




# Artificial Intelligence models for assessing the evaluation process of complex student projects

Jose Divasón , Francisco Javier Martínez-de-Pisón , Ana Romero , and Eduardo Sáenz-de-Cabezón 

**Abstract**—The evaluation of student projects is a difficult task, especially when they involve both a technical and a creative component. We propose an AI-based methodology to help in the evaluation of complex projects in Engineering and Computer Science courses. This methodology is intended to evaluate the assessment process itself allowing to analyze the influence of each variable in the final grade, to discover possible biases, inconsistencies and discrepancies, and to generate appropriate rubrics that help to avoid them. As an example of its application, we consider the evaluation of the projects submitted in an undergraduate introductory course on Computer Science. Using data collected from the evaluation during five academic years, we follow the proposed methodology to create AI models and analyze the main variables which are involved in the assessment of the projects. The proposed methodology can be applied to other courses and degrees, where both technical and creative components are considered to evaluate the projects.

**Index Terms**—automated grading, computer science, evolutionary algorithms, machine learning, rubric

## I. INTRODUCTION

**I**N many computer and technical courses of different undergraduate and master's degrees, a part of the evaluation is based on projects and assignments. The rise of this teaching and assessment methodology, in particular project-based learning (PBL), is mainly due to its efficiency in the development of professional skills and transferable competences in students [1]. PBL has been defined as “an active student-centered form of instruction which is characterized by students' autonomy, constructive investigations, goal-setting, collaboration, communication and reflection within real-world practices” [2, p. 1]. PBL “is based on the constructivist finding that students gain a

deeper understanding of material when they actively construct their understandings by working with and using ideas in realworld contexts” [3, p. 275] and relies on three principles: first, the learning process is context-specific; second, learners are actively involved in the process; third, they achieve their goals through social learning and the sharing of knowledge and understanding. PBL has strong connections to inquiry-based learning [4] and problem-based learning among others [5]. A main distinct characteristic of PBL with respect to problem based learning and others is that PBL typically leads to the creation of a final product, such as a written or oral report, or a design or model [3], [6]. In PBL, the emphasis is on the construction of new knowledge, and on the application or integration of previously acquired knowledge, rather than on the actual acquisition of it [7].

PBL can promote self-regulated learning, intrinsic motivation and students' conceptual knowledge by being part of a systematic process of goal-setting, planning, documenting and reflecting on learning [8], [9]. However, the use of this methodology raises various difficulties that have certain peculiarities in fundamental Computer Science courses. These particular difficulties include the challenge of developing basic skills and attitudes which are important for effective learning in later courses. Skills like team work, good coding habits, documentation habits or broad problem solving skills, rather than purely technical ones, have been identified as difficult to teach and evaluate in courses on foundations of Computer Science [10].

Among the difficulties and challenges encountered in relation to project-based methods are those related to their evaluation [11], [12]. The evaluation of projects of both technical and creative nature is diffi-

cult for several reasons. One is the intrinsic difficulty of evaluating the creative part of the project, for it is prone to subjectivity of the evaluator. Another one is due to the open-ended nature of the tasks involved in PBL projects, which do not have a unique solution. Finally, there is the complexity of the evaluation of transferable competences and skills acquired in the development of the project [13]. The tools for such an evaluation include scoring rubrics, Likert scale assessment forms, the use of check-lists, or the actual performance of students' artifacts related to the initial goals of the project [14], [15]. But in general, there is a lack of studies on the evaluation of the students' learning processes and the quality of students' artifacts, and on the quality of the measurement instruments [6]. Our proposal intends to be a contribution in this direction.

Fairness in assessment has received attention in the recent years. Most of the research has been centered on the assessment process itself, but literature beyond assessment provides a more comprehensive conception of assessment fairness at the classroom level [16]. Threats to assessment fairness include bias, discrepancies and inconsistencies. Bias is a prejudice for or against a person or group, considered to be unfair and that tends to affect the evaluation of her contribution to the project. Discrepancies are unexpected differences among the evaluation outcome of different teachers for the same project. Inconsistencies are unexpected differences in the evaluation of similar quality projects by the same teacher. All these are usually very difficult to detect by the involved teachers themselves [17]. The detection of discrepancies, inconsistencies and biases in the evaluation is carried out in the literature in two ways: exogenous (based on external measures of student performance) and endogenous (based on their own grading), although the latter has shown to be more adequate [18]. On the other hand, different methodologies have been proposed to detect discrepancies between teachers in the evaluation of projects in Engineering and Computer Science education [19], [20]. There is not much literature on whether the choice of evaluation schemes themselves has an impact on the fairness of the assessment. Some research however has been done on the difficulties of particular aspects of some evaluation schemes, such as summative evaluation [21], achievement tests [22], team work assessment [23] and student peer evaluation [24].

In summary, the evaluation of PBL projects is a

complicated task due to several aspects of the learning process that are involved, including team work, transferable skills and open-ended tasks. The assessment of this kind of projects can partly be automated, although needs also human input. This is prone to bias and inconsistencies, and also discrepancies in case several teachers are involved in the grading.

Our contribution consists of a methodology for assessing student project work. In particular, we present an instantiation of it which is based on machine learning (ML) to discover hidden knowledge in the evaluation process to identify discrepancies, inconsistencies and biases in the grading of an IT project. This hidden knowledge is used to enhance tools used in the evaluation process. We study the application of this methodology in the evaluation of a project in the first semester of the first year of the degrees in Software Engineering and Mathematics at the University of La Rioja (Spain). The project consists in the development of a website made up of several pages, it is carried out by small groups (of two or three members) and is evaluated by the team of teachers of the course so that each project receives a grade by only one of the teachers involved. The proposed methodology is focused on optimizing and explaining ML models to identify the variables that influence the final grade of the projects. In particular, we can detect discrepancies between teachers by observing the variables that correspond to each of the teachers involved in the evaluation process, so that we identify projects that are equivalent with respect to the evaluation criteria, but differ in the final grade depending on the teacher who marks them. Observing variables such as Gender, the Degree that each student is studying or others, we can detect biases in the evaluation.

After this introduction, Section II presents multi-objective optimization methods and details GAparsimony, a ML technique to search parsimonious models using genetic algorithms that optimize the feature selection and algorithm's hyper-parameters. That section also introduces the concept of explainable ML and concretely SHAP, a method for explaining the predictions of ML models. Section III presents the proposed methodology. In Section IV we apply this methodology and explain the academic context. Finally, Section V discusses the results and in Section VI we present the conclusions and further work.

## II. PRELIMINARIES

### A. Multi-objective optimization methods

One of the most important steps in ML is the feature selection and the hyper-parameter optimization for finding optimal models. A multi-objective optimization task deals with the common need of optimizing several (and commonly conflicting) objective functions simultaneously, such as performance and complexity. This is a more difficult task than single-objective optimization problems, since usually there is no single solution that simultaneously optimizes several objectives, being a wide research area with a multitude of available methods [25].

In this study, multi-objective optimization methods are proposed to simultaneously optimize the performance and the complexity of ML models. As an instance, it is proposed the use of *GAparsimony* [26], a specific multi-objective optimization tool that allows a dominant primary objective (performance improvement) and a secondary objective (parsimony, i.e., reduction of model complexity).

The search for parsimonious (low complexity) models is also one of the current challenges in the field of machine learning. It follows the principle of parsimony, also called “Occam’s razor rule”: all other factors being equal, the simplest explanation is usually the most likely. Applied to the world of machine learning, among models with similar performance it is recommended to choose those with less complexity as they tend to be better generalizers of the problem. In addition, they are easier to understand and more robust against disturbances in their inputs. The mechanisms used within ML algorithms, such as regularization or feature selection, are focused in this regard.

*GAparsimony* performs a search for low-complexity models using genetic algorithms. The ultimate goal is to obtain high-performance and low-complexity models through the use of feature selection, adjustment of algorithm training hyper-parameters, and parsimony-based selection. The last one is what differentiates *GAparsimony* from other similar multi-objective methods. Indeed, previous experiments with other common techniques have shown that the simultaneous optimization of both performance and complexity usually produces an evolution of solutions that are not optimal [27]. In *GAparsimony*, the selection of the best individuals or solutions in each generation is

carried out following a parsimony search principle that consists of two consecutive steps: a pre-selection of the most accurate models and, among those with a similar cost, a promotion to higher positions of those with less complexity. The *GAparsimony* package [28] is available in both R and Python languages. To perform GA optimization with *GAparsimony* it is necessary to define the chromosomes of each individual model to be trained with the corresponding ML algorithm. A chromosome is defined by a combination of the values assigned to the algorithm’s training hyper-parameters and a vector that indicates which are the selected input features for that individual. In particular, each individual  $i$  of each generation  $g$  is defined with a chromosome  $\lambda_g^i$  formed by the concatenation of two vectors  $P$  and  $Q$ , where  $P$  corresponds to the algorithm’s training hyper-parameters, and  $Q$  represents a vector of probabilities used in the selection of the features so that the variable  $j$  will be included in the model if  $q_j \geq 0.5$ . As the objective function (denoted as  $J$ ), the cross-validation RMSE,  $J = RMSE_{val}$  was used. Finally, the complexity of the model was defined as the number of selected features  $N_{FS}$ . This measure of complexity has proven to be very effective in the past [29]. *GAparsimony* flowchart is shown in Figure 1. A first generation is created with a Latin Hypercube Sampling (LHS) in order to uniformly distribute the first individuals in the search space. After the calculation of each generation, *GAparsimony* sorts models by  $J$ . In a second step, individuals with similar cost are re-ordered by promoting the less complex solutions to the top positions. In this case, two individuals are considered similar if the absolute difference of their  $J$ s is less than a predefined *ReRank* parameter. After selecting the best individuals in each generation, *GAparsimony* carried out the classic processes of crossing the chromosomes of the best individuals to create the next generation of individuals; as well as the mutation of chromosomes in order to create more diversity of solutions in later generations. The procedure is repeated until the maximum number of generations  $G$  is reached.

### B. Interpretable and explainable ML models

The rise of ML is also increasing the interest in obtaining understandable outputs. An interpretable AI model is a model that provides understandable outputs

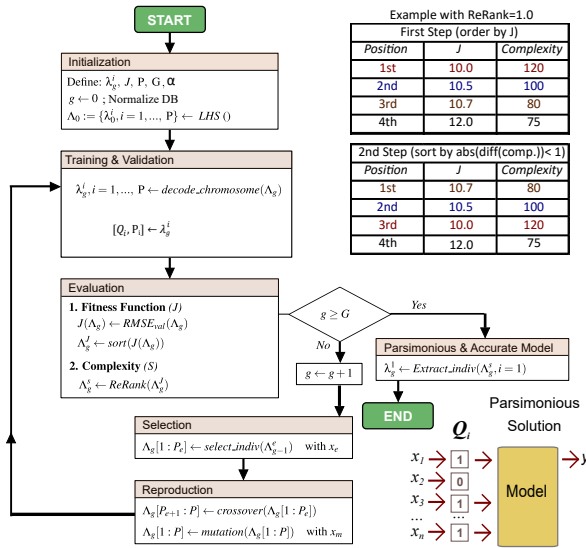


Fig. 1. Flowchart of the *GParsimony* methodology with an example of the two order steps with  $ReRank = 1.0$

that allow humans to know how it came to specific predictions. In contrast, explainable ML is where other ML techniques are needed to explain a black-box model [30]. Normally, classical ML models are inherently interpretable (like linear regression), but offer lower predictive power. On the contrary, black-box models (like ANN) usually offer high performance, but it is very difficult, if not impossible at all, to know exactly how their outcomes are obtained.

SHAP (SHapley Additive exPlanations) [31] is a novel method to explain the output of any machine learning model, including complex ones. It is based on Shapley values, a well-known solution concept in cooperative game theory. Given a model, the goal of SHAP is to explain the prediction of an instance by computing the contribution of each feature to the prediction. The idea behind the SHAP computations is relatively simple: it tries to quantify the contribution that each player (each feature) brings to the game (a single prediction). Shapley values are based on the idea that the outcome of each possible combination of features should be considered to determine the importance of a single feature. First, the power set  $F$  of all possible combinations of features is built. Shapley values are then computed for each feature. To do it, a model is trained with that feature present.

Let us denote the set of some features  $S$ , where the  $i$ -th feature, denoted simply as  $i$ , is included, i.e.,  $i \in S$ . Another model is also trained without that feature, i.e., over the set of features  $S - \{i\}$ . Then, predictions of the two models are compared, and this is repeated for all possible subsets  $S' \subseteq F - \{i\}$ . Then, the Shapley value for the  $i$ -th feature is computed as a weighted average of all possible differences.

Finally, the SHAP formula is presented as:

$$g(z') = \phi_0 + \sum_{i=1}^M \phi_i z'_i$$

where  $g$  is the explanation model,  $z' \in \{0, 1\}^M$  is the coalition vector (the vector that states which features have been selected in a subset of the power set),  $M$  is the number of features and  $\phi_i$  is the Shapley value for the  $i$ -th feature. This way, the formula uses Shapley values as an additive feature attribution method. Since the exact computation of SHAP values is challenging if the number of features is high, in practice their computation is approximated by means of various sophisticated techniques [31].

### III. METHODOLOGY

We propose the following seven-step methodology:

- 1) Identification of possible variables that could influence the grade
- 2) Data extraction
- 3) Preprocessing and normalization
- 4) (Optional) Basic ML training and optimization
- 5) Selection of one or several ML models and perform feature selection and hyper-parameter optimization.
- 6) Obtain the influence of the selected variables on the model's output.
- 7) Analyze the results in order to:
  - a) Generate a rubric or refine an existing one
  - b) Detect possible discrepancies, inconsistencies and biases

#### A. Identification of possible variables that could influence the grade

This step is better carried out by experienced teachers in both the course and the evaluation of the projects, in co-design with an ML expert. This could help the explainability of this approach [32]. It

is recommended to classify the variables into three groups:

- Technical variables: those that can be clearly quantified in an unequivocal and unbiased manner. For instance, if some requirement is satisfied, or the size of the deliverables.
- Style variables: those more subjective and related to the creative part of the project. For instance, appearance or readability.
- Context-based variables: those about the context of the student, such as gender, the degree she/he is studying, or the teacher who assigned (and/or graded) the task.

Any variable that could introduce bias should be added to the context-based group. Style variables can alert on possible inconsistencies and discrepancies, and context variables can alert on discrepancies and bias. This separation into groups help us to discover biases and discrepancies, being a simpler methodology than other methods that have been proposed [19].

The identification of variables should be done with respect to the requirements of the projects and their goals [32]. That is, teachers must identify variables (as many as possible) that could influence the evaluation of each project. Some variables could be encoded as either binary or numeric, but numeric variables are preferable, since they provide more information about the quality and evaluation than binary ones.

The identification of variables can be done in different ways, as it is very context-dependent. However, we recommend separating this process into several parts. On the one hand, the criteria or essential elements involved in the quality of the student's work should be identified and included. This part of the process is relatively similar to the identification of observable elements for the design of a rubric, and therefore one can follow some of the strategies presented in the literature, such as the first three steps of the methodology by C. A. Mertler's [33]:

- 1) Re-examine the learning objectives to be addressed by the task.
- 2) Identify specific observable attributes that one wants to see (as well as those one does not want to see) your students demonstrate in their product, process, or performance.
- 3) Brainstorm characteristics that describe each attribute.

Such observable characteristics will be the variables. Note that, in programming contexts, one can draw inspiration by the well-known Google Style Guides for each language [34] to identify some of the style variables. On the other hand, any other variable that is believed to have had an influence on the correction in previous years should also be included. Teachers with experience in the evaluation in previous years are important for this task. It is also desirable that ML experts guide teachers in this process.

Once this is done, all the variables should be grouped into technical and style variables. Finally, context variables, such as teacher and grade, should always be added, even if they are not thought to have had any influence. All the steps can be carried out by teachers separately, and finally discussing together their results to unify the identification of variables. In this step of unifying is where a ML expert can also help and bring knowledge that may also help with the explainability of the approach.

### B. Data extraction

Once the variables are identified, one has to create a database with the information of previous courses. Data extraction for technical and contextual variables should be automatized by means of scripts that analyze the projects of previous courses. Style variables are harder to measure. At least two experienced teachers of the course should analyze independently each project and agree a score for each variable. Preferably, the score should have values in a 4-point Likert scale [35], [36]. This avoids the neutral term [15], provides enough information for the AI models and eases the evaluation by the teachers (specially in its subjective parts) compared to a Likert scale with more points or other rating scales. The more data are extracted, the better. The minimum number of students will depend on the type of the course, projects, number of identified variables and so on. The final grade of each project must be added to the database, since this will be the target variable for the ML models.

### C. Preprocessing and normalization

In order to homogeneously approach the use of various statistical and ML algorithms, standard ML techniques must be applied to the data. This step is important, since ML algorithms require meaningful and

manageable data to correctly operate and to provide useful knowledge, predictions or descriptions [37]. Some examples of common preprocessing steps are the elimination of erroneous samples, handling missing values, normalization of the numerical variables using *z-Score* and binarization of categorical variables.

#### *D. (Optional) Basic ML training and optimization*

The goal is to train models based on the database to estimate the final grade of the project from the variables that were identified in the first step. The use of existing libraries in R or Python is encouraged, since both languages provide enough tools that ease this process and are widely used for this purpose. Both classical algorithms (like RIDGE, KNN, SVR and decision trees) and more advanced ones (like ANN and XGBoost) should be chosen for a basic ML training. Indeed, any ML model suitable for a regression problem can be employed.

The models should be fitted by grid search to optimize their hyper-parameters. To homogenize the adjustment and validation process of the various regression models, the square root of the mean square error (RMSE) must be selected as metric cost. The reason for choosing this metric is to avoid extreme cases where the prediction fails too much, that is, the use of RMSE penalizes estimates that are further from the real value than if we used other metrics such as the mean absolute error (MAE). The relation between RMSE and MAE is also important, for a RMSE value much higher than the MAE, indicates that the regression model has a high number of errors that are much larger than their average. Also, cross-validation is required to ensure robustness of the estimates.

This task is optional, but helps to estimate the accuracy and complexity of the ML models, which is important since some of the subsequent steps of the methodology are costly in terms of computing effort. Thus, at least one ML model with good RMSE and MAE, and a reasonable training time, should be chosen. It is possible that none of the models obtains an acceptable RMSE and MAE. This would suggest that either the initial variable selection is not adequate or the grid-search didn't provide a good estimate. There are three possibilities to proceed:

- 1) Rethink the initial variable selection (Step III-A).

- 2) Perform a more robust technique for hyper-parameter selection, not just a grid-search. Also, try using more complex models.
- 3) In case that a review of Step III-A does not work and/or a robust hyper-parameter selection is not feasible, one can proceed with the methodology using several models to get a better picture.

#### *E. Selection of one or several ML models and perform feature selection and hyper-parameter optimization*

The goal of this step is to obtain accurate but simpler models (using only a subset of the variables identified in the first step and optimized hyper-parameters). Simpler models provide more robust predictions, since they minimize the collinearity between variables and prevent overfitting. There exist many feature selection techniques (filter methods, wrapper methods, and embedding methods). State-of-the-art feature selection works [38] include bandit-based algorithms, such as halving [39] and Hyperband [40]. Well-studied meta-heuristics (such Particle Swarm Optimization [41]) are also valid for this step; indeed, this step can be performed with any technique that allows a robust feature selection. As a particular instance of this step of the methodology, it is proposed the use of GAparsimony, which is available in both Python and R. This technique provides accurate parsimonious models by combining feature selection (FS), model hyper-parameter optimization (HO), and parsimonious model selection (PMS). Using GAparsimony, the variables that are selected by the best model in the evolutionary process would be the ones that influence the rubric. Usually, this step is a costly process in terms of computing effort. For this reason, at least one ML model (with good RMSE and MAE) should be selected, but also in balance with the performance. The previous step would help for this purpose. Large databases could require simpler algorithms (or higher computational resources).

#### *F. Obtain information about the selected features in the best models and their influence.*

The previous step helps detecting which variables have influence, but not how much. To measure this influence one could analyze the weights of each variable, i.e., their importance. Some algorithms have a built-in feature importance (interpretable models), such as

many classical models. However, if a black-box model is used (like ANN), more advanced techniques are needed. For this, one can use techniques that allow explaining the contribution of each variable in the prediction of the best model (such as LIME [42]). Concretely, SHAP is proposed as a particular instance of this step when black-box models are used, since it unifies other existing methods and showed improved computational performance and better consistency with human intuition than previous approaches [31].

#### *G. Generation of rubrics, detection of discrepancies and biases*

The variables selected by the best model and the study of their influence can be used to generate rubrics and to get evidences of biases, inconsistencies and discrepancies. Nevertheless, it is recommended to supervise the results and manually weight the variables to generate the rubric, unless MAE and RMSE are very good. In this case, a rubric could be automatically generated by means of the previous techniques with reliable results. In general, the variables that have the most influence on the variability of the grade, as selected by this methodology, should be taken into account to fine-tune the evaluation criteria around them in the design of rubrics.

In the case that variables from the style set are chosen by the models as the ones influencing the grade, the teachers should observe them closely, considering further precision on the rubric around the aspects evaluated by these variables, since this is a signal of possible discrepancies or inconsistencies on these aspects. In case that context-based variables are chosen by the models, there is an alert on possible biases or discrepancies, especially if the chosen variables correspond to teachers or gender. Statistical tests must be carried out to confirm if there are significant differences. If so, bias has been detected. Additionally, correlation tests between unselected context-based variables and the selected variables would also give evidence of bias.

### IV. EXAMPLE OF APPLICATION OF THE METHODOLOGY

#### *A. Academic context and task overview*

*Computer Systems* is a course common to the degrees in Software Engineering and in Mathematics at the University of La Rioja. It is taught in the

first semester of the first year as a common course for both degrees and consists of six European Credit Transfer and Accumulation System (ECTS) credits. The teaching hours are divided into a weekly theory hour and two weekly computer lab sessions (90 minutes each). The autonomous work of the students is estimated at 90 hours throughout the semester. Students in both degrees are mixed in the theory and the computer lab sessions. Around 75 students take the course each year, of which approximately 50 belong to the Software Engineering degree and 25 to the degree in Mathematics. The first part of the course is a basic introduction to computer architecture, internet protocols, character encoding, file systems and basic administration of Operating Systems. The second part focuses on the HTML5 and CSS3 languages and consists of an introduction to web page design.

The evaluation of the course is divided into three parts: 60% of the final grade depends on a written test, 20% on the deliverables of the laboratory sessions and 20% on a project carried out in groups which consists of the creation of a website related to the contents of the course. To carry out the project, students are divided into groups of two or three people regardless of the degree they are studying. Each group is assigned a topic and the students must collaborate to create a website that has exactly the same content for all group members (HTML files must be the same), but each student must apply a different design to their website (different CSS files). Thus, each group presents a project for each student, who will receive an individual grade. Students have thirty days to finish their projects. The project delivered must satisfy a series of requirements. These requirements are designed for students to develop HTML and CSS code that is not only correct but also in accordance with the standards and following the design principles of HTML5: compatibility, usability, interoperability and universal access [43]. Some optional tasks are also proposed, such as versions of the website provided in different languages, a *responsive design* or meeting the WCAG accessibility guidelines at any of its levels [44]. Throughout the course, special emphasis is placed on the importance of validating HTML and CSS documents and adhering to standards. On the one hand, this facilitates future developments and updates, avoids potential problems and contributes to the acquisition of good habits, which is crucial in the

first Computer Science courses [45], [46], these being some of the transferable skills and professional competencies of the subject that are intended to be improved with the project. On the other hand, validation is an efficient method to check for errors in HTML and CSS, making it easier to learn these languages [47], [48]. Despite these advantages, many students do not validate their code in introductory web development courses [48], so it was forced as a requirement.

The evaluation of the proposed task has technical aspects that are easily quantifiable, such as checking the number of different HTML tags and CSS properties used, the number of languages or whether or not the web page is responsive. On the contrary, the creative part is not so easy to evaluate because it is more subjective, *e.g.* the general appearance, functionalities, or contents. Since the Computer Systems course belongs to the first year of two degrees and has many computer lab groups, there has been a variable number of teachers who have taught and graded this task over the years. Also, it is usual for each course to have new teachers who never taught the course before. This makes us suspect that, although we take measures to avoid it (such as meetings before and after the evaluation), there is no uniformity in the way of grading, especially when evaluating the creative part. Therefore, the application of the proposed methodology in this context can help detect possible discrepancies and inconsistencies between the assessments of the different teachers. It can also help to detect which variables have the most influence on the grade to propose a rubric.

### B. Applying the methodology

First, two teachers with a wide experience in the course analyzed the past projects and the requirements to determine which variables could influence the final grade of the projects. A total of 33 were considered. A brief description of all of them is presented at <https://bit.ly/3iOFTn7>. Concretely, they identified:

- 23 technical variables, such as the number of HTML files, CSS files, their size, the number of different HTML and CSS tags used, etc.;
- 6 style variables: the general appearance, functionality, content, positioning, contrasts and readability of the code;

- 4 context-based variables: degree, gender of the student, teacher who corrected and number of members of the group.

Once the variables had been determined, data were extracted from students' submissions in years 2015/16, 2016/17, 2017/18, 2018/19 and 2019/20. A total of 325 projects were treated. To extract the information for the 23 variables of the technical part, the data were obtained automatically by means of an *ad-hoc* software tool implemented to analyze the projects. The context variables were obtained automatically from the student and grade lists. The creative part (style variables) does not allow automation and these variables are harder to measure. When collecting such data, we tried to avoid introducing discrepancies from different teachers, therefore two experienced teachers of the course (the same who identified the variables) manually analyzed the projects following a blind review process. This way, data were collected to feed the models and then all the projects were analyzed under equal conditions.

After the preprocessing step (normalization and binarization of categorical variables), the database finally comprised 322 instances (3 samples that were incomplete were eliminated) and 38 input features plus the output variable to estimate, *Grade*, which ranges from 0 (lowest grade) to 2 (highest grade) with increments of 0.05. Several ML algorithms suitable for a regression problem of this type were selected. In this case, the techniques used were: Ridge regression (RIDGE), regression support vector machines (SVR), artificial neural networks (ANN), k-nearest neighbors (KNN), decision trees (DT) and extreme gradient boosting machines (XGB). All models were optimized by grid search. A cross-validation with 5 folds repeated 10 times was designed to ensure robustness in the estimates. Table I shows the hyper-parameters of the best models obtained as well as the mean absolute error (MAE) and the RMSE.

The next step requires to select a model to be optimized. In this case, ANN was chosen because it obtained the lowest MAE and RMSE. Despite ANN requires high computational resources to be trained, it is a feasible choice here due to the small size of the dataset. GAparsimony was chosen as feature selection and hyper-parameter optimization technique. The *ReRank* variable of the optimization process with GA was set at 0.01 to balance complexity and  $RMSE_{val}$ .



TABLE I  
RESULTS OBTAINED BY THE DIFFERENT MODELS (TOGETHER  
WITH THEIR OPTIMIZED HYPER-PARAMETERS) IN PYTHON

Model	MAE	RMSE
RIDGE (alpha=29)	0.106	0.138
KNN (n_neighbors=1)	0.095	0.148
ANN (neuron_num=8, alpha=0.06)	0.088	0.120
DTreeRegressor(mae,random,max_depth=25)	0.097	0.132
SVR (C=1.0, eps=0.003, gamma=0.025119)	0.088	0.123
XGB (mdep=5, nroun=250, subsam=0.95, colsam=0.30)	0.089	0.122

This optimization process was defined with a population of 40 individuals evaluated in 40 iterations but with a stop criterion if the error  $RMSE_{val}$  did not improve in 20 consecutive generations. The selection process used the 20% of the best individuals (elitists). Regarding the choice of these parameters, most of them are indeed the default ones in GAparsimony, since the tool is precisely intended to be used with small databases (< 10000 rows, < 100 variables), see [29]. The goal of the other modifications (population size, early stopping) was to achieve a lower convergence time.

### C. Results

Through the optimization process (of both hyper-parameters and variables) with GAparsimony, the best parsimonious ANN model was obtained with 14 variables. This model is an MLP (multilayer perceptron) that requires 21 neurons, the activation function is *tanh*, the solver for weight optimization is *lbfgs* and alpha is set to 0.514. Specifically, this best parsimonious ANN model reached a RSME of 0.116. Note that the best previous ANN model with all variables obtained a RMSE of 0.12 (cf. Table I). Thus, the best model obtained after applying the GAparsimony methodology is better and simpler.

Since ANN is a black-box model, the impact of the selected variables on the best parsimonious model is studied by means of SHAP.

- Figure 2 shows the average impact of the selected variables (the global importance of each feature). A longer bar means more impact on the final grade.
- Figure 3 shows a scatter plot that combines the importance and effect of each selected variable (each feature) on the grade. Each point is a SHAP

TABLE II  
VARIABLES SELECTED BY THE BEST PARSIMONIOUS ANN

ANN	
UI	Teacher2
AccessibilityNotSatisfied	Teacher3
PotentialAccErrors	GeneralAppearance
HTMLTags	IMGLLabel
ImageFiles	Functionality
GroupMembers	Content
Teacher1	Contrasts

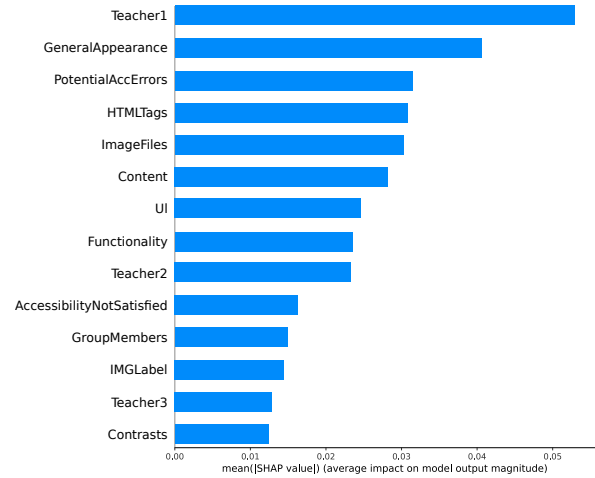


Fig. 2. Average impact on the grade of the selected variables using SHAP

value for a feature and an instance. Points on the left hand side of the x-axis have a negative impact on the grade, and similarly those placed on the right hand side have a positive impact. Colors represent the value of the feature from low (blue values) to high (red values). The variables are ordered according to their importance.

Table II provides the list of the variables selected by the best model, which is an evidence of their influence on the final grade. In view of the data, the following results are observed.

Three context-based variables are selected: Teacher1, Teacher2 and Teacher3. Those are binary variables. The SHAP analysis shows that Teacher1 causes a high negative impact on the grade, since red points (the ones where the value is 1, i.e., when that teacher graded the projects) are placed

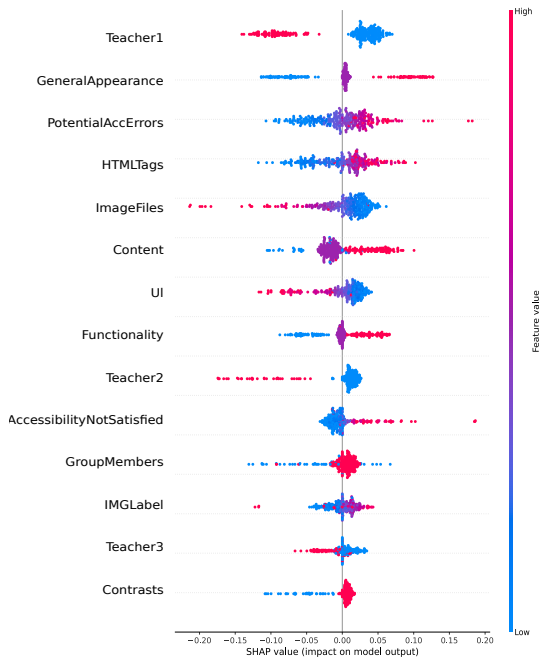


Fig. 3. SHAP value impact on model (ANN) output (the grade)

on the left in Figure 3. Similarly, there are some evidences that both `Teacher2` and `Teacher3` also influence negatively the grade, but in a lower manner. This, together with the fact that those variables were selected by the best model, gives some evidence of discrepancies in the grading among different teachers.

Several style variables were also selected: `Content`, `Functionality`, `GeneralAppearance` and `Contrasts`. `GeneralAppearance` is particularly significant, for it has the highest impact (a better appearance results in a better grade, as expected). Since those variables are subjective, we raise an alert on possible discrepancies or inconsistencies and a careful study should be done to generate an appropriate rubric.

Let us note that neither `Gender` nor `Degree` were selected, which leads us to suspect that there are no apparent biases in the grades of the projects based on these two features. It could happen that these variables were related or derived from others, but statistical tests were performed and they show a very small negligible correlation between those two variables and the rest (both Pearson and Spearman correlation coefficients are in between  $-0.15$  and  $+0.11$  for every variable

compared with `Gender` and `Degree`).

Moreover, the variable `GroupMembers` is selected by the best model. This means that the number of members of the group is taken into account in the evaluation; which, in principle, is the expected result: three-person projects should require more effort than projects developed in pairs for the same grade. However, SHAP analysis reveals here that 3-member groups have a minor positive influence on the grade (there are many red dots centered on the x-axis), but there are many cases where 2-member groups are penalized (blue dots placed at the left). This is not expected and gives evidence of an inconsistency.

The other variables selected by the best model can be grouped in several sets:

- The number of HTML tags has a positive impact. This makes sense, since students that employ most of the tags studied in the course are expected to obtain better grades. This variable is important, and it might also comprise more information that is scattered over different variables, like the number of tables or JavaScript code.
- The accessibility variables `AccessibilityNotSatisfied` and `PotentialAccErrors` are both chosen. Let us remark that accessibility is an optional requirement. Surprisingly, the SHAP values shows that projects with more errors and the ones not satisfying accessibility criteria have higher marks. This could seem to be an inconsistency, since the care of accessibility should increase the grade. However, there is a correlation (confirmed by means of statistical tests) between the variables `PotentialAccErrors` and `AccessibilityNotSatisfied` and other variables which increase the grade such as `HTMLTags`, `TotalHTMLBytes` or `IMGLabel`. Thus, projects with many HTML tags and large HTML files have more accessibility problems than projects with fewer HTML tags and code. Since accessibility is an optional requirement and the number of HTML tags increases the mark, these projects have a high mark.
- Variables related to images and lists: `ImageFiles`, `IMGLabel` and `UI`. A small number of images and UI tags (lists) have a positive impact on the grade. This is a somewhat

TABLE III  
CONFIGURATION OF SVR AND XGBOOST BEST PARSIMONIOUS  
MODELS

SVR	XGBoost
C = 2.7940	n_estimators = 763
Gamma = 0.0763	max_depth = 5
Epsilon = 0.0776	subsample = 0.4837
kernel = <i>rbf</i>	colsample_bytree = 0.5259
	gamma = 0.0068
	reg_alpha = 0.097
	reg_lambda = 1.2358

unexpected result, since in principle more images and lists should result in better projects with higher marks. After carefully studying the submitted projects, the more experienced teachers found that on many occasions students included many images and lists in the websites, but neglected other aspects, specially the positioning, which lead to lower grades. In what regards images and other elements, it is not just the amount of them what ensures the quality of the webpage, but their sensible contribution to a good result.

To check the robustness of the results, the methodology was also applied to the second and third models obtained in Step III-D to see if similar results are obtained. This would also allow obtaining a better picture of the whole process and would also be useful in case a high RMSE was obtained in step III-D. Concretely, the methodology was applied with SVR and XGB (also using GAParsimony, SHAP and the same repeated cross validation as in the case of ANN). SVR achieved an RMSE of 0.117327 with 18 variables, while XGB obtained 0.116321 with 13 variables. In each case, the best configuration is indicated in Table III. The SHAP analyses of SVR and XGB are presented in Figures 4 and 5, which show the robustness of the approach. 13 of the 14 variables chosen by ANN are also selected by SVR. Furthermore, the influence of the variables in the models is very similar, as it can be clearly seen, for instance, with Teacher1, Teacher2, GeneralAppearance, Contents and so on. ANN and XGB share 9 variables, again showing similar behaviors in the model output (see for instance the SHAP values of the most representative one, GeneralAppearance).

As a final comparison with a baseline approach,

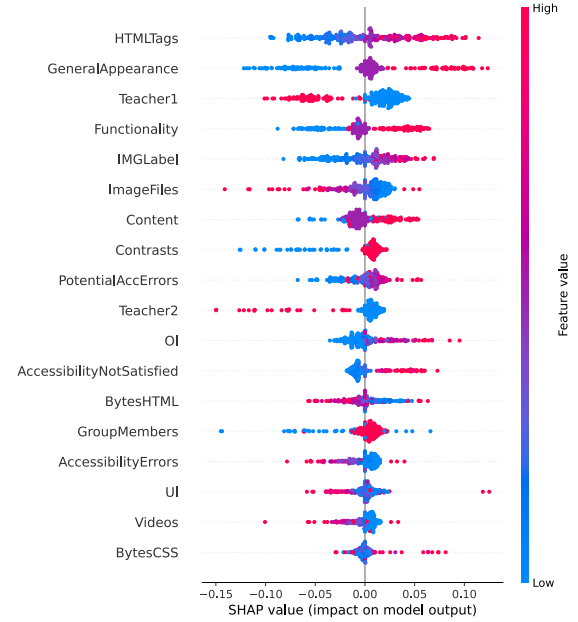


Fig. 4. SHAP value impact on model (SVR) output (the grade)

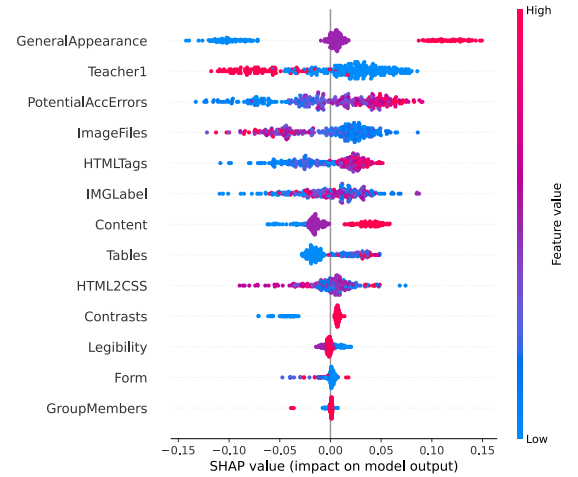


Fig. 5. SHAP value impact on model (XGB) output (the grade)

note that Step III-D of the methodology could also give some idea about the importance of features, since some of the ML techniques have an intrinsic feature selection step built in, such as decision trees and XGB. Thus, one can explore which are the variables most used by the models obtained in Step III-D (recall that

Scoring Rubric Framework						
Attributes	Criteria	Scale of Score				
		(Optional)	1. Not Acceptable	2. Below Expectations	3. Meets Expectations	4. Exceeds Expectations
Attribute 1		Cell 1,1	Cell 1,2	Cell 1,3	Cell 1,4	Cell 1,5
Attribute 2		Cell 2,1	Cell 2,2	Cell 2,3	Cell 2,4	Cell 2,5
...						
Attribute n		Cell n,1	Cell n,2	Cell n,3	Cell n,4	Cell n,5
Total Score						(%)

Fig. 6. Scoring Rubric Framework taken from [49]

not all models permit this). For this task, the XGB and decision tree models presented in Table I were chosen. The analysis of feature importance reveals important differences between both models: only one variable matches between the five variables with the highest importance; only 3 of them are shared between the top-10 variables. Even worse, the most important variable for the decision tree is `GeneralAppearance`, but it is the third least important variable according to XGB. Therefore, the results are not similar to each other, although they may help to give an idea about the variables. A few of the important variables do coincide with those selected according to the methodology (presented in Table II), but this is not the case for most of them. In fact, according to both the baseline approach with XGB and decision tree, `GroupMembers` barely has influence, but the methodology with the SHAP analysis clearly showed inconsistencies. Something similar happens with `GeneralAppearance`: following the methodology it is selected as one of the most important variables (by any of the three models), but the baseline approach with XGB states completely the opposite.

Finally, the obtained insights on the role of the variables can be applied to the improvement of the evaluation process, by means of a finer tuning of the tools used for evaluation, such as rubrics. For this task, the scoring rubrics-assisted reading (SRAR) technique and its Scoring Rubric Framework [49] can be used (see Figure 6).

As an example, taking into account the results produced by the optimization process with `GAparsimony` and the SHAP analysis, the following variables are considered to be included in the rubric using the SRAR technique: `GeneralAppearance`, `HTMLTags`, `Content`, `Functionality`, and `Contrasts`. These variables are selected by the best parsimonious ANN model for which the SHAP analysis in Figure 3 reveals a coherent behavior (that is, the variables for which high values, represented in red,

Attributes	Criteria	Scale of Score					Score
		(Optional)	1. Not acceptable	2. Below expectations	3. Meets expectations	4. Exceeds expectations	
General disposition	Correct		There are elements that cannot be seen	Some elements cannot be seen when the window size changes	Every element is correctly displayed	The overall design is correct and accessible	5%
Consistency	Consistent and coherent		Different presentations for analogous elements make the webpage difficult to understand	There are unnecessary differences in analogous elements	Analogous elements have coherent aspect	There is a clear Style Guide for the webpage	5%
Typography	Clear and coherent		The size, font or style of the text makes part of it unreadable	The size, font or style of the text makes part of it hard to read or is incoherent	The typography is coherent, simple and clear	The typography scheme is well chosen, accessible and pleasant	5%
Simplicity	Simple and efficient		The web page is too busy and it is difficult to find the elements and the information	The web page is somewhat busy and some elements or information cannot be found easily	The different elements are included appropriately	Design is appealing, accessible and easy to understand	5%
Layout	Attractive and usable		The website has an unattractive layout	The website has a very simple layout, without changing the position of elements	The website has an attractive and usable layout	The website has a very attractive and usable layout	5%
Total score							25%

Fig. 7. Example Rubric for the variable `GeneralAppearance`

are placed on the right-hand side and therefore have a positive impact). The score corresponding to each one of these variables is given according to the average impact depicted in Figure 2. The other variables that were also selected by the best parsimonious model are not included in the rubric because, as explained previously, their behavior is not coherent.

Consider the variable `GeneralAppearance`. It is highly influential and its influence has a linear behavior; therefore, it is a variable which should be included in the rubric. Following the Scoring Rubric Framework in Figure 6, the recommendations for developing scoring rubrics [50], [33] and after a meeting among teachers, a particular rubric for this variable is produced, which can be seen in Figure 7. This rubric meets the requirements of giving formative and fine-grained feedback to the students, and at the same time is a good way to reduce discrepancies in the assessments. In addition to the selected variables, the proposed rubric includes an additional feature called `OtherElements` where some other relevant aspects for the development of the website are considered. This feature has a low score in the rubric since it corresponds to features that have not been selected in the `GAparsimony` and SHAP analysis. The scores of the scoring rubrics are in percentages and the total score is 110%, since teachers consider that the maximum total score can be obtained without the need of reaching the maximum score in each attribute. The rubric we developed following this approach can be found at <https://bit.ly/3ix9JMR>.

## V. DISCUSSION

This paper is framed into the general principle of (automatically) finding hidden knowledge in data to improve a certain process. In our case, the process is teaching –in particular the evaluation of complex tasks in Engineering and Computer Science–, and the data are the grades from previous years and the work delivered by the students. Finally, the knowledge is about what factors are taken into account for grading, the consistency of the grading and possible discrepancies or biases. The following subsections discuss different aspects of the proposed methodology.

### A. Evaluation of the assessment process in PBL

One of our main goals is to evaluate the assessment process in PBL, as a contribution to help resolve the lack of studies on the evaluation of the quality of the measurement instruments [6]. Our methodology shows that ML algorithms can be used to perform this evaluation, helping the discovery of biases, inconsistencies or discrepancies, and also providing ways for the improvement and tuning of the already used evaluation methods, such as rubrics.

The incorporation of teachers in the process, in particular in the selection of the variables, can provide deeper understanding of the evaluation process and help the identification of possible flaws. There is a need of attention to interpretable features of ML models, what is called the *interpretable feature space* in [32]. This should be implemented as a context-aware analysis of what it means for a feature to be *interpretable* for a given purpose [51], [32]. In many ML projects, the goal for developers and theorists is to identify features that maximize model performance, often without consideration for whether the designed features are meaningful to domain experts. To solve this problem, [52] introduced a classification of user types of ML models. This classification was extended in [32], based on their goals related to the ML models. These user types are defined as Developers, Theorists, Ethicists, Decision Makers and Impacted Users. The main target group of our proposal is Decision Makers, which in this context are teachers and graders, who are not expected to have previous knowledge on ML but are domain experts in the subject they are teaching using PBL. Our methodology is a way to consider the ways in which features need to be interpretable in the

educational context, in particular in what respects to decision making by teachers without ML experience. This is one of the main goals of our proposal.

Together with the incorporation of the teachers in the feature selection and explanation of the model's findings, our proposal includes an automatization component. Automatic and semi-automatic grading has a long tradition in Computer Science teaching, in particular in introductory programming courses [53], [54]. There is a variety of systems that provide clear advantages such as improving the immediacy of feedback given to students and optimizing the teachers use of time, but show also some limitations like the difficulty in usage by students due to UI/UX difficulties, the amount of syntax errors that beginners make, or language barriers. In our case, a full automatization is solely used to check the correctness of the HTML and CSS code generated by the students, and this is done via the W3C validators, which are standard tools. We therefore benefit from the advantages (quick feedback and optimization of instructors' time) and at the same time avoid the usage difficulties. Other features of the work that are automatically graded are counting tasks, such as the requested number of files. These are included in what we called technical variables.

### B. Feedback and explainable ML

One of the main research issues in automatic grading is how to provide meaningful feedback to the students, so that their learning process is improved. For objective task assessment, such as evaluating the correctness of code in programming courses, this includes the analysis of the feedback itself, see [55]. In more complex tasks, such as automated essay evaluation, in which ML techniques play a major role, it is important to explore the explainability of the whole process, to avoid a black-box effect [56]. The issue of interpretable and explainable ML models has recently become a hot topic in the Artificial Intelligence community [57], in which SHAP techniques [31] are playing a relevant role. We used SHAP to analyze our models so that they have a pedagogical value and can help, as stated in [58], to (a) discover the decision-making process that drives our model (b) fine-tune the prediction process to improve generalizability and interpretability, and (c) help providing personalized, formative, and fine-grained feedback to students. The

proposed instance of the methodology of using evolutionary optimization to obtain parsimonious ML models and explain them using SHAP could be helpful to teachers in achieving some of these goals. In the first place, the decision-making process is more transparent than in other ML approaches since we can observe and study the evolution of the models as these tend to use a smaller set of variables to explain the grades; moreover, the SHAP study allows us to characterize the influence of each variable, which allows us to focus on particular aspects of the teaching-learning process that could have been overseen otherwise. This is, in part, what means to *discover hidden knowledge in data*. Secondly, the tuning of the prediction process is itself one of the main characteristics of evolutionary optimization. Finally, the identification of relevant variables and qualitative/quantitative findings on their influence on the grading can help the instructors to develop more accurate and informative rubrics that provide fine-grained feedback to the students.

### C. GAparsimony

The use of GAparsimony in this methodology provides some advantages with respect to basic optimization steps (like grid search and similar techniques). For instance, feature selection (via a parsimony criterion), permits building simpler models. Simpler models generalize better, and permit generating simpler rubrics since fewer variables are involved. The optimization process could also obtain models with better accuracy.

Note that the percentage of appearance of each variable within the elitists of the last generation can also be computed. Such percentages provide information about how many times a variable has been selected in the best models. Thus, they could provide evidences on which variables influence on the final grade. However, the fact of obtaining information from the last generation instead of the best model does not provide better results: the models are rather similar and this step would reintroduce collinearity and relations between variables, which make the explanations harder.

### D. Robustness

The first step of the methodology (identification of variables) is particularly important. For this purpose, an adequate initial variable selection manually performed with expert knowledge of the problem

combined with ML competence is ideal. The omission of one or more important variables will significantly affect the results, but the *quality* of the predictions can be estimated thanks to the initial RMSE obtained in Step III-D (if the results are not good enough at that point, could be that some variable is missing, so that one can go back to Step III-A to start identification again). Apart from the identification in Step III-A, different optimization techniques and the choice of the initial model could generate similar results (the more robust, the more similar), but not totally the same. In any case, a study with several models may help further. Compared to a baseline approach, where the weights of a simple model are studied, this methodology is more powerful since it allows the use of more complex models (and likely with higher predictive power) without losing the model's explainability.

With respect to Step III-F, the case study shows that SHAP results remain very similar even when changing models. However, if different models show very different selected features and SHAP values, then it is likely either one has started with a model with low performance (therefore, its predictions are not reliable) or there are high correlations between variables, since a high correlation in features often produces multiple equally optimal selections.

### E. Rubric development

The development and improvement of rubrics in software development projects like ours is closely related to the reading techniques on software quality checking [59], [60]. Such reading techniques enhance the rubrics based on check-lists [61] and have been successfully applied in a similar context [49], concretely using the scoring rubrics-assisted reading (SRAR) technique. This is the approach we follow to develop the rubric in our case study, together with the recommendations on rubric development [50], [33].

Teachers should develop the complete rubric analyzing the knowledge obtained from the methodology. Note that if the methodology produces a model with a very high predictive power and no bias is found, teachers could use the identified variables and their SHAP values to generate automatically, at least partially, a rubric. However, it is more advisable that teachers develop it based on the findings of the methodology and following the recommendations [50], [33]. The proposed methodology has as an advantage that

it helps in the accomplishment of several of such recommendations, although, as expected, it cannot totally guarantee them. For instance, it helps with the recommendations “the criteria set forth within a scoring rubric should be clearly aligned with the requirements of the task” and “the criteria should be fair and free from bias”. A drawback is that the rubric is developed afterwards and requires the evaluation of previous projects to form the dataset. However, this can be interesting when there is no rubric available or the existing rubric is not sufficiently refined.

There are many research works on automatic scoring and feedback, but far fewer on automatic rubric generation. For the concrete case of essays, deep neural networks and natural language processing tools have recently been used to predict finer-grained rubric scores and to uncover important features of an effective rubric scoring model [56], [58]. In an educational programming environment, code-chunks and classical interpretable models (linear regression and LASSO) have been used to generate automatically generated rubric parameters [62]. That work can be viewed as an instance of our methodology (note that our methodology also allows one the use of more advanced techniques and models).

#### F. Detection of discrepancies, bias and inconsistencies

The other way to apply the obtained results is to detect discrepancies in the assessments, which was one of our goals. The best model optimized with GAParsimony and applied to our example indicates that indeed some teacher's variables are highly influential, despite the fact that we hold assessment meetings to unify criteria. Clearly, this is not enough, and the actual data tell us that this unification is not so clear. The direction of the influence of each of the teachers can be seen in the SHAP analysis. For instance, *Teacher1* tends to give lower marks, as seen in Figure 3. A way to avoid these discrepancies to some extent is to use the scoring rubrics that tune the scoring in the more influential style variables. With respect to biases, these can be detected by the context-based variables, in the sense that if any of them is identified as a highly influential variable, then a bias can be present and we should inspect this. The absence of context-based variables indicates that biases could not be detected by the model, which is some evidence in favor of the absence of biases, although not completely conclusive.

#### Limitations

It is important to remark that totally automatic grading is not pursued by this methodology. In fact, if teachers do not strictly follow an existing rubric, low RMSE and MAE errors are not common due to the subjectivity of the evaluation task, i.e. a high intrinsic error is expected when building the models. However, the effort of pursuing a model that minimizes the error provides information and conclusions about the evaluation task, since the process and the optimized models detect the most influential variables. If RMSE and MAE are extremely high, then there could be high inconsistencies between the project assessments, including data of the same teacher. Moreover, a review of Steps III-A and III-B is recommended to detect a lack of identified variables or data, since one should not extract conclusions from models with very high RMSE and MAE. This represents a way to help teachers collect and lower possible discrepancy or bias issues for example by co-grading or improving the evaluation instruments at the points that deal with the selected variables.

Feature selection and optimization techniques, in particular GAParsimony, usually demand high computational resources since hundreds of models need to be trained. Thus, if the dataset is very big, the model with the lowest RMSE could not be suitable for GAParsimony and one needs to select a simpler model. Alternatively, one could try to undersample the dataset (taking only a subset of the students), or reducing the number of identified variables in Step III-A. Also, multi-fidelity optimization techniques (like Hyperband [40]) could be useful in this situation, since they are common approaches to solve the problem of limited resources and time.

Another known limitation is that the methodology cannot guarantee the absence of bias if no context-based variables are selected by this methodology. However, the selection of such context-based variables does provide evidence of bias existence. In addition, further analyses could be performed, for example, Step III-D could be applied to predict the context-based variables. Then, bias has been detected if one gets a low RMSE value.

#### VI. CONCLUSIONS AND FUTURE WORK

This paper has presented a high level approach to the evaluation of the assessment process based on the



PBL methodology and also to the automatization of such an assessment process. The goal is to minimize bias, discrepancies and inconsistencies. We do it by automatizing some parts of the process, by evaluating the contribution of selected variables that might induce bias or discrepancies, and by explaining and improving the evaluation through actionable insight gained from the ML models. Specifically, this paper presents a general methodology for assessing project evaluation, having one particular *instantiation* with GAParsimony and SHAP techniques, for feature selection and explanation of the model output, respectively.

The methodology provides a way to work on rubric development based on explainable ML models and previous teaching experience, with the explicit intention of reducing biases and discrepancies. This is important since one of the fundamental challenges of applying project-based learning methodologies is the design of adequate evaluation systems [1], [12], and a correctly established rubric provides a standardization of the assessment by specifying a guide for grading according to clearly established criteria in advance.

We can therefore state that the application of methodology (and its proposed instantiation based on genetic algorithms with SHAP explanation for rubric development and improvement) is a contribution to make a practical, explainable, and trustworthy use of Machine Learning in the evaluation of complex projects in Engineering and Computer Science studies.

As future work, we plan to provide an automatic software tool to facilitate the use of the proposed methodology in similar situations. The evaluation of the rubrics produced or improved by this methodology has been left out of the scope of this paper. It is however a natural extension of the present work. In this respect, the Cronbach alpha index [63] is the main measure to consider, which has already been used to evaluate the consistency of rubrics in computer science and programming courses [64], [65]. For this, we propose the following cycle: evaluate student's work, evaluate the used rubric if possible, assess the evaluation process using the herein described methodology finding the most important variables in order to introduce a rubric or improve the previous one, evaluate the students using the new rubric and start over. The new rubric should help avoiding biases, discrepancies and inconsistencies, and should in principle obtain better results in rubric evaluation metrics, such as a higher

Cronbach alpha index.

## ACKNOWLEDGMENT

This work is supported by grant PID2020-116641GB-I00 funded by MCIN/ AEI/ 10.13039/501100011033.

## REFERENCES

- [1] J. Chen, A. Kolmos, and X. Du, "Forms of implementation and challenges of PBL in engineering education: a review of literature," *European Journal of Engineering Education*, vol. 46, no. 1, pp. 90–115, 2021.
- [2] D. Kokotsaki, V. Menzies, and A. Wiggins, "Project-based learning: a review of the literature," *Improving Schools*, vol. 19, no. 3, pp. 267–27, 2016.
- [3] J. S. Krajcik and N. Shin, *Project-Based Learning*, 2nd ed., ser. Cambridge Handbooks in Psychology. Cambridge University Press, 2014, p. 275–297.
- [4] S. M. Al-Balushi and S. S. Al-Aamri, "The effect of environmental science projects on students' environmental knowledge and science attitudes," *International Research in Geographical & Environmental Education*, vol. 23, no. 3, pp. 213–227, 2014.
- [5] L. Helle, P. Tynjälä, and E. Olkinuora, "Project-based learning in post-secondary education. Theory, practice and rubber sling shots," *Higher Education*, vol. 51, pp. 287–314, 2006.
- [6] P. Guo, N. Saab, L. S. Post, and W. Admiraal, "A review of project-based learning in higher education: Student outcomes and measures," *International Journal of Educational Research*, vol. 102, p. 101586, 2020.
- [7] M. Prince and R. Felder, "Inductive teaching and learning methods: Definitions, comparisons, and research bases," *Journal of Engineering Education*, vol. 95, pp. 123–138, 2006.
- [8] S. Bell, "Project-based learning for the 21st century: skills for the future," *The Clearing House: A Journal of Educational Strategies, Issues and Ideas*, vol. 83, no. 2, pp. 39–43, 2010.
- [9] M. Barak, "From "doing" to "doing with learning": reflection on an effort to promote self-regulated learning in technological projects in high school," *European Journal of Engineering Education*, vol. 37, no. 1, pp. 105–116, 2012.
- [10] J. Kay, M. Barg, A. Fekete, T. Greening, O. Hollands, and J. H. Kingston, "Problem-Based Learning for Foundation Computer Science Courses," *Computer Science Education*, vol. 10, no. 2, pp. 109–128, 2000.
- [11] J. Chaparro-Peláez, S. Iglesias-Pradas, F. J. Pascual-Miguel, and A. Hernández-García, "Factors affecting perceived learning of engineering students in problem-based learning supported by business simulation," *Interactive Learning Environments*, vol. 21, no. 3, pp. 244–262, 2013.
- [12] T. Kumberger, "Revising a design course from a lecture approach to a project-based learning approach," *European Journal of Engineering Education*, vol. 38, no. 3, pp. 254–267, 2013.
- [13] J. L. Martín Núñez, E. Tovar Caro, and J. R. Hilerá González, "From Higher Education to Open Education: Challenges in the Transformation of an Online Traditional Course," *IEEE Transactions on Education*, vol. 60, no. 2, pp. 134–142, 2017.
- [14] Y. Doppelt, "Assessment of Project-Based Learning in a Mechatronics context," *Journal of Technology Education*, vol. 16, pp. 7–24, 2005.



- [15] T. Gomez-del Rio and J. Rodriguez, "Design and assessment of a project-based learning in a laboratory for integrating knowledge and improving engineering design skills," *Education for Chemical Engineers*, vol. 40, pp. 17–28, 2022.
- [16] A. Rasooli, H. Zandi, and C. DeLuca, "Re-conceptualizing classroom assessment fairness: A systematic meta-ethnography of assessment literature and beyond," *Studies in Educational Evaluation*, vol. 56, pp. 164–181, 2018.
- [17] P. Steinke and P. Fitch, "Minimizing bias when assessing student work," *Research & Practice in Assessment*, vol. 12, pp. 87–95, 2017.
- [18] D. W. Allen, "Toward a common currency: Using item response theory to adjust for grading inconsistency," 2007, Mathematics, Statistics, and Computer Science Honors Projects, MacAlester College. [Online]. Available: [https://digitalcommons.macalester.edu/mathcs\\_honors/](https://digitalcommons.macalester.edu/mathcs_honors/)
- [19] J. M. Montero, R. San-Segundo, J. Macías-Guarasa, R. de Córdoba, and J. Ferreiros, "Methodology for the analysis of instructors' grading discrepancies in a laboratory course," *International Journal of Engineering Education*, vol. 22, no. 5, pp. 1053–1062, 2006.
- [20] S. Pero and T. Horváth, "Detection of inconsistencies in student evaluations," in *Proceedings of the 5th International Conference on Computer Supported Education (CSEDU-2013)*. New York, NY: ACM, 2013, pp. 207–216.
- [21] S. K. Green, R. L. Johnson, D.-H. Kim, and N. S. Pope, "Ethics in classroom assessment practices: Issues and attitudes," *Teaching and Teacher Education*, vol. 23, no. 7, pp. 999–1011, 2007.
- [22] T. M. Haladyna, S. B. Nolen, and N. S. Haas, "Raising standardized achievement test scores and the origins of test score pollution," *Educational Researcher*, vol. 20, no. 5, pp. 2–7, 1991.
- [23] G. Gweon, S. Jun, S. Finger, and R. C. P., "Towards effective group work assessment: even what you don't see can bias you," *International Journal of Technology and Design Education*, vol. 27, pp. 165–180, 2015.
- [24] L. Wittie, J. Bennett, C. Merrill, J. Graham, and T. Schwab, "Bias in first-year engineering student peer evaluations," in *ASEE Virtual Annual Conference*. Curran Associates Inc., 2021.
- [25] N. Gunantara, "A review of multi-objective optimization: Methods and its applications," *Cogent Engineering*, vol. 5, no. 1, p. 1502242, 2018.
- [26] R. Urraca, E. Sodupe-Ortega, J. Antonanzas, F. Antonanzas-Torres, and F. M. de Pisón, "Evaluation of a novel GA-based methodology for model structure selection: The GAPARSIMONY," *Neurocomputing*, vol. 271, pp. 9–17, 2018.
- [27] A. Sanz-Garcia, J. Fernandez-Ceniceros, F. Antonanzas-Torres, A. Pernia-Espinoza, and F. J. Martinez-de Pison, "GAPARSIMONY: A GA-SVR approach with feature selection and parameter optimization to obtain parsimonious models for predicting temperature settings in a continuous annealing furnace," *Applied Soft Computing*, vol. 35, pp. 13–28, 2015.
- [28] F. J. Martínez-De-Pisón, "GAparsimony: GA-based optimization R package for searching accurate parsimonious models. R package version 0.9-4," <https://github.com/jpison/GAparsimony>, 2020.
- [29] F. J. M. de Pisón, J. Ferreira, E. Fraile, and A. Pernia-Espinoza, "A comparative study of six model complexity metrics to search for parsimonious models with GAparsimony R package," *Neurocomputing*, vol. 452, pp. 317–332, 2021.
- [30] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.
- [31] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 4768–4777.
- [32] A. ZYTEK, I. Arnaldo, D. Liu, L. Berti-Equille, and K. Veeramachaneni, "The need for interpretable features: Motivation and taxonomy," *SIGKDD Explor. Newsl.*, vol. 24, no. 1, p. 1–13, jun 2022.
- [33] C. A. Mertler, "Designing scoring rubrics for your classroom," *Practical assessment, research, and evaluation*, vol. 7, no. 1, p. 25, 2000.
- [34] G. Team, "Google style guides," <https://google.github.io/styleguide/>, 2022.
- [35] P. Blumberg, "Assessing Students During the Problem-Based Learning (PBL) process," *Medical Science Educator*, vol. 15, pp. 92–99, 2005.
- [36] M. Wilson and K. Sloane, "From principles to practice: An embedded assessment system," *Applied Measurement in Education*, vol. 13, pp. 181–208, 2000.
- [37] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015, vol. 72.
- [38] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020.
- [39] Z. Karnin, T. Koren, and O. Somekh, "Almost optimal exploration in multi-armed bandits," in *International Conference on Machine Learning*. PMLR, 2013, pp. 1238–1246.
- [40] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.
- [41] Y. Zhang, S. Wang, and G. Ji, "A comprehensive survey on particle swarm optimization algorithm and its applications," *Mathematical Problems in Engineering*, pp. 1–38, 2015.
- [42] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You?": Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, p. 1135–1144.
- [43] M. Stachowiak and A. van Kesteren, "HTML design principles," W3C, W3C Working Draft, Nov. 2007. [Online]. Available: <https://www.w3.org/TR/2007/WD-html-design-principles-20071126/>
- [44] A. Campbell, A. Kirkpatrick, M. Cooper, and J. O. Connor, "Web content accessibility guidelines (WCAG) 2.1," W3C, W3C Recommendation, Jun. 2018, <https://www.w3.org/TR/2018/REC-WCAG21-20180605/>.
- [45] A. Cuoco, E. P. Goldenberg, and J. Mark, "Habits of mind: An organizing principle for mathematics curricula," *The Journal of Mathematical Behavior*, vol. 15, no. 4, pp. 375–402, 1996.
- [46] S.-N. A. Joni and E. Soloway, "But my program runs! discourse rules for novice programmers," *Journal of Educational Computing Research*, vol. 2, no. 1, pp. 95–125, 1986.
- [47] T. H. Park and S. Wiedenbeck, "Learning web development: Challenges at an earlier stage of computing education," in *Proceedings of the 7th International Workshop on Computing Education Research*. ACM, 2011, pp. 207–216.
- [48] T. H. Park, B. Dorn, and A. Forte, "An analysis of HTML and CSS syntax errors in a web development course," *ACM*

- Transactions on Computing Education*, vol. 15, no. 1, pp. 1–21, 2015.
- [49] E. O. Mkpojiogu and A. Hussain, “Can scoring rubrics be used in assessing the performance of students in software requirements engineering education?” *Journal of Telecommunication, Electronic and Computer Engineering*, vol. 9, no. 2-11, pp. 115–119, 2017.
- [50] B. M. Moskal, “Recommendations for developing classroom performance assessments and scoring rubrics,” *Practical Assessment, Research, and Evaluation*, vol. 8, no. 1, p. 14, 2002.
- [51] I. Arnaldo and K. Veeramachaneni, “The holy grail of “systems for machine learning”: Teaming humans and machine learning for detecting cyber threats,” *ACM SIGKDD Explorations Newsletter*, vol. 21, no. 2, p. 39–47, nov 2019.
- [52] A. Preece, D. Harborne, D. Braines, R. Tomsett, and S. Chakraborty, “Stakeholders in explainable AI,” in *AAAI FSS-18: Artificial Intelligence in Government and Public Sector*. Arlington, Virginia: AAAI, 2018, p. 6.
- [53] K. M. Ala-Mutka, “A survey of automated assessment approaches for programming assignments,” *Computer Science Education*, vol. 15, no. 2, pp. 83–102, 2005.
- [54] H. Aldriye, A. Alkhalaf, and M. Alkhalaf, “Automated grading systems for programming assignments: A literature review,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 3, 2019.
- [55] Q. Hao, J. P. Wilson, C. Ottaway, N. Iriumi, K. Arakawa, and D. H. S. IV, “Investigating the essential of meaningful automated formative feedback for programming assignments,” in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2019, pp. 151–155.
- [56] V. S. Kumar and D. Boulanger, “Automated essay scoring and the deep learning black box: How are rubric scores determined?” *International Journal of Artificial Intelligence in Education*, vol. 31, no. 3, pp. 538–584, 2021.
- [57] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning,” 2017.
- [58] V. S. Kumar and D. Boulanger, “Explainable automated essay scoring: Deep learning really has pedagogical value,” *Frontiers in Education*, vol. 5, 2020.
- [59] V. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, L. Sørumgård, and M. Zelkowitz, “The empirical investigation of perspective-based reading,” *Empirical Software Engineering*, vol. 1, pp. 133–164, 1996.
- [60] P. Runeson and T. Berling, “Evaluation of a perspective based review method applied in an industrial setting,” in *IEE Proceedings: Software*, vol. 150 (3). IEE, 2003, pp. 177–184.
- [61] E. O. Mkpojiogu, N. L. Hashim, and A. Hussain, “Enhancing the quality of software requirements artifacts with scoring rubrics-assisted reading technique,” *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 8, pp. 467–474, 2019.
- [62] N. Diana, M. Eagle, J. Stamper, S. Grover, M. Bienkowski, and S. Basu, “Data-driven generation of rubric criteria from an educational programming environment,” in *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*. ACM, Mar. 2018.
- [63] L. J. Cronbach, “Coefficient alpha and the internal structure of tests,” *Psychometrika*, vol. 16, pp. 297–334, 1951.
- [64] N. d. C. Alves, C. G. von Wangenheim, J. C. R. Hauck, and A. F. Borgatto, *A Large-Scale Evaluation of a Rubric for the Automatic Assessment of Algorithms and Programming Concepts*. New York, NY, USA: ACM, 2020, p. 556–562.
- [65] D. Saito, R. Yajima, H. Washizaki, and Y. Fukazawa, “Validation of rubric evaluation for programming education,” *Education Sciences*, vol. 11, p. 656, 10 2021.



**Jose Divasón** Jose Divasón received the degrees in Computer Science and Mathematics from the University of La Rioja (Spain) in 2011 and finished his Ph.D. in computer science in 2016. He has been a lecturer at University of La Rioja since 2017. His research interests include interactive theorem proving and computer algebra.



**Francisco Javier Martínez-de-Pisón** Javier is the Head of the EDMANS group. Professor at University of La Rioja. His research activities focus on soft computing, data mining and machine learning methods to solve problems in several fields as industry, energy, agriculture and business.



**Ana Romero** Ana Romero received the B.S. and Ph.D. degrees from University of La Rioja (Spain) in 2003 and 2007 respectively. She has worked at University of La Rioja as a lecturer (2007 – 2018), and currently is an Associate Professor at the same University. Her main research interest is computational algebraic topology and applications.



**Eduardo Sáenz-de-Cabezón** Eduardo Sáenz-de-Cabezón received the B.S. and Ph.D. degrees from University of La Rioja (Spain) in 2001 and 2008 respectively. He has been an Associate Professor at University of La Rioja since 2010. His main research interest is on combinatorial commutative algebra.