sustainability

MDPI

Article

The Effect of Green Software: A Study of Impact Factors on the Correctness of Software

David Gil [1,*], Jose Luis Fernández-Alemán [2], Juan Trujillo [3], Ginés García-Mateos [2], Sergio Luján-Mora [3] and Ambrosio Toval [2]

1 Lucentia Research Group, Department of Computing Technology and Data Processing, University of Alicante, 03690 San Vicente del Raspeig, Alicante, Spain

2 GIIS, DIS, University of Murcia, 30003 Murcia, Murcia, Spain; aleman@um.es (J.L.F.-A.); ginesgm@um.es (G.G.-M.); atoval@um.es (A.T.)

3 Lucentia Research Group, Department of Software and Computing Systems, University of Alicante, 03690 San Vicente del Raspeig, Alicante, Spain; jtrujillo@dlsi.ua.es (J.T.); sergio.lujan@ua.es (S.L.-M.)

* Correspondence: david.gil@ua.es; Tel.: +34-965903400

Received: 23 August 2018; Accepted: 25 September 2018; Published: 28 September 2018

Abstract: Unfortunately, sustainability is an issue very poorly used when developing software and hardware systems. Lately, and in order to contribute to the earth sustainability, a new concept emerged named Green software which is computer software that can be developed and used efficiently and effectively with minimal or no impact to the environment. Currently, new teaching methods based on students' learning process are being developed in the European Higher Education Area. Most of them are oriented to promote students' interest in the course's contents and offer personalized feedback. Online judging is a promising method for encouraging students' participation in the e-learning process, although it still has to be researched and developed to be widely used and in a more efficient way. The great amount of data available in an online judging tool provides the possibility of exploring some of the most indicative attributes (e.g., running time, memory) for learning programming concepts, techniques and languages. So far, the most applied methods for automatically gathering information from the judging systems are based on statistical methods and, although providing reasonable correlations, these methods have not been proven to provide enough information for predicting grades when dealing with a huge amount of data. Therefore, the great novelty of this paper is to develop a data mining approach to predict program correctness as well as the grades of the students' practices. For this purpose, powerful data mining technologies taken from the artificial intelligence domain have been used. In particular, in this study, we have used logistic regression, decision trees, artificial neural network and support vector machines; which have been properly identified as the most suitable ones for predicting activities in the e-learning domains. The results have achieved an accuracy of around 74%, both in the prediction of the program correctness as well as in the practice grades' prediction. Another relevant issue provided in this paper is a comparison among these four techniques to obtain the best accuracy in predicting grades based on the availability of data as well as their taxonomy. The Decision Trees classifier has obtained the best confusion matrix, and time and memory efficiency were identified as the most important predictor variables. In view of these results, we can conclude that the development of green software leads programmers to implement correct software.

Keywords: data mining; e-learning; experience report; online judging; logistic regression; decision trees; artificial neural network; support vector machines; MOOC (massive open online courses)

1. Introduction

Information and Communications Technology (ICT) plays a part in many aspects of human activity. The role of ICT in the UN Sustainable Development Goals [1] has been highlighted by many authors [2–4]. However, ICT plays a dual role [5]: ICT can help to reduce the environmental impact of activities in the environment, but, at the same time, it is also a significant user of resources, so it is also a main contributor to the negative impact of human activities. The ICT industry represents approximately 2% of global carbon dioxide (CO_2) emissions [6]; this amount of emissions is equivalent to aviation [7]. Furthermore, it is estimated that ICT will account for 3% of global emissions by 2020 [8], but recent research, based on the extrapolation from the current trends, argues that "for the near future (up to 2020) the current energy consumption plateau will remain while the number of users continuous to grow but at a decreasing rate" [9]. Clearly, there is exponentially growing energy consumption caused by the ICTs [10], which has a negative impact on sustainability. The ICT Green House Gas Emissions (GHGE) relative contribution could grow until 14% of the 2016-level worldwide GHGE by 2040. Regarding energy efficiency, the software system performance can be improved from different angles such as coding styles and algorithms [11]. There is not a standardized definition of green (or sustainable) software [12]. Nevertheless, the protection of resources is a commonly addressed issue. We will focus on environmentally sustainable software that is used efficiently and effectively.

After a decade of the implantation of the European Higher Education Area (EHEA) [13,14] in Spain, we are starting to observe profound changes in the processes of teaching/learning, thanks to the development of new methodologies that it boosted. It is well known that the purpose of the EHEA was to develop new teaching methods based on the students' learning process rather than the teacher's point of view. Under this perspective, the new methods should stimulate students' interest and offer appealing material, fair assessment and relevant feedback.

Based on these ideas and principles, we developed an innovative experience with a course on "Algorithms and Data Structures" (ADS) in the second year of a Computer Science Degree, using a web-based automatic judging system called Mooshak [15]. The course was organized as a series of activities in a continuous evaluation context. Many of these activities used the online judging system; some of them were done individually, while others were done collaboratively by groups of students. These three elements (online judging, continuous evaluation and collaborative work) were the keys to generating motivation and enthusiasm among students. The effect of that experience was an important descent on the dropout rates and a general improvement on marks and pass rates [16]. The amount of data collected was a valuable material to be used in decision-making processes. In fact, since the academic performance and future professional achievements are considered to be related concepts, analysis of the success factors behind online collaborative programming activities may help to understand and improve performance in team programming within a large programming project.

During the last several years, improvements in the Artificial Intelligence (AI) methods have made it possible to develop expert systems and Decision Support Systems (DSS) in many different fields such as business, health, psychology, environmental science and education [17,18]. Among the good classifiers in the AI field, we highlight Artificial Neural Networks (ANNs) and Decision Trees (DT) as the most widely chosen for the construction of DSS [19,20]. In many cases, the goal is to establish groups or clusters with the data that have similar features. These cases are unsupervised since there are no references or classification expected and, therefore, these cases are data-driven insight.

We have previously demonstrated that AI methods are capable of improving the accuracy of the final classification as well as selecting the best features as, very often, the number of features to deal with is huge. For instance, we have experienced this in classification tasks for male fertility [21,22] and urology diagnosis [23,24]. This will lead to the knowledge discovery in databases, data mining or the process of extracting patterns from large data sets.

As mentioned above, the online judging system named Mooshak [25] was used to evaluate the work of the students. This learning environment is a free automatic tool to evaluate the correctness of computer programs. Although this system was designed to support programming competitions,

it has also been used in both the evaluation of programs in computer programming courses and the assessment of test exams [16,26–28]. Students, instructors and system administrators have different web-based interfaces. The instructor introduces a set of pairs (program input, expected output from the program) for each problem proposed in Mooshak. This set is used in the correctness of the computer programs submitted for that problem by the students. An important advantage of using the online judge is the availability of a lot of information about the submissions made by the students, which can be analyzed in real time by the teachers.

Educational data mining (EDM) is an emerging research field aiming at discovering knowledge, making decisions, and providing recommendations. In spite of education being a novel data mining application, a recent review identified 240 educational data mining works describing 222 EDM approaches and 18 EDM tools between 2010 and the first quarter of 2013 [29]. Previous reviews of EDM [30,31] identified more than 250 studies carried out between 1973 and 2010. Studies were classified in various educational categories (e.g., analysis and visualization of data, providing feedback for supporting, instruction recommendations for students, predicting students' performance, student modeling, student behavior modeling, student assessment, grouping students) and different variants of e-learning systems (e.g., web-based courses, learning management systems, and adaptive and intelligent web-based educational systems).

Regarding Green Computer Science as part of ICT and sustainable development [32], regrettably ICT sustainability has not been widely addressed in education [5]. Pattinson highlights that "ICT sustainability does not feature more strongly in school or University ICT curricula, and that, where it does appear, it is often a sub-part of a broader topic" [5]. These are some examples that are exceptions where sustainability is taken into account and the benefits are explained in every work. For instance, the objective of the paper presented by [33] is inserted into a wider finality: to provide recommendations regarding the redesign of pre-service teacher training curricula and learning programmes.

In [34], they used data from 2413 students in grades 6, 9, and 12 from 51 schools across Sweden to study the effectiveness of Education for Sustainable Development (ESD). The results of this study reveal the key role ESD plays in addressing sustainable development (SD), paving the way for a more sustainable future.

The review [35] analysis has emphasised the lack of environmental competences amongst pre-service teacher students and the gaps in the teacher training curriculums regarding environmental education (EE). The overall scarcity of research in this area, jointly with certain gaps and methodological limitations, affirms the need for strengthening the evidence base. Although sustainability is one of the greatest problems to affect humanity nowadays, it does not appear to be receiving the coverage that it might deserve in education [5]. The attention given to ICT sustainability as a subject in the education curriculum is insufficient and sustainability should be considered "a core element of the teaching, education and research in ICT" [5]. However, there exist some initiatives, such as a green computer science program [36] or a master program in green ICT [37].

Two kinds of works have been carried out in EDM: approaches based on predictive models and approaches based on descriptive models. Predictive approaches, which depict around 60% of all of the studies proposed in EDM in the last four years (e.g., [38,39]), usually employ supervised learning functions to estimate unknown values of dependent variables based on the values of related independent variables [40]. In contrast, descriptive models (e.g., [41,42]) frequently use unsupervised learning functions to obtain patterns that explain the structure and relations of the mined data [43]. The implementation of a model is developed by a task. For example, clustering [38,41], association rules [44] and correlation analysis [42] generate descriptive models, whereas classification [45], regression [39] and categorization [46] produce predictive models. Most of the studies carried out in the last four years use classification and clustering for EDM. After some task is chosen, there are many methods and techniques to build the approach. Bayes theorem and DT [38,45], instances-based learning [47] and hidden Markov model [46] are the top-four most used methods in

EDM, whereas LR [39], frequencies [39] and hierarchical clustering [47] are the most used techniques in EDM. Lastly, an algorithm, equation, and/or frame are implemented to mine the source data. K-means, expectation maximization, J48, and Naive Bayes are the most deployed algorithms [38,47]. Statistical equations, including descriptive [41] are the most popular equations, and several versions of Bayesian networks [45] are the most used frames.

Several studies have compared the accuracy of data mining techniques [48,49]. Data mining techniques were employed to compare the achievements of Computer Engineering Department students in Karabük University [49]. Age, gender, type of high school graduation and whether the students were studying in distance education or regular education were the parameters studied. Two prediction/classification methods, ANN, and DT were used and compared to each other in order to study the achievements of Computer Engineering Department students in Karabük University [49]. DT algorithms produced the best prediction results with 97.8% overall accuracy in comparison with 94.4% for ANN.

Various EDM studies have investigated which type of modules or features influence students in learning collaboratively [50]. Data mining techniques were used to classify chat messages in online communication with the aim of providing learners with real-time adaptive feedback while learning collaboratively [51]. A frequent sequential pattern mining algorithm was used to find the patterns characterizing some aspects of teamwork, and to identify significant sequences indicative of problems in teams of five to seven students [52]. Three types of events (wiki event, ticket event and SVN event) reflecting the students learning process were analyzed to draw conclusions from the traces of their collaborations. Clustering technique on data collected in a software development project course was applied to characterize the collaborative work of stronger and weaker students [53]. In this work. seven groups of five to seven students (43 students in total) were clustered based on 11 attributes that seemed to capture essential aspects of team performance (e.g., average number of lines deleted per wiki edit).

To the best of our knowledge, no previous data mining studies have analyzed such large data sets from an online judgement tool. The only previous work found in the literature was statistical analyses (no an artificial intelligent technique) of 89,402 programs written to 60 specifications in an online judge, which was carried out to study the effectiveness of software diversity [54]. The results of our empirical study have been used to identify factors which have an effect on students' performance. It will also help address future directions as regards improving students' knowledge and skills in programming courses.

The aim of this paper is to identify the main factors to predict program correctness, practice grades of students and the use of computers by students by using a data mining approach. These identified factors allow us to analyze the reasons why students are consuming electrical resources such as computers, monitors, and so on. The consumption of computers has become an issue because of high electricity costs and problems with power supply capacity. In many cases, students use computers inefficiently because computers draw around 70–90% of its maximum power usage even when doing no useful work [8]. Consequently, the objective is to generalize the lesson learned for good practices of students in Education for Sustainable Development (ESD).

One of the novel aspects of our proposal is the use of big data provided by online judgement systems to obtain the predictor variables. In our approach, we make use of AI methods in order to tackle this problem. In particular, we use four supervised techniques: Logistic Regression (LR), DT, Multilayer Perceptron (MLP) and Support Vector Machines (SVM). Some of the AI techniques can indicate rules and correlations among input variables and the predictable variables. This will help to identify the key factors to predict the program correctness, and thus the programmer's performance.

The remaining part of the paper is organized as follows. In Section 2, we start by defining the materials and methods of the study (samples of the study). Section 3 shows the results of experiments carried out. We then proceed by providing available data as well as a detailed explanation of the

different values of our database. Finally, Section 4 analyzes the results, draws the relevant conclusions and presents future work.

2. Materials and Methods

The procedure followed in this work is summarized in Figure 1. This scheme is represented in three steps:

1. Step 1: Among the different AI techniques, we choose the four methods LR, DT, MLP and SVM.
2. These four methods LR, DT, MLP and SVM are the basis to construct the model.
3. Finally, the model is built through the training model and then it will be capable to predict/classify.

Figure 1. Methodology employed in this work.

The machine learning software package used in our experiments is WEKA [55]. It includes the most efficient AI algorithms, such as LR, C4.5, backpropagation and SVM, among others.

2.1. Data Description

The information used in our analysis was collected along a period of seven academic years, from 2008/09 to 2013/14 with similar groups of practices. It has been possible to analyze this period as these groups have similar features. The thematic and the difficulty of the proposed problems in the practices did not vary from year to year. During this long period, a change in the curriculum took place motivated by the implantation of the EHEA. Thus, the first three years of the study correspond to the former Computer Science Engineering (CSE) and the last four to the new Degree in Computer Science (DCS). From the total of 505 students analyzed, 137 (27.1%) are from CSE and 368 (72.9%) from DCS.

Students were recruited from a course on ADS which was included both in CSE and DCS, with a very similar content, teaching methodology and schedule of activities. In a certain sense, ADS teachers anticipated the methodological changes in the subject, so the results of all years are comparable. Basically, ADS is a course in advanced programming concepts, which stresses issues of algorithms and data representation. This course introduces topics such as data structures, abstract data types, formal specifications and graph theory. It is a second-year course, which takes place after two introductory programing courses in the first year. The programming languages used to illustrate the concepts studied are C and C++, and Maude as a formal specification language. Both in CSE and DCS, ADS are organized as weekly lectures along a semester, laboratory sessions, a programming project and a final exam.

More specifically, the course consists of four topics: T1—Formal specifications; T2—Sets and maps with hash tables; T3—Sets and maps with trees; T4—Graph representation and problems. Each of these topics is evaluated in a continuous evaluation context, by means of a specific activity. T1 is evaluated with a practical activity in the online judge using Maude. T2 and T3 are assessed by means of partial exams in the middle of the semester. In addition, T4 is carried out with a programming activity in C/C++ to solve some graph problems using the online judge.

In addition, the programming project is a compulsory activity where students have to face a problem of higher difficulty related to the concepts studied in the course, but mainly related to topics T2 and T3. The project to solve varies from year to year; for example, in the 2008/09 course, the students had to develop the internal engine of a spellchecker. The activity is decomposed into a series of exercises that have to be programed in C or C++. These exercises are evaluated in the online judge.

They present an increasing degree of difficulty, from basic input/output format until reaching the last problem corresponding to the whole project. The project has to be done cooperatively by a group of two students in a co-located software development environment [56]. However, in some cases, students are allowed to do the activity individually (for example, if they do not find a classmate, or if their mates drop out).

The information collected for the present work corresponds to the submissions made by the students to the programming project. This includes submission-specific data such as time and date of the submission, number of the exercise, memory and time consumed by the program, and the result of the evaluation (whether the program is accepted or not). These data are augmented with information about the students, and the context where the submissions are done such as gender, marks of the students in the activities of the course, number of times the student has been previously enrolled in ADS, and whether the project is done individually or in a group of two. Table 1 presents the list of the variables used in this study.

Table 1. Input variables.

Variable Name	Data Type	Description
NUMBER_ENROLL	Number	Number of times the student has been previously enrolled in ADS
NUMBER_FAIL	Number	Number of times the student has failed ADS in previous enrollments (each enrollment entitles a maximum of three evaluations)
GENDER	Binary	Student's gender
ABSENT	Number	Number of absences of the student to the weekly lectures
M_T1..M_T4	Number	Mark of the student in the activities of topics T1, T2, T3 and T4
PRACTICE	Number	Mark of the student practices
THEORY	Number	Mark of the student theory (mean of marks of T1...T4)
GROUP_SIZE	Binary	Whether the project is done individually or in a group of two
TURN	Text	Turn in with the student is studying (morning or afternoon turns)
EXERCISE	Number	Number of the exercise of the submission (numbers 0XX are basic input/output problems, 2XX are related to topic 2, and 3XX to topic 3)
LANGUAGE	Text	Programming language used (C or C++)
PROG_SIZE	Number	Size of the program submitted
RUNNING TIME	Number	Execution time consumed by the program in the online judge tests (in ms)
MEMORY	Number	Memory used by the program in the online judge tests (in bytes)
SUBMIT_TIME	Number	Time when the submission is done
WEEK_DAY	Number	Day of the week when the submission is done, from Sunday (0) to Saturday (6)
DAYS_DEADLINE	Number	Number of days until the deadline of the activity
GRADE	Number	Final mark of the student
YEAR	Number	Academic course
EVALUATION	Text	Evaluation of the program done by the online judge (accepted or not accepted). This is the output variable

2.2. AI Methods

2.2.1. Logistic Regression

Probably one of the most commonly used techniques for prediction is LR, where, in many cases, it is compared with other usual techniques such as ANN and DT [57–60]. It is a multivariable method which means that it tries to establish a functional relationship between input data (predictors or independent variables) and one output (outcome, dependent variable); generally speaking, the outcome variable of a LR is categorical [61]. For the aim of this work, we will focus on both binary LR which could predict membership of only two categorical outcomes (acceptance of a program, correct or not) as well as general LR to predict four outcomes (grades A, B, C and D). The grades are defined as follows: A = mark in [10, 9]; B = mark in (9, 7]; C = mark in (7, 5]; and D = mark in (5, 0].

The LR model identifies the probability of the default event occurring as the following Formula (1):

$$P(Y) = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + \ldots)}}, \quad (1)$$

where $P(Y)$ is the probability of Y occurring, which can be also expressed Y belonging to a certain class. x_n are predictor variables and b_n are coefficient to be determined by the LR.

2.2.2. Decision Trees

A DT classifier [62] is represented in the form of a tree structure; each node is either a decision node, which specifies some test to be carried out on a single attribute-value, with one branch and sub-tree for each possible outcome of the test, or a leaf node, which indicates the value of the target attribute (class) of examples.

Among the tools for classification and prediction, DT is one of the most powerful and popular. In opposition to other AI methods, such as MLP, the main advantage of DT is due to the fact that they represent rules. Rules can be readily expressed so that persons can understand them or even directly used in a database. For further information about this advantage with a categorization, we can see a classical and recurrent example in the bibliography: the forecasting prediction to play tennis [63].

For some applications, the accuracy of a classification or prediction is the most important result. In such situations, the user is not necessarily concerned about how the model works. However, in other situations, the ability to explain the reason for a decision is critical.

DT provides a good explanation applicable to the life habits of the population and therefore interprets problems very much according to the principles of mathematical and statistical principles [64]. A DT may be used to classify a given example; one begins at the root and follows the path provided by the answers to the questions in the internal nodes until a leaf is reached. The DT algorithm family includes well-known algorithms, such as CART [65], ID3 [66], and C4.5 [67]. The algorithm chosen in our study is C4.5.

The classification and regression trees (CART or C&RT) method of Breiman, Friedman, Olshen, and Stone [65] generates binary DT. In the real world, the chances of biased binary outcomes are few, but the binary method facilitates interpretation and analysis [68–70]. Therefore, our study employs the binary method in the DT to classify the programs implemented for the students. A simple representation of DT consists of two types of nodes:

- Decision nodes: Usually represented by circles. From the circles appear the arcs with the diverse decisions.
- Leaf nodes: Represented by squares.

DT are commonly used in decision analysis in order to help identify a strategy most likely to reach a goal. Another use of DT is as a descriptive means for calculating conditional probabilities.

2.2.3. ANN—Multilayer Perceptron

An MLP [71–73] is composed of three or more layers of neurons; each layer is fully connected to the next one. Usually, there are three types of layers (see Figure 2):

- An input layer receives external inputs.
- One or more hidden layers transform the inputs into something useful for the output layer.
- Finally, an output layer generates the classification results.

Every neuron in the output and hidden layers is a computational element with a nonlinear activation function. The basis of the network is that when data are introduced at the input layer, the network neurons perform calculations in the following layers until an output value is obtained at each of the output neurons. The final output provides an appropriate class for the input data.

In an MLP, every neuron in the input and the hidden layers is connected to all neurons in the next layer by weighted connections. The neurons of the hidden layers compute weighted sums of their inputs and they add a threshold. The activity of the neurons is calculated from the resulting sums by applying a sigmoid activation function.

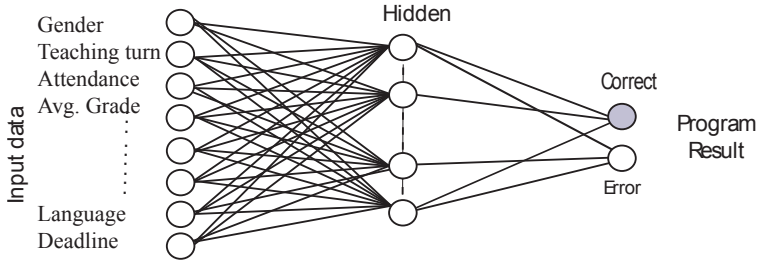

Figure 2. The architecture of the MLP network consists of input layer, hidden layer and output layer. The input layer represents the input data (the input data set is described in Section 2.1). The output layer represents the classification result and it contains as many outputs as the number of classes in a particular problem. The hidden layer is calculated in the experimentation.

The learning process is defined in the following formula:

$$v_j = \sum_{i=1}^{p} w_{ji} x_i + \theta_j \quad , \qquad y_j = f_j(v_j), \tag{2}$$

where:

- p is the number of inputs,
- v_j is the linear combination of inputs $x_1, x_2, ..., x_p$,
- the threshold θ_j, w_{ji} is the connection weight between the input x_i and the neuron j,
- and f_j is the activation function of the j_{th} neuron, and y_j is the output.

The sigmoid function is a common choice as an activation function. This mathematical function is defined as:

$$f(t) = \frac{1}{1 + e^{-t}}. \tag{3}$$

In an MLP, a single neuron is able to linearly separate its input space into two subspaces by a hyperplane defined by the weights and the threshold: the weights define the direction of this hyperplane and the threshold term θ_j offsets it from the origin.

In our proposal, a supervised learning method denominated backpropagation [74] is used to train the MLP. This method is used in conjunction with an optimization method such as gradient descent, for the adaptation of the weights. In this method, the network is presented with input examples as well as the complementary desired output. The following steps are carried out in this algorithm:

1. All the weight vectors w are initialized with small random values from a pseudorandom sequence generator.
2. Three basic steps are repeated until the convergence is achieved, that is, the error E is below a preset value.

    - The weight vectors w_i are updated by

$$w(t+1) = w(t) + \Delta w(t), \tag{4}$$

    where

$$\Delta w(t) = -\eta \partial E(t)/\partial w. \tag{5}$$

    - Compute the error $E(t+1)$,

    where t is the iteration number, w is the weight vector, and η is the learning rate.

The error function E is minimized by adapting the weights and the thresholds of the neurons:

$$E = \frac{1}{2}\sum_{p=1}^{n}(d_p - y_p)^2. \tag{6}$$

In the error function, y_p is the actual output and d_p is the desired output for input pattern p.

The minimization of E can be accomplished by gradient descent, a process where the weights are adjusted to change the value of E in the direction of its negative gradient.

For the construction of the three-layer MLP architecture, we can conclude that layers 1 and 3 are the easiest ones. Layer 1 corresponds directly to the input vector and layer 3 is the output layer with two outputs for classification: correct and error program output. Layer 2 (the hidden layer) consists of the number of hidden neurons, which means the most advanced question in the network's architecture. The choice of the number of neurons in a hidden layer leads to a trade-off between performance and the risk of overfitting [71]. The number of hidden neurons will significantly have effects on the ability of the network to generalize from the training data to unknown examples [75]. Findings of the experiments provided evidence that a low number of neurons for this layer leads to a poor performance for both training and test sets. At the opposite edge, a high number of neurons keep a good generalization and consequently a high accuracy, for training and test sets, although the risk of overfitting is high [76]. The tests carried out between these two options leads to meet the optimal solution for this layer with eight neurons (see Figure 2).

2.2.4. Support Vector Machines

Although the foundations of SVM are dated back to 1963, current SVM was proposed in a seminal paper presented in 1992 [77]. A detailed description of SVM can be found in [78,79]. A typical two class problem is similar to the problem of predicting programs as either correct or incorrect.

In a two class classification problem, the estimation of a function $f : \Re^N \rightarrow \{\pm 1\}$ using training data is required. These data are l N-dimensional patterns x_i and class labels y_i, where

$$(x_1, y_1), ..., (x_l, y_l) \in \Re^N \times \{\pm 1\}, \tag{7}$$

such that f will classify new samples (x, y) correctly.

The SVM classifier, as described by [80,81], satisfies the following conditions:

$$\begin{cases} \mathbf{w}^T\varphi(x_i) + b \geq +1 & if \quad y_i = +1, \\ \mathbf{w}^T\varphi(x_i) + b \leq -1 & if \quad y_i = -1, \end{cases} \tag{8}$$

which is equivalent to

$$y_i[\mathbf{w}^T\varphi(x_i) + b] \geq 1, \quad i = 1, 2, ..., l. \tag{9}$$

Here, training vectors x_i are mapped into a higher dimensional space by the function φ. The equations of Label (8) construct a hyperplane $\mathbf{w}^T\varphi(x_i) + b = 0$ in this higher dimensional space that discriminates between the two classes. Each of the two half-spaces defined by this hyperplane corresponds to one class, H_1 for $y_i = +1$ and H_2 for $y_i = -1$. The SVM classifier therefore corresponds to decision functions:

$$y(x) = sign[\mathbf{w}^T\varphi(x_i) + b]. \tag{10}$$

The SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. The margin of a linear classifier is the minimal distance of any training point to the hyperplane which is the distance between H_1 and H_2.

2.2.5. Evaluation

We have used a cross validation method, which is widely used to ensure the generalization of the AI methods. A k-fold cross-validation method has been applied for the performance assessment of network. This experimental design method, also called k rotation estimation, has been widely used to minimize the bias associated with the random sampling of the data set:

$$Acc_{cv} = \frac{1}{n}\sum_{x_i \in S}\delta(I(S_i, x_i), y_i), \tag{11}$$

where:

- n is the size of the dataset S,
- x_i is the example of S,
- y_i is the target of x_i,
- and S_i is the probable target of x_i by the classifier function I.

Therefore, it is concluded that:

$$\delta(i, j) = \begin{cases} 1, & if \quad i = j, \\ 0, & otherwise. \end{cases} \tag{12}$$

In our study, the value of k is set to 10, and consequently a ten-fold cross-validation method has been applied for the performance assessment of the model. The data has been divided into ten sets (S_1, S_2, ..., S_{10}).

3. Results

In this section, we expose the results obtained starting from the experiments carried out. In problems of classification in multiple classes, the confusion matrix can be a very useful tool to look into the strengths and weaknesses of the classifiers. This matrix relates the real classes of the samples with the classes predicted by the system. The prediction results by applying a 10-fold cross validation to the four methods used are presented in Table 2. The values of the confusion matrix are accepted and not accepted, according to the result of the program output. The letter "A" means accepted program in Mooshak and the letter "R" means rejected program in Mooshak. Two blocks of results are shown: collaborative programming activities and individual programming activities. Since the output variable had two nominal values, the confusion matrix shows a 2 × 2 square matrix where the correct predictions (the shaded cells) are placed at the diagonal from the upper left to the lower right corner. The rows of the confusion matrixes represent the actual and the columns represent the predictions. The four AI methods were evaluated by obtaining the following measures: classification accuracy, sensitivity, specificity, positive predictive value and negative predictive value (Table 3).

DT obtained the best confusion matrix. Time and memory efficiency were identified as the most important predictor variables as shown in Figure 3. This, as it has been aforementioned, is the purpose of this paper: to find the correlation between the grades and other variables. Accordingly, this fact will lead us to find indicators that allow us to reduce the consumption of electrical devices by students when developing practical works.

Figure 4 shows the plot of the execution times of the last submission for each collaborative group, as a function of the final mark in the practice. Execution times are relative to the best time for each year. In order to quantify the correlation, a linear model is fitted using the least squares method. The result proves the inverse relation between both variables (the shorter time the better mark). In any case, the parameter that indicates the goodness of fit (R2) has a very low value of 0.0555. This fact indicates that time efficiency is not the only parameter in the evaluation of collaborative work.

Regarding memory efficiency, Figure 5 shows the plot of the memory consumed by the last submission for each collaborative group, as a function of the final mark in the practice. Memory is relative to the best memory usage for each year. The correlation between memory and mark has also been studied with the fitting of a linear model using the least squares method, obtaining a negative slope. The R2 parameter, with value 0.0533, shows once again that the mark is not explained only with the memory efficiency, although it is a very important factor.

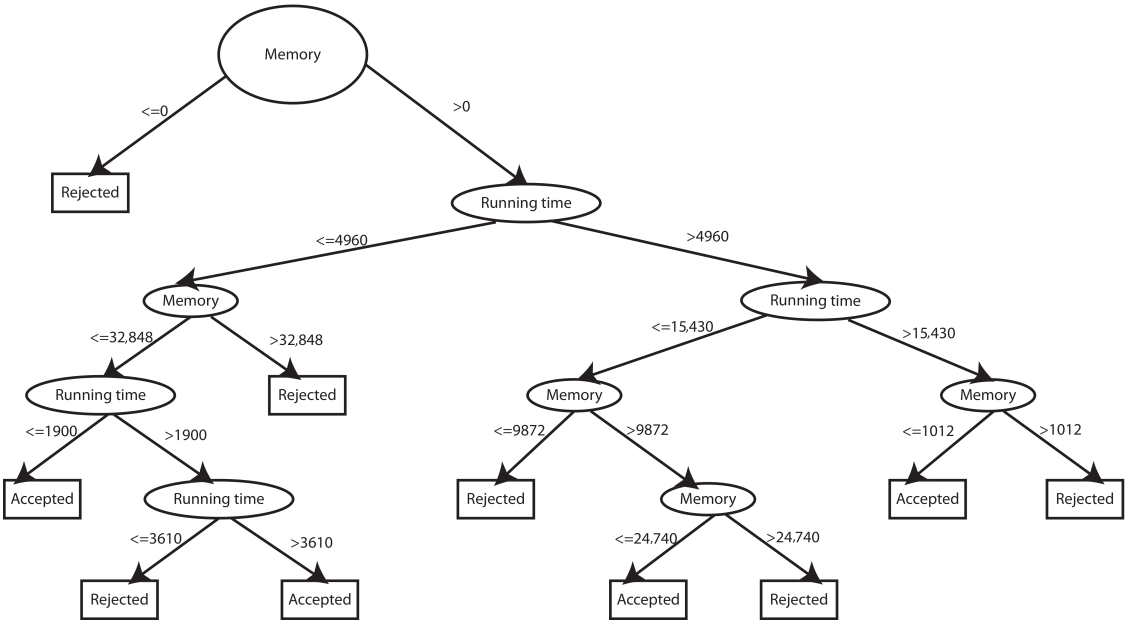

Figure 3. The experimentation carried out with a DT helps to predict the acceptance of a program.

Table 2. Definition of the confusion matrix MLP, SVM, DT and LR classifiers.

	DT		**MLP**		**SVM**		**LR**	
Actual	**Predicted**		**Predicted**		**Predicted**		**Predicted**	
Collaborative programming activities								
	A	R	A	R	A	R	A	R
A	942	304	809	437	916	330	894	352
R	351	734	487	598	538	547	493	592
Individual programming activities								
	A	R	A	R	A	R	A	R
A	1034	492	769	757	359	1167	511	1015
R	461	1708	716	1453	279	1890	427	1742

Table 3. Equations according to the DT, MLP, SVM and LR classifiers.

		DT	MLP	SVM	LR
Collaborative programming activities					
Class. acc. (%)	$\frac{TP+TN}{TP+FP+FN+TN} \times 100 =$	71.9%	60.4%	62.8%	63.7%
Sensitivity (%)	$\frac{TP}{TP+FN} \times 100 =$	75.6%	64.9%	73.5%	71.7%
Specificity (%)	$\frac{TN}{FP+TN} \times 100 =$	67.6%	55.1%	50.4%	54.6%
Pos. pred. (%)	$\frac{TP}{TP+FP} \times 100 =$	72.9%	62.4%	63.0%	64.5%
Neg. pred. (%)	$\frac{TN}{FN+TN} \times 100 =$	70.7%	57.8%	62.4%	62.7%
Individual programming activities					
Class. acc. (%)	$\frac{TP+TN}{TP+FP+FN+TN} \times 100 =$	74.2%	60.1%	60.9%	61.0%
Sensitivity (%)	$\frac{TP}{TP+FN} \times 100 =$	67.8%	50.4%	23.5%	33.5%
Specificity (%)	$\frac{TN}{FP+TN} \times 100 =$	78.7%	67.0%	87.1%	80.3%
Pos. pred. (%)	$\frac{TP}{TP+FP} \times 100 =$	69.2%	51.8%	56.3%	54.5%
Neg. pred. (%)	$\frac{TN}{FN+TN} \times 100 =$	77.6%	65.7%	61.8%	63.2%

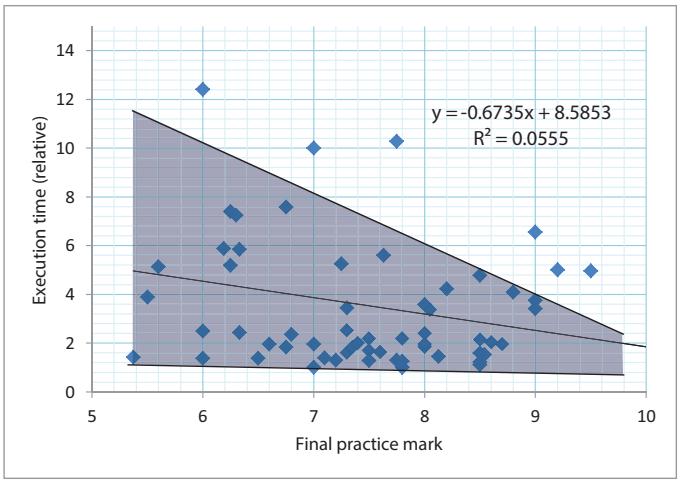

Figure 4. Final mark of the practice of each collaborative group. Execution times are relative to the best time for each year.

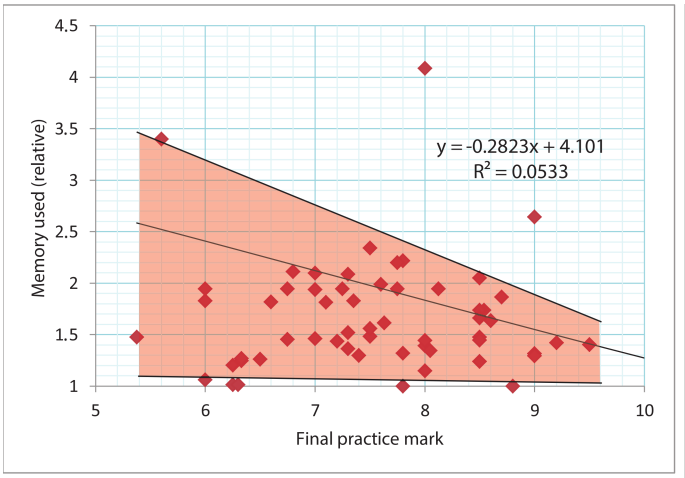

Figure 5. Final mark of the practice of each collaborative group with respect to memory consumption of the last submission of the group to the last problem (which contains the whole practice). Memory is relative to the best memory usage for each year.

Figure 6 shows two sample fragments of source code in C++, exemplifying the inverse relationship between execution time and code correction. Both samples correspond to the implementation of the hashing function in a hash table class. The sample on the left is a good choice for the function, using an iterative multiplicative method with base 17. However, the sample on the right is an incorrect implementation; the use of *pow* function produces an overflow, so the result is saturated to the *max* integer. Therefore, the code on the left takes 0.456 s on the test case for this problem, while the code on the right takes 4.964 s.

```
int TablaHash::Hash(string URL) {
   int n=0;

   for(int i=0; i<URL.length(); i++)
      n=n*17+URL[i];

   return abs(n)%M;
}
```

```
int h(string url, int m)
{
   unsigned int res=0;
   for(int i=0;i<url.length();i++)
   {
      unsigned int n=url[i];
      res+=n*pow(3,i);
   }
   return res%m;
}
```

Figure 6. Two fragments of sample of code showing good (left) and bad (right) programming practices.

4. Discussion

4.1. Academic Findings

Teaching methods have been changing dramatically in recent years. For example, the appearance of Massive Open Online Courses (MOOCs) offers many advantages but also presents new challenges. The correlations obtained among input variables and the predictable variable help to identify the key factors to predict the performance in the practices of the students. In a MOOC, there are thousands of students and a teacher cannot track each student individually as in traditional methods. For this reason, techniques that have been successfully applied in other environments (e.g., Biology, Internet of Things and so on) are of particular interest in these new educational scenarios, even more so when dealing with a huge amount of data.

One of the contributions of this work is to lay the foundation for automatic prediction of the programmer's performance by using data provided by online judgement systems. Two key factors associated with green software were identified to predict the programmer's competence: execution time and memory used. However, we have not found the programming language as a key factor in determining the correctness of the programs. This finding contrasts with the fact that previous research revealed that errors made by programmers may depend on programming language [54]. These results provide insights into computer programming learning. It means that, if we teach how to develop green software (with low memory and time consumption), we will improve the performance of the programmer as he will acquire good practices to implement correct programs. While performance considerations are a key aspect of many software systems, undergraduate curricula do not always explicitly teach students about tools and techniques for improving software performance. Although this topic is included in courses designed in the IEEE/ACM 2013 Curriculum Guidelines for Undergraduate Degree Programs in Computer Science [82], the topic is sometimes addressed incidentally by issues ranging from architectural matters in machine organization courses, to asymptotic complexity in algorithm design courses.

As shown in our study, when teaching students how to design programs with performance issues in mind, as well as to identify and rectify performance bottlenecks, it can lead novice programmers to

develop correct programs. Notice that using design patters can help to reuse efficient solutions [83]. Design patterns describe high quality practical solutions to common programming problems. Students should understand the principles of good program design to achieve a clearer system architecture, thus reducing the resource consumption (memory usage and execution time). However, design patterns are used only marginally in programming courses [84].

4.2. Educational Data Mining Findings

The aim of this paper has been to apply four data mining techniques (DT, MLP, SVM and LR) to predict students' results in a learning environment, in particular, the prediction of the program correction and the practice grades. Our findings show that DT is the best predictor as compared with MLP, SVM and LR classifiers. Nevertheless, factors such as efficiency (time taken to build a model) and understanding (easy of interpreting the developed model) should be also considered according to the particular application context. When choosing a method over another for a prediction problem, in addition to prediction accuracy, we should also consider factors like efficiency (time it takes to build a model), interpretability (ease of understanding of the developed model), deployability (ease of deploying the model for actual use) and theoretical justification.

Particularly, we would like to emphasize the high percentages of all the measures (classification accuracy, sensitivity, specificity, positive predictive value and negative predictive value) for the Decision Trees. The rest of the techniques obtain good percentages but around 10% lower. This indicates how suitable the Decision Trees are for the prediction in this environment.

4.3. Sustainability Findings

ICT for sustainability is an emerging research and educational area that encompasses different fields [85]: Environmental Informatics, Computational Sustainability, Sustainable HCI, Green ICT, and ICT for Sustainability. Energy consumption of data centers is rapidly becoming an environmental problem [86]: in 2016, global data centers used roughly 416 terawatts or about 3% of the total electricity consumed, approximately 40% more than the entire electricity consumed in United Kingdom. The forecast is that this consumption will double every four years. In this paper, we have explored how EDM can be used to support Green ICT: how to reduce the environmental impacts of ICT hardware and software. The improvements in power effectiveness of data center have mainly focused on new data center designs and more efficient components [87]. However, theoretically, power usage effectiveness is approaching, future efficiency gains will be minor and new approaches are needed. For example, most Green ICT practices do not yet particularly tackle software-specific aspects to improve sustainability [88] as we have proposed in this paper. In recent times, artificial intelligence has been employed for reducing power consumption in data centers [89]. In a MOOC, where thousands of students can be accessing the course at the same, the power consumed by the data centers where the course is hosted can be very significant. The data mining approach presented in this paper to predict program correctness as well as the grades of the students' practices helps to reduce the usage of data centers that support a MOOC. Data centers used efficiently and effectively, with less memory and time consumption, means minimal impact to the environment.

As future work, we will apply new data mining techniques (such as machine learning based on big data or deep learning) to the critical problem of automatic assessment of students to find stronger evidence on the impact of green software on program correctness. In addition, we will be able to generalize these achievements in order to analyze other factors associated with green software such as pattern design [90] to identify new predictor variables that affect program correctness. We hope that our future findings help MOOC teachers and organizers to better know students and be able to improve their education programs for sustainable development.

Author Contributions: D.G. contributed to the following: the conception and design of the study, acquisition of data, analysis, and interpretation of data, drafting the article and approval of the submitted version. The authors J.L.F.-A. and G.G.-M. contributed to the study by design, conception, interpretation of data, and critical revision.

S.L.-M. made the following contributions to the study: analysis and interpretation of data, drafting the article and approval of the submitted version. The authors J.T. and A.T. contributed to the study by its critical revision. All authors read and approved the final manuscript.

Funding: This work has been funded by the Spanish Ministry of Economy and Competitiveness (MINECO/FEDER) under the granted project SEQUOIA-UA (TIN2015-63502-C3-3-R), project GINSENG-UMU (TIN2015-70259-C2-2-R) supported by the Spanish Ministry of Economy, Industry and Competitiveness and European FEDER funds. This work has also been partially funded by University of Alicante, under project GRE14-10 and by the Generalitat Valenciana, Spain, under project GV/2016/087.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. United Nations. Transforming Our World: The 2030 Agenda for Sustainable Development. 2015. Available online: https://sustainabledevelopment.un.org/post2015/transformingourworld/publication (accessed on 22 August 2018).
2. Mohamed, M.; Murray, A.; Mohamed, M. The role of information and communication technology (ICT) in mobilization of sustainable development knowledge: A quantitative evaluation. *J. Knowl. Manag.* **2010**, *14*, 744–758, doi:10.1108/13673271011074872. [CrossRef]
3. Tjoa, A.M.; Tjoa, S. The Role of ICT to Achieve the UN Sustainable Development Goals (SDG). In Proceedings of the 6th IFIP World Information Technology Forum (WITFOR), San José, Costa Rica, 12–14 September 2016; pp. 3–13, doi:10.1007/978-3-319-44447-5_1. [CrossRef]
4. Lehr, W. Why ICTs are Critical for Sustainable Development. 2018. Available online: https://news.itu.int/icts-are-critical-for-sustainable-development/ (accessed on 22 August 2018).
5. Pattinson, C. ICT and Green Sustainability Research and Teaching. *IFAC-PapersOnLine* **2017**, *50*, 12938–12943, doi:10.1016/j.ifacol.2017.08.1794. [CrossRef]
6. Avgerinou, M.; Bertoldi, P.; Castellazzi, L. Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency. *Energies* **2017**, *10*, 1470, doi:10.3390/en10101470. [CrossRef]
7. Gartner. Gartner Estimates ICT Industry Accounts for 2 Percent of Global CO_2 Emissions. 2007. Available online: https://www.gartner.com/newsroom/id/503867 (accessed on 22 August 2018).
8. UK Parliamentary Office of Science and Technology. ICT and CO_2 Emissions. 2008. Available online: https://www.parliament.uk/documents/post/postpn319.pdf (accessed on 22 August 2018).
9. Malmodin, J.; Lundén, D. The Energy and Carbon Footprint of the Global ICT and E&M Sectors 2010–2015. *Sustainability* **2018**, *10*, 3027.
10. Belkhir, L.; Elmeligi, A. Assessing ICT global emissions footprint: Trends to 2040 & recommendations. *J. Clean. Prod.* **2018**, *177*, 448–463.
11. Michanan, J.; Dewri, R.; Rutherford, M.J. GreenC5: An adaptive, energy-aware collection for green software development. *Sustain. Comput. Inform. Syst.* **2017**, *13*, 42–60. [CrossRef]
12. Kern, E.; Hilty, L.M.; Guldner, A.; Maksimov, Y.V.; Filler, A.; Gröger, J.; Naumann, S. Sustainable software products—Towards assessment criteria for resource and energy efficiency. *Future Gener. Comput. Syst.* **2018**, *86*, 199–210. [CrossRef]
13. European Ministers in charge of Higher Education. The Bologna Declaration of 19 June 1999: Joint Declaration of the European Ministers of Education. 1999. Available online: https://www.eurashe.eu/library/bologna_1999_bologna-declaration-pdf/ (accessed on 22 August 2018).
14. Schwarz, S.; Westerheijden, D.F. (Eds.) *Accreditation and Evaluation in the European Higher Education Area*; Springer: Dordrecht, The Netherlands, 2004; Volume 5.
15. Garcia-Mateos, G.; Fernandez-Aleman, J.L. Make Learning Fun with Programming Contests. In *Transactions on Edutainment II*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 246–257.
16. García-Mateos, G.; Fernández Alemán, J. A course on algorithms and data structures using online judging. In Proceedings of the 14th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE), Paris, France, 6–9 July 2009; pp. 45–49.
17. Sang, X.; Liu, X.; Qin, J. An analytical solution to fuzzy TOPSIS and its application in personnel selection for knowledge-intensive enterprise. *Appl. Soft Comput.* **2015**, *30*, 190–204. [CrossRef]

18. Hamedi, M.; Salleh, S.H.; Noor, A.M. Facial neuromuscular signal classification by means of least square support vector machine for MuCI. *Appl. Soft Comput.* **2015**, *30*, 83–93. [CrossRef]
19. Paliwal, M.; Kumar, U.A. Neural networks and statistical techniques: A review of applications. *Expert Syst. Appl.* **2009**, *36*, 2–17. [CrossRef]
20. Kumar, D.S.; Sathyadevi, G.; Sivanesh, S. Decision Support System for Medical Diagnosis Using Data Mining. *Int. J. Comput. Sci. Issues* **2011**, *8*, 147.
21. Gil, D.; Girela, J.L.; De Juan, J.; Gomez-Torres, M.J.; Johnsson, M. Predicting seminal quality with artificial intelligence methods. *Expert Syst. Appl.* **2012**, *39*, 12564–12573. [CrossRef]
22. Girela, J.L.; Gil, D.; Johnsson, M.; Gomez-Torres, M.J.; De Juan, J. Semen Parameters Can Be Predicted from Environmental Factors and Lifestyle Using Artificial Intelligence Methods. *Biol. Reprod.* **2013**, *88*, 1–8. [CrossRef] [PubMed]
23. Gil, D.; Johnsson, M.; Garcia Chamizo, J.; Soriano, A.; Ruiz, D. Application of artificial neural networks in the diagnosis of urological dysfunctions. *Expert Syst. Appl.* **2009**, *36*, 5754–5760. [CrossRef]
24. Gil, D.; Johnsson, M. Using support vector machines in diagnoses of urological dysfunctions. *Expert Syst. Appl.* **2010**, *37*, 4713–4718. [CrossRef]
25. Leal, J.; Silva, F. Mooshak: A Web-based Multi-site Programming Contest System. *Softw. Pract. Exp.* **2003**, *33*, 567–581. [CrossRef]
26. Fernández-Alemán, J.L.; Carrillo de Gea, J.; Rodríguez Mondéjar, J. Effects of competitive computer-assisted learning versus conventional teaching methods on the acquisition and retention of knowledge in medical surgical nursing students. *Nurse Educ. Today* **2011**, *31*, 866–871. [CrossRef] [PubMed]
27. Fernández Alemán, J.L. Automated Assessment in a Programming Tools Course. *IEEE Trans. Educ.* **2011**, *54*, 576–581. [CrossRef]
28. Montoya-Dato, F.J.; Fernández Alemán, J.L.; García-Mateos, G. An Experience on Ada Programming Using On-Line Judging. In Proceedings of the 14th Ada-Europe International Conference on Reliable Software Technologies (Ada-Europe), Brest, France, 8–12 June 2009; pp. 75–89.
29. Peña-Ayala, A. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Syst. Appl.* **2014**, *41*, 1432–1462. [CrossRef]
30. Romero, C.; Ventura, S. Educational Data Mining: A Review of the State of the Art. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2010**, *40*, 601–618. [CrossRef]
31. Peña-Ayala, A.; Domínguez, R.; Medel, J. Educational data mining: A sample of review and study case. *World J. Educ. Technol.* **2009**, *1*, 118–139.
32. Hilty, L.M.; Hercheui, M.D. ICT and Sustainable Development. In *What Kind of Information Society? Governance, Virtuality, Surveillance, Sustainability, Resilience*; Berleur, J., Hercheui, M.D., Hilty, L.M., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 227–235, doi:10.1007/978-3-642-15479-9_22.
33. Alvarez-García, O.; Sureda-Negre, J.; Comas-Forgas, R. Assessing environmental competencies of primary education pre-service teachers in Spain: A comparative study between two universities. *Int. J. Sustain. High. Educ.* **2018**, *19*, 15–31. [CrossRef]
34. De Pauw, J.B.; Gericke, N.; Olsson, D.; Berglund, T. The effectiveness of education for sustainable development. *Sustainability* **2015**, *7*, 15693–15717. [CrossRef]
35. Álvarez-García, O.; Sureda-Negre, J.; Comas-Forgas, R. Environmental education in pre-service teacher training: A literature review of existing evidence. *J. Teach. Educ. Sustain.* **2015**, *17*, 72–85. [CrossRef]
36. Zalewski, J.; Sybramanian, N. Developing a Green Computer Science Program. In Proceedings of the Seventh Annual IEEE Green Technologies Conference (GreenTech), New Orleans, LA, USA, 15–17 April 2015; pp. 95–102.
37. Klimova, A.; Rondeau, E.; Andersson, K.; Porras, J.; Rybin, A.; Zaslavsky, A. An international Master's program in green ICT as a contribution to sustainable development. *J. Clean. Prod.* **2016**, *135*, 223–239. [CrossRef]
38. Anaya, A.R.; Boticario, J.G. Content-free Collaborative Learning Modeling Using Data Mining. *User Model. User-Adapt. Interact.* **2011**, *21*, 181–216. [CrossRef]
39. Muldner, K.; Burleson, W.; Van de Sande, B.; VanLehn, K. An analysis of students' gaming behaviors in an intelligent tutoring system: Predictors and impacts. *User Model. User-Adapt. Interact.* **2011**, *21*, 99–135. [CrossRef]

40. Hong, S.J.; Weiss, S.M. Advances in predictive models for data mining. *Pattern Recognit. Lett.* **2001**, *22*, 55–61. [CrossRef]
41. He, W. Examining students' online interaction in a live video streaming environment using data mining and text mining. *Comput. Hum. Behav.* **2013**, *29*, 90–102. [CrossRef]
42. Hsu, J.L.; Chou, H.W.; Chang, H.H. EduMiner: Using text mining for automatic formative assessment. *Expert Syst. Appl.* **2011**, *38*, 3431–3439. [CrossRef]
43. Peng, Y.; Kou, G.; Shi, Y.; Chen, Z. A Descriptive Framework for the Field of Data Mining and Knowledge Discovery. *Int. J. Inf. Technol. Decis. Mak.* **2008**, *7*, 639–682. [CrossRef]
44. Hsieh, T.C.; Wang, T.I. A mining-based approach on discovering courses pattern for constructing suitable learning path. *Expert Syst. Appl.* **2010**, *37*, 4156–4167. [CrossRef]
45. Nandeshwar, A.; Menzies, T.; Nelson, A. Learning patterns of university student retention. *Expert Syst. Appl.* **2011**, *38*, 14984–14996. [CrossRef]
46. Jin, W.; Lehmann, L.; Johnson, M.; Eagle, M.; Mostafavi, B.; Barnes, T.; Stamper, J. Towards Automatic Hint Generation for a Data-Driven Novice Programming Tutor. In Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), San Diego, CA, USA, 21–24 August 2011; pp. 91–96.
47. Hou, H.T. A case study of online instructional collaborative discussion activities for problem-solving using situated scenarios: An examination of content and behavior cluster analysis. *Comput. Educ.* **2011**, *56*, 712–719. [CrossRef]
48. Sen, B.; Uçar, E.; Delen, D. Predicting and analyzing secondary education placement-test scores: A data mining approach. *Expert Syst. Appl.* **2012**, *39*, 9468–9476. [CrossRef]
49. Sen, B.; Ucar, E. Evaluating the achievements of computer engineering department of distance education students with data mining methods. *Procedia Technol.* **2012**, *1*, 262–267. [CrossRef]
50. Mohamad, S.K.; Tasir, Z. Educational Data Mining: A Review. *Procedia Soc. Behav. Sci.* **2013**, *97*, 320–324. [CrossRef]
51. Anjewierden, A.; Kolloffel, B.; Hulshof, C. Towards educational data mining: Using data mining methods for automated chat analysis to understand and support inquiry learning processes. In Proceedings of the International Workshop on Applying Data Mining in e-Learning (ADML), Crete, Greece, 18 September 2007; pp. 27–36.
52. Kay, J.; Maisonneuve, N.; Yacef, K.; Zaïane, O. Mining Patterns of Events in Students' Teamwork Data. In Proceedings of the Workshop on Educational Data Mining at the 8th International Conference on Intelligent Tutoring Systems (ITS), Jhongli, Taiwan, 26–30 June 2006; pp. 45–52.
53. Perera, D.; Kay, J.; Koprinska, I.; Yacef, K.; Zaiane, O.R. Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 759–772. [CrossRef]
54. Meulen, M.; Revilla, M.A. The Effectiveness of Software Diversity in a Large Population of Programs. *IEEE Trans. Softw. Eng.* **2008**, *34*, 753–764. [CrossRef]
55. Witten, I.H.; Frank, E.; Hall, M.A.; Pal, C.J. *Data Mining: Practical Machine Learning Tools and Techniques*; Morgan Kaufmann: San Francisco, CA, USA, 2016.
56. Fernández-Alemán, J.L.; Carrillo-de Gea, J.M.; Meca, J.V.; Ros, J.N.; Toval, A.; Idri, A. Effects of Using Requirements Catalogs on Effectiveness and Productivity of Requirements Specification in a Software Project Management Course. *IEEE Trans. Educ.* **2016**, *59*, 105–118. [CrossRef]
57. Ngai, E.W.T.; Xiu, L.; Chau, D.C.K. Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Syst. Appl.* **2009**, *36*, 2592–2602. [CrossRef]
58. Chen, M.Y. Predicting corporate financial distress based on integration of decision tree classification and logistic regression. *Expert Syst. Appl.* **2011**, *38*, 11261–11272. [CrossRef]
59. Kurt, I.; Ture, M.; Kurum, A. Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Syst. Appl.* **2008**, *34*, 366–374. [CrossRef]
60. Ngai, E.W.T.; Hu, Y.; Wong, Y.; Chen, Y.; Sun, X. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decis. Support Syst.* **2011**, *50*, 559–569. [CrossRef]
61. Le Cessie, S.; van Houwelingen, J. Ridge estimators in logistic regression. *Appl. Stat.* **1992**, *41*, 191–201. [CrossRef]

62. Polat, K.; Gunes, S. A novel hybrid intelligent method based on C4.5 decision tree classifier and one-against-all approach for multi-class classification problems. *Expert Syst. Appl.* **2009**, *36*, 1587–1592. [CrossRef]
63. Lichman, M. UCI Machine Learning Repository. Available online: https://archive.ics.uci.edu/ml/index.php (accessed on 22 August 2018).
64. Brida, J.; Risso, W. Hierarchical structure of the German stock market. *Expert Syst. Appl.* **2010**, *37*, 3846–3852. [CrossRef]
65. Breiman, L.; Friedman, F.H.; Olshen, R.A.; Stone, C.J. *Classification and Regression Trees*; Brooks/Cole Publishing: Monterey, CA, USA, 1984,
66. Quinlan, J. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
67. Quinlan, J. *C4.5: Programs for Machine Learning*; Morgan Kaufmann: San Francisco, CA, USA, 1993.
68. Ture, M.; Tokatli, F.; Kurt, I. Using Kaplan-Meier analysis together with decision tree methods (C&RT, CHAID, QUEST, C4.5 and ID3) in determining recurrence-free survival of breast cancer patients. *Expert Syst. Appl.* **2009**, *36*, 2017–2026.
69. Yang, C.C.; Prasher, S.; Enright, P.; Madramootoo, C.; Burgess, M.; Goel, P.; Callum, I. Application of decision tree technology for image classification using remote sensing data. *Agric. Syst.* **2003**, *76*, 1101–1117. [CrossRef]
70. Aitkenhead, M. A co-evolving decision tree classification method. *Expert Syst. Appl.* **2008**, *34*, 18–25. [CrossRef]
71. Bishop, C.M. *Neural Networks for Pattern Recognition*; Oxford University Press, Inc.: Oxford, UK, 1995.
72. Ripley, B. *Pattern Recognition and Neural Networks*; Cambridge University Press: Cambridge, UK, 1996.
73. Haykin, S. *Neural Networks: A Comprehensive Foundation, Englewoods Cliffs*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1999.
74. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
75. Vellido, A.; Lisboa, P.J.; Vaughan, J. Neural networks in business: A survey of applications (1992–1998). *Expert Syst. Appl.* **1999**, *17*, 51–70. [CrossRef]
76. Rynkiewicz, J. General bound of overfitting for MLP regression models. *Neurocomputing* **2012**, *90*, 106–110. [CrossRef]
77. Guyon, I.; Boser, B.; Vapnik, V. A training algorithm for optimal margin classifiers. In Proceedings of the 5th Annual Workshop of Computational Learning Theory (COLT), Pittsburgh, PA, USA, 27–29 July 1992; pp. 144–152.
78. Burges, C. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167. [CrossRef]
79. Theodoridis, S.; Koutroumbas, K. Pattern Recognition. *IEEE Trans. Neural Netw. Learn. Syst.* **2008**, *19*, 376.
80. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
81. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: New York, NY, USA, 1995.
82. Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*; ACM: New York, NY, USA, 2013.
83. Mayvan, B.B.; Rasoolzadegan, A.; Yazdi, Z.G. The state of the art on design patterns: A systematic mapping of the literature. *J. Syst. Softw.* **2017**, *125*, 93–118. [CrossRef]
84. Fojtik, R. Design Patterns in the Teaching of Programming. *Procedia Soc. Behav. Sci.* **2014**, *143*, 352–357. [CrossRef]
85. Hilty, L.M.; Aebischer, B. ICT for Sustainability: An Emerging Research Field. In *ICT Innovations for Sustainability*; Springer: Cham, Switzerland, 2015; pp. 3–36.
86. Danilak, R. Why Energy Is a Big And Rapidly Growing Problem for Data Centers. 2017. Available Online: https://www.forbes.com/sites/forbestechcouncil/2017/12/15/why-energy-is-a-big-and-rapidly-growing-problem-for-data-centers/ (accessed on 22 August 2018).
87. Schomaker, G.; Janacek, S.; Schlitt, D. The Energy Demand of Data Centers. In *ICT Innovations for Sustainability*; Springer: Cham, Switzerland, 2015; pp. 113–124.
88. Fernandez, H.; Procaccianti, G.; Lago, P. Economic aspects of green ICT. In *Green in Software Engineering*; Springer: Cham, Switzerland, 2015; pp. 107–127.

89. Dayarathna, M.; Wen, Y.; Fan, R. Data Center Energy Consumption Modeling: A Survey. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 732–794, doi:10.1109/COMST.2015.2481183. [CrossRef]
90. Sahin, C.; Cayci, F.; Gutiérrez, I.L.M.; Clause, J.; Kiamilev, F.; Pollock, L.; Winbladh, K. Initial explorations on design pattern energy usage. In Proceedings of the First International Workshop on Green and Sustainable Software (GREENS), Zurich, Switzerland, 3 June 2012; pp. 55–61.