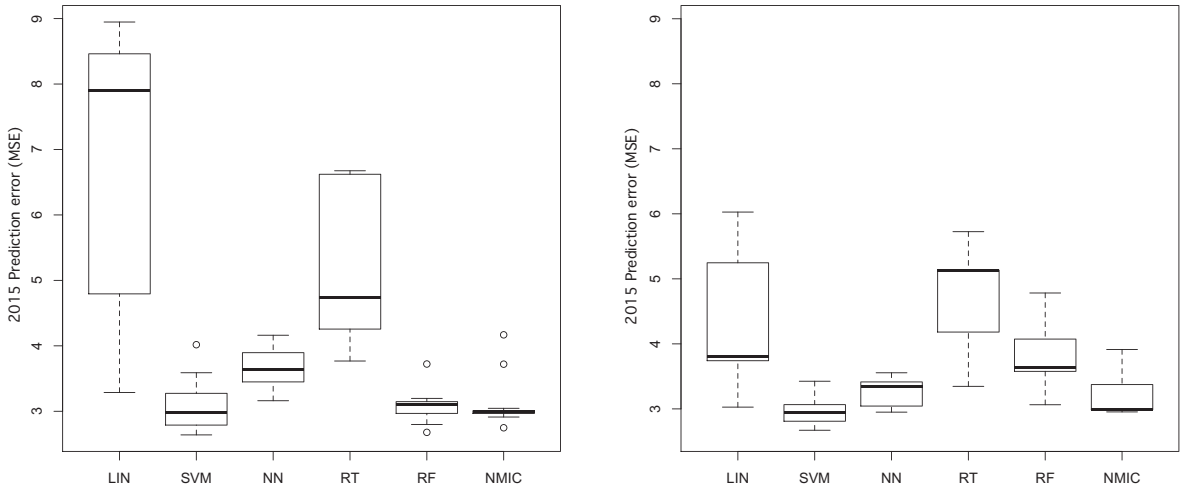

Fig. 4. Left part: Boxplot showing the statistical differences between the test error for the AG problem in 2015 of the combination of Neural Networks (NN), Support Vector Machines (SVM), Regression Trees (RT), Random Forests (RF) and NMIC. Right part: Same AG problem, features selected by means of the fuzzy Fisher information-based method. The sets of features obtained by the second method does not improve the AG problem, however the accuracy of Fisher's method will be significantly better when the pass rate is estimated with a direct method, see Fig. 6).

In this respect, observe that standard deviations for fuzzy random forest-based selections depend on the different lists of input variables that are generated for each execution of the algorithm, but Fisher selections are deterministic and depend only on the resamplings of the train and test sets.
The bootstrap estimate of the standard deviation has also been computed for the models that depend on the whole set of variables. In this case, the variance is higher for those methods that overtrain (Linear, NN, RT), and small for those cases (SVM, RF, NMIC) where the learning method includes an implicit downweight of the irrelevant inputs.

3. Fig. 4 contains a graphical assessment of the statistical relevance of the differences between the MSE of models learnt with input subsets of different dimensions (from 1 to 19 input variables) for the two feature selection methods. LIN and RT have large differences between the best and the worst error, meaning that the number of inputs of the model is important. Other methods such as SVM or NMIC are less affected by the dimension of the input.

The main practical conclusion that can be drawn from this experimentation is that the subsets of features found by the fuzzy extension of the Random Forest method proposed in [14] and the Fisher information-based method are not significantly different for the AG problem. However, this assert is no longer true if the direct approach is used. A significant difference in the accuracies of the predictions of the pass rate exists between Fuzzy Random Forest and Fisher's methods, as shown by the next group of experiments.

3.2. Direct vs indirect approaches for estimating the pass rate

The same methods seen before for estimating the grades will be applied again for both sets of features, but the pass rate will be estimated in an indirect approach. Two different criteria are considered:

1. Prediction error: Absolute value of the difference between the true pass rate in 2015 and the predicted value.

Table 4
Delta and prediction errors of all the indirect models and the direct estimator. The last column comprises models learned from the complete set of features (without selection). Numbers in parentheses are bootstrap estimates of the standard deviation of the results, or the standard deviation of the centerpoints of the interval-valued errors.

	Fisher		Fuzzy random forest		Without selection	
Method	P. Error 2015	Delta 2014–2015	P. Error 2015	Delta 2014–2015	P. Error 2015	Delta 2014–2015
LIN	0.059 (0.052)	0.060 (0.072)	0.067 (0.045)	0.094 (0.063)	0.519 (0.306)	0.475 (0.304)
SVM	0.031 (0.016)	0.075 (0.021)	0.076 (0.001)	0.062 (0.026)	0.077 (0.001)	0.063 (0.001)
NN	0.026 (0.044)	0.071 (0.076)	0.055 (0.026)	0.070 (0.067)	0.061 (0.045)	0.039 (0.032)
RT	0.075 (0.075)	0.079 (0.042)	0.075 (0.149)	0.054 (0.119)	0.150 (0.177)	0.138 (0.120)
RF	0.022 (0.024)	0.067 (0.050)	0.077 (0.026)	0.024 (0.024)	0.081 (0.012)	0.100 (0.068)
NMIC	0.060 (0.012)	0.077 (0.017)	0.073 (0.032)	0.064 (0.022)	0.069 (0.043)	0.062 (0.001)
LWA	**[0.009,0.014]** (0.005)	**[0.013,0.019]** (0.014)	–	–	–	–

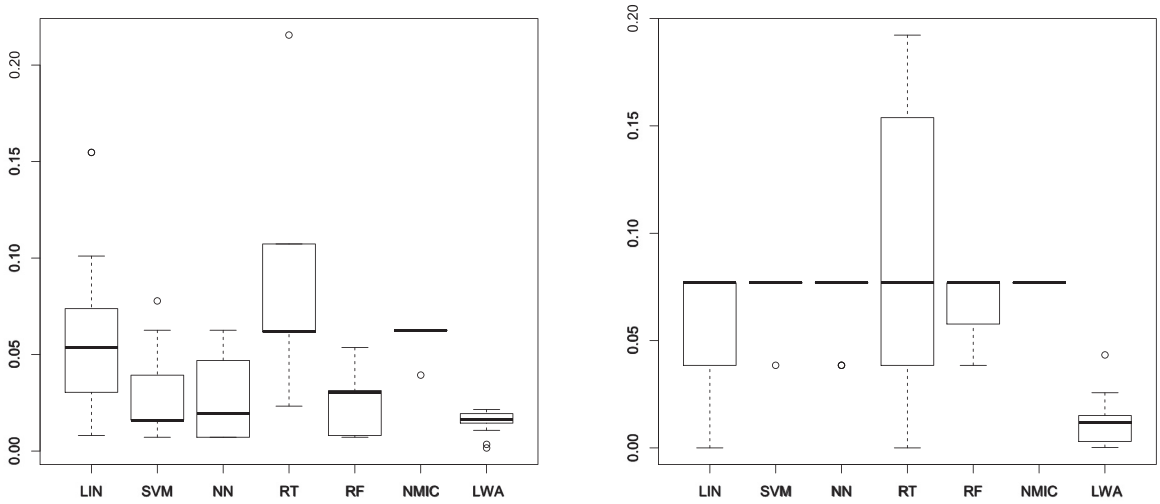

Fig. 5. Dispersion of prediction error (left) and delta (right) for all the considered methods.

2. Delta error: Absolute value of the difference between the predicted *increments* of the pass rate between 2014 and 2015.

For instance, if the true pass rate was 0.8 in 2014 and 0.9 in 2015, and a model predicts a pass rate of 0.85 in 2014 and 0.87 in 2015, the prediction error is $|0.9 - 0.87| = 0.03$ and the delta error is $|(0.9 - 0.8) - (0.87 - 0.85)| = 0.08$, i.e. the model predicted an increment of 0.02 of the pass rate and the actual increment was 0.1. The delta error measures the sensitivity of the estimator, being high in models with constant or almost constant output.

In Table 4 both magnitudes are summarized. The row labeled 'LWA' is the direct estimation (Likelihood Weighted Average for fuzzy data) described in the preceding section. Additional results are displayed in Figs. 5 and Fig. 6. In Fig. 5 two sets of boxplots are provided that show the dispersion of the delta and the prediction error for all the considered methods. In the left part of Fig. 6, a Pareto front showing the two criteria is shown. The right part of the same Fig. 6 summarizes the test errors of different ensembles of sizes between 1 and 19, making use of the most important variables according to the fuzzy extension of the observed Fisher information-based method (red line). The boxplots in the same graph show the dispersion of the test errors of the indirect methods when applied to subsets of features of the same size, when chosen according to the method in [14]. Observe that the accuracy of the ensemble is always better than that of any of the indirect methods, thus it can be concluded that (a) the set of features that produce the best direct or indirect model are not the same, and (b) the accuracy of both is also different. Lastly, let be remarked that boostrap estimates of the standard deviation of the pass rates have been added to Table 4, applying the same methodology that was already explained in this section. The standard deviation of the centerpoints of all interval-valued estimates has been used when needed.

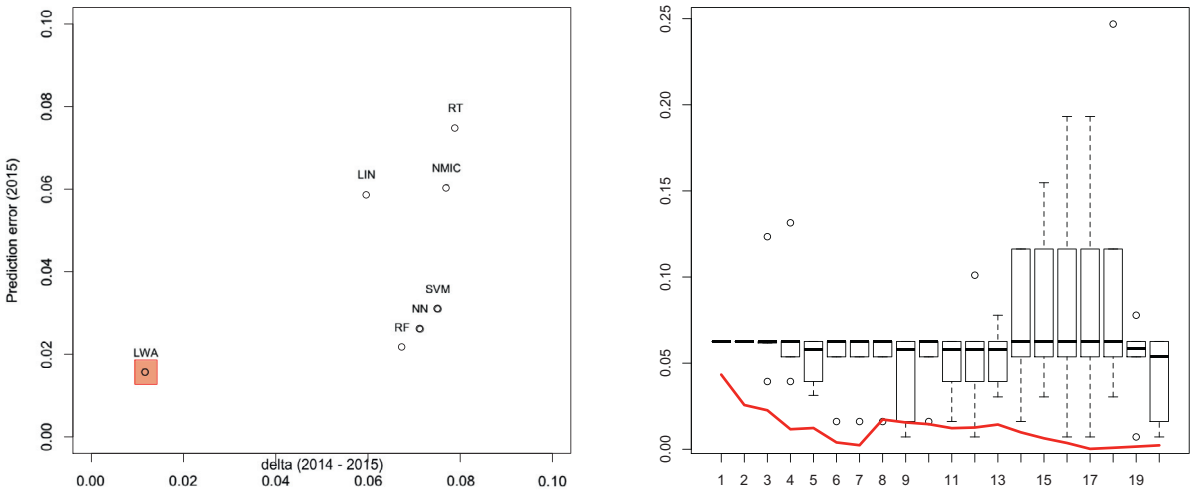

Fig. 6. Left part: Pareto front showing prediction error and delta (or the support of the fuzzy error in the direct estimation). LWA in combination with the Fisher method outperforms every alternative for both criteria to be met. Right part: Test errors of different ensembles of sizes between 1 and 19, formed by the most important variables according to the Fisher method (red line) and boxplots of the dispersion of the test errors of the indirect methods when applied to subsets of features of the same size, chosen according to [14]. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4. Concluding remarks and future work

A strategy for finding the most informative code metrics and predicting the pass rate of a course is presented. An extension of the observed Fisher information for vague data was used to assess the relevance of a feature. Two different procedures for doing this prediction, so called “direct” and “indirect” approaches, were compared. In the indirect approach, an AG model is used to predict the grade of each student and subsequently the fraction of predictions that pass the cut-off threshold is computed. In the direct approach, a model is defined that outputs the pass rate without an intermediate AG stage. The direct model was defined as the average of the outcomes of a function, obtained with machine learning techniques, that is applied to the vector of features that characterizes each student. Two questions were raised: (a) whether the accuracies of direct and indirect approaches are the same, and (b) whether the most relevant features are the same for either case. A real-world case was shown where the answer was negative to both questions.

From a methodological point of view, an ensemble of predictors has been used to predict the pass rate. Each ensemble contains so many predictors as input variables, and these are weighted according to their likelihoods. The uncertainty in the knowledge about the value of the software metrics is propagated through the model and influences the predictors and their weights. In future works, different kinds of ensembles will be explored where each member depends on more than one variable. In particular, random ferns [16], along with its extension to imprecise data, will be studied. Also, the simultaneous estimation of different parameters of the course than the pass rate will be tackled: the estimation of multidimensional parameters has already been introduced in this paper but the experimental analysis has been restricted to scalar parameters.

Acknowledgments

This work was supported in part by the Spanish Ministry of Science and Innovation (MICINN) and the Regional Ministry of the Principality of Asturias under Grants TIN2014-56967-R and FC-15-GRUPIN14-073.

References

[1] M. Abdellatief, et al., A mapping study to investigate component-based software system metrics, J. Syst. Softw. 86 (3) (2013) 587–603.
[2] A. Abran, Software Metrics and Software Metrology, Wiley-IEEE Computer Society Press, 2010.
[3] D. Agrawal, C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: PODS '01 Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2001. 247–255.
[4] D. Arnow, O. Barshay, On-line programming examinations using web to teach, in: Proceedings of the 4th Annual SIGCSE/SIGCUE ITiCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '99), 1999. 21–24.
[5] G. Bortolan, R. Degani, A review of some methods for ranking fuzzy subsets, Fuzzy Sets Syst. 15 (1) (1985).
[6] B. Cheang, A. Kurnia, A. Lim, W. Oon, On automated grading of programming assignments in an academic institution, Comput. Edu. 41 (2) (2003) 121–131.

[7] J. Cole, Using Moodle: Teaching with the Popular Open Source Course Management System, US: O'Reilly Media, Inc., 2005.
[8] I. Couso, L. Sanchez, Higher order models for fuzzy random variables, Fuzzy Sets Syst. 159 (3) (2008) 237–258.
[9] D. Dubois, H. Prade, Fuzzy sets - a convenient fiction for modeling vagueness and possibility, IEEE Trans. Fuzzy Syst. 2 (1) (1994) 16–21.
[10] F. Jurado, M. Redondo, M. Ortega, Using fuzzy logic applied to software metrics and test cases to assess programming assignments and give advice, J. Netw. Comput. Appl. 35 (2) (2012) 695–712.
[11] A. Kurnia, A. Lim, B. Cheang, Online judge, Comput. Edu. 36 (4) (2001) 299–315.
[12] D.S. McNamara, S.A. Crossley, R.D. Roscoe, L.K. Allen, J. Dai, A hierarchical classification approach to automated essay scoring, Assess. Writ. 23 (2015) 35–59.
[13] K.P. Murphy, Machine Learning : A Probabilistic Perspective, MIT Press, 2012.
[14] J. Otero, M.R. Suárez, A. Palacios, I. Couso, L. Sánchez, Selecting the most informative inputs in modelling problems with vague data applied to the search of informative code metrics for continuous assessment in computer science online courses, Lecture Notes Comput. Sci. 8536 (2014) 299–308.
[15] J. Otero, A.M. Palacios, M.R. Suárez, L.A. Junco, I. Couso, L. Sánchez, A procedure for extending input selection algorithms to low quality data in modelling problems with application to the automatic grading of uploaded assignments, Sci. World J. (2014) 1–10.
[16] M. Özuysal, M. Calonder, V. Lepetit, P.F. Fua, Keypoint recognition using random ferns, IEEE Trans. Pattern Anal. Mach. Intell. 32 (3) (2010) 448–461.
[17] E.B. Page, New computer grading of student prose, using modern concepts and software, J. Exp. Edu. 62 (2) (1994) 127–142.
[18] K.A. Reek, A software infrastructure to support introductory computer science courses, in: K.J. Klee (Ed.), Proceedings of the Twenty-Seventh SIGCSE Technical Symposium on Computer Science Education (SIGCSE '96), ACM, New York, NY, USA, 1996, pp. 125–129.
[19] Y. Saeys, T. Abeel, Y. Peer, Robust feature selection using ensemble feature selection techniques, Lecture Notes Comput. Sci., 5212, W. Daelemans, B. Goethals, K. Morik Springer Berlin Heidelberg, 2008, 313–325.
[20] L. Sanchez, J. Otero, I. Couso, Obtaining linguistic fuzzy rule-based regression models from imprecise data with multiobjective genetic algorithms, Soft Comput. vol. 13 (no. 5) (2008) 467–479.
[21] L. Sanchez, I. Couso, J. Casillas, Genetic learning of fuzzy rules on low quality data, Fuzzy Sets Syst. 160 (17) (2009) 2524–2552.
[22] M. D., Shermis, J. C., Burstein (Eds.). (2003). Automated essay scoring: A cross-disciplinary perspective. Routledge.
[23] M. Vujosevic-Janicica, M. Nikolica, D. Tosica, V. Kuncak, Software verification and graph similarity for automated evaluation of students assignments, Inf. Softw. Technol. Volume 55 (Issue 6) (2013) 1004–1016.
[24] T. Wang, X. Su, P. Ma, Y. Wang, K. Wang, Ability-training-oriented automated assessment in introductory programming course, Comput. Edu. 56 (1) (2011) 220–226.