Applied Intelligence (2020) 50:4506–4528
https://doi.org/10.1007/s10489-020-01776-3

Multi-split optimized bagging ensemble model selection for multi-class educational data mining

MohammadNoor Injadat[1] · Abdallah Moubayed[1] · Ali Bou Nassif[2,1] · Abdallah Shami[1]

Published online: 22 July 2020

Abstract
Predicting students' academic performance has been a research area of interest in recent years, with many institutions focusing on improving the students' performance and the education quality. The analysis and prediction of students' performance can be achieved using various data mining techniques. Moreover, such techniques allow instructors to determine possible factors that may affect the students' final marks. To that end, this work analyzes two different undergraduate datasets at two different universities. Furthermore, this work aims to predict the students' performance at two stages of course delivery (20% and 50% respectively). This analysis allows for properly choosing the appropriate machine learning algorithms to use as well as optimize the algorithms' parameters. Furthermore, this work adopts a systematic multi-split approach based on Gini index and p-value. This is done by optimizing a suitable bagging ensemble learner that is built from any combination of six potential base machine learning algorithms. It is shown through experimental results that the posited bagging ensemble models achieve high accuracy for the target group for both datasets.

Keywords e-Learning · Student Performance Prediction · Optimized Bagging Ensemble Learning Model Selection · Gini Index

1 Introduction

Data mining is rapidly becoming a part of software engineering projects, and standard methods are constantly revisited to integrate the software engineering point of view. Data mining can be defined as an extraction of data from a dataset and discovering useful information from it [28, 34]. This is followed by the analysis of the collected data in order to enhance the decision-making process [17]. Data mining uses different algorithms and tries to uncover certain patterns from data [1]. These techniques techniques have proved to be effective solutions in a variety of fields including education, network security, and business [29, 50, 66]. Hence, they have the potential to also be effective in other fields such as medicine and education. Educational Data Mining (EDM), a sub-field of data mining, has emerged that specializes in educational data with the goal of better understanding students' behavior and improving their performance [12, 22]. Moreover, this sub-field also aims at enhancing the learning and teaching processes [17]. EDM often takes into consideration various types of data such as administrative data, students' performance data, and student activity data to gain insights and provide the appropriate recommendation [35, 48].

The rapid growth of technology and the Internet has introduced an interactive opportunities to help education field to improve the teaching and learning processes. In turn, this has led to the emergence of the field of e-learning. This field can be defined as "the use of computer network technology, primarily over an intranet or through the Internet, to deliver information and instruction to individuals" [33, 61]. There are various challenges facing

✉ MohammadNoor Injadat
minjadat@uwo.ca

Abdallah Moubayed
amoubaye@uwo.ca

Ali Bou Nassif
anassif@sharjah.ac.ae

Abdallah Shami
abdallah.shami@uwo.ca

[1] Electrical & Computer Engineering Department, University of Western Ontario, London, ON, Canada

[2] Computer Engineering Department, University of Sharjah, Sharjah, UAE

e-learning platforms and environment [49]. This includes the assorted styles of learning, and challenges arising from cultural differences [16]. Other challenges also exist such as pedagogical e-learning, technological and technical training, and e-learning time management [38]. To this end, personalized learning has emerged as a necessity in order to better cater to the learners' needs [30]. Accordingly, this personalization process has become a challenging task [13], as it requires adapting courses to meet different individuals' needs. This calls for adaptive techniques to be implemented [8, 14]. This can be done by automatically collecting data from the e-learning environment [8] and analyzing the learner's profile to customize the course according to the participant's needs and constraints such as his/her location, language, currency, seasons, etc. [8, 44, 46].

Many of the previous works in the literature focused on predicting the performance of the students by adopting a binary classification model. However, some educators prefer to identify not only two classes of students (i.e. Good vs. Weak), but instead they divide the students into several groups and consider the associated multi-class classification problem [58]. This is usually done because the binary model often identifies a large number weak students, many of which are not truly at risk of failing the course. Accordingly, this work considers two datasets at two different stages of the course, namely at 20% and 50% of the coursework, and divides the students into three groups, namely Weak, Fair, and Good students. Accordingly, the datasets are analyzed as a set of multi-class classification problems.

Multi-class classification problems can be solved by naturally extending the binary classification techniques for some algorithms, [3]. In this work, we consider various classification algorithms, compare their performances, and use Machine Learning (ML) techniques aiming to predict the students' performance in the most accurate way. Indeed, we consider K-nearest neighbor (k-NN), random forest (RF), Support Vector Machine (SVM), Multinomial Logistic Regression (LR), Naïve Bayes (NB) and Neural Networks (NN) and use an optimized systematic ensemble model selection approach coupled with ML hyper-parameter tuning using grid search optimization.

In this paper, we produced a bagging of each type of model and the bagging was used for the ensembles as opposed to single models. Bagging is itself an ensemble algorithm as it consists of grouping several models of the same type and defining a linear combination of the individual predictions as the final prediction on an external test sample, as explained in Section 6. Bagging is one of the best procedures to improve the performance of classifiers as it helps reduce the variance in many hard decision problems [10, 52]. The empirical fact that bagging improves the classifiers' performance is widely documented [9], and in fact ensemble methods placed first in many prestigious ML competitions, such as the Netflix Competition [54], KDD 2009 [24], and Kaggle [32]. Furthermore, a multi-split framework is considered for the studied datasets in order to reduce the bias of the ML models investigated as part of the bagging ensemble models.

The main disadvantage of bagging, and other ensemble algorithms, is the lack of interpretation. For instance, a linear combination of decision trees is much harder to interpret than a single tree. In the same way, bagging several variable selections gives little clues about which of the predictor variables are actually important.In this paper, in order to have a rough idea of which variables are the best predictors for each algorithm, we decided to average, for each variable, its importance in every model and this average is assigned to the variable and defined to be its *averaged importance*. This was done in order to better highlight the features that are truly important across the multiple splits under consideration.

The remainder of this paper is organized as follows: Section 2 presents some of the previous related work and their limitations; Section 3 summarizes the research contributions of this work; Section 4 describes the datasets under consideration and defines the corresponding target variables for both datasets; Section 5 describes the performance measurement approach adopted; Section 6 presents the methodology used to choose the best classifiers for the multi-class classification problem; Section 7 discusses the architecture used for training NN and shows the features' importance for each classifier for each dataset; Section 8 presents and discusses the experimental results both in terms of Gini Indices (also called Gini coefficient) and by using confusion matrices; and finally, Section 9 lists the research limitations, proposes multiple future research opportunities, and concludes the paper.

2 Related work and limitations

2.1 Related work

Educational data mining has become a rich field of research with the demand for empirical studies and research by academia increasing in recent years. This is due to the competitive advantages that can be gained from such kind of research. Data mining can be used to evaluate and analyze the different factors that improve the knowledge gaining, skills improvement of the learners, and makes the educational institution offer a better learning experience with highly qualified students or trainees [60].

Several researchers have explored the use of data mining techniques in an educational setting. Authors of [37] used data mining techniques to analyze the learner's web usage and content-based profiles to have an on-line automatic recommendation system. In contrast, Chang et al. proposed a k-NN classification model to classify the learner's style [11]. The results of this model was used to help the educational institution management and faculties to improve the courses' contents to satisfy the learner's needs [11].

Another related study that used simple leaner regression to check the effect of the student mother's education level and the family's income in learner's academic level was presented in [26].

On the other hand, Baradwaj and Pal used classification methods to evaluate the students' performance using decision trees [6]. The study was conducted using collected data from previous year's database to predict the student result at the end of the current semester. Their study aimed to provide a prediction that will help the next term instructors identify students that they may need help.

Other researchers [7] applied Naïve Bayes classification algorithm to predict students' grades based on their previous performance and other important factors. The authors discovered that, other than students' efforts, factors such as residency, the qualification standards of the mother, hobbies and activities, the total income of the family, and the state of the family had a significant effect on the students' performance.

Later, the same authors used Iterative Dichotomiser 3 (ID3) decision tree algorithm and if-then rules to accurately predict the performance of the students at the end of the semester [56] based on different variables like Previous Semester Marks, Class Test Grades, Seminar Performance, Assignments, Attendance, Lab Work, General Proficiency, and End Semester Marks.

Similarly, Moubayed et al. [51, 53] studied the student engagement level using K-means algorithm and derived a set of rulers that related student engagement with academic performance using Apriori association rules algorithm. The results analysis showed a positive correlation between students' engagement level and their academic performance in an e-learning environment.

Prasad et al. [57] used J48 (C4.5) algorithm and concluded that this algorithm is the best choice for making the best decision about the students' performance. The algorithm was also preferred because of its accuracy and speed.

Ahmed and Elaraby conducted a similar research in 2014 [2] using classification rules. They analyzed data from a course program across 6 years and were able to predict students' final grades. In similar fashion, Khan et al. [36] used J48 (C4.5) algorithm for predicting the final grade of Secondary School Students based on their previous marks.

Kostiantis et al. [40] proposed an incremental majority voting-based ensemble classifier based on 3 base classifiers, namely NB, k-NN, and Winnow algorithms. The authors' experimental results showed that the proposed ensemble model outperformed the single base models in a binary classification environment.

Saxena [62] used k-means clustering and J48 (C4.5) algorithms and compared their performance in predicting students' grades. The author concluded that J48 (C4.5) algorithm is more efficient, since it gave higher accuracy values than k-means algorithm. Authors in [59] used and compared K-Means and Hierarchical clustering algorithms. They concluded that K-means algorithm is more preferred to hierarchical clustering due to better performance and faster model building time.

Wang et al. proposed an e-Learning recommendation framework using deep learning neural networks model [65]. Their experiments showed that the proposed framework offered a better personalized e-learning experience. Similarly, Fok et al. proposed a deep learning model using TensorFlow to predict the performance of students using both academic and non-academic subjects [21]. Experimental results showed that the proposed model had a high accuracy in terms of student performance prediction.

Asogbon et al. proposed a multi-class SVM model to correctly predict students' performance in order to admit them into appropriate faculty program [4]. The performance of the model was examined using an educational dataset collected at the University of Lagos, Nigeria. Experimental results showed that the proposed model adequately predicted the performances of students across all categories [4].

In a similar fashion, Athani et al. also proposed the use of a multi-class SVM model to predict the performance of high school students and classify them into one of five letter grades A-F [5]. The goal was to predict student performance to provide a better illustration of the education level of the schools based on their students' failure rate. The authors used a Portuguese high school dataset consisting mostly of the students' socio-economic descriptors as features. Their experiments showed that the proposed multi-class SVM model achieved high prediction accuracy close to 89% [5].

Jain and Solanki proposed a comparative study between four tree-based models to predict the performance of students based on a three-class output [31]. Similar to the work of Athani et al., the authors in this work also considered the Portuguese high school dataset consisting mostly of the students' socio-economic descriptors as features. Experimental results showed that the proposed tree-based model also achieved high prediction accuracy with a low execution time [31].

2.2 Limitations of related work

The limitations of the related work can be summarized as follows:

- Do not analyze the features before applying any ML model. Any classification model is directly applied without studying the nature of the data being considered.

- Mostly consider the binary classification case. Such cases often lead to identifying too many students which are not truly in danger of failing the course and hence would not need as much help and attention. Even when multi-class models were considered, the features used were mostly focused on students' socio-economic status rather than their performance in different educational tasks.
- Often use a single classification model or an ensemble model built upon randomly chosen group of base classifiers. Moreover, to the best of our knowledge, only majority voting-based ensemble models are considered.
- Often predict the performance of students from one course to the other or from one year to the other. Performance prediction is rarely considered during the course delivery.
- Often use the default parameters of the utilized algorithms/techniques without optimization.

3 Research contribution

To overcome the limitations presented in Section 2.2, our research aims to predict the students' performance during the course delivery as opposed to other previous works that perform the prediction at the end of the course. The multi-class classification problem assumes that their is a proportional relationship between the students' efforts and seriousness in the course and their final course performance and grade.

More specifically, our work aims to:

- *Analyze* the collected datasets and visualize the corresponding features by applying different graphical and quantitative techniques (e.g. dataset distribution visualization, target variable distribution, and feature importance).
- *Optimize* hyper-parameters of the different ML algorithms under consideration using *grid search* algorithm.
- *Propose* a systemic approach to build a multi-split-based (to reduce bias) bagging ensemble (to reduce variance) learner to select the most suitable model depending on multiple performance metrics, namely the Gini index (for better statistical significance and robustness) and the target class score.
- *Study* the performance of the proposed ensemble learning classification model on multi-class datasets.
- *Evaluate* the performance of the proposed bagging ensemble learner in comparison with classical classification techniques.

Note that in this work, the term *Gini index* refers to the Gini coefficient that is calculated based on the Lorenz curve and area under the curve terms [43]. Therefore, the remainder of this work adopts to the term *Gini index*.

4 Dataset and target variable description

4.1 Dataset description

In this section,the two datasets under consideration are described at the two course delivery stages (20% and 50% of the coursework). This corresponds to the results of a series of tasks performed by University students. Moreover, Principal Components Analysis (PCA) is conducted to better visualize the considered datasets.

- *Dataset 1*: The experiment was conducted at the University of Genoa on a group of 115 first year engineering major students [63]. The dataset consists of data collected using a simulation environment named Deeds (Digital Electronics Education and Design Suite). This e-Learning platform allows students to access the courses' contents using a special browser and asks the students to solve problems that are distributed over different complexity levels.

  Table 1 shows a summary of the different tasks for which the data was collected. It is worth mentioning that 52 students out of the original 115 students registered were able to complete the course.

  The 20% stage consists of the grades of tasks ES 1.1 to ES 3.5. On the other hand, the 50% stage consists of tasks ES. 1.1 to ES 5.1.

  To improve the accuracy of the classification model, empty marks were replaced with a 0. Moreover, all tasks' marks were converted to a scale out of 100. Furthermore, all decimal point marks were rounded to the nearest 1 to maintain consistency.
- *Dataset 2*: This dataset was collected at the University of Western Ontario for a second year undergraduate Science course. The dataset is composed of two main parts. The first part is an event log of the 486 students enrolled. This event log dataset consists of 305933 records. In contrast, the other part, which is under consideration in this research, is the grades of the 486 students in the different evaluated tasks. This includes assignments, quizzes, and exams.

  Table 2 summarizes the different tasks evaluated within this course. The 20% stage consists of the results of Assignment 01 and Quiz 01. On the other hand, the 50% stage consists of the grades of Quiz 01, Assignments 01 and 02, and the midterm exam.

  Similar to Dataset 1, all empty marks were replaced with a value of 0 for better classification accuracy.

Table 1 Dataset 1 - Features

Feature	Description	Type	Value/s
Id	Student Id.	Nominal	Std. 1,..,Std. 52
ES 1.1	Exc. 1.1 Mark	Numeric	0..2
ES 1.2	Exc. 1.2 Mark	Numeric	0..3
ES 2.1	Exc. 2.1 Mark	Numeric	0..2
ES 2.2	Exc. 2.2 Mark	Numeric	0..3
ES 3.1	Exc. 3.1 Mark	Numeric	0..1
ES 3.2	Exc. 3.2 Mark	Numeric	0..2
ES 3.3	Exc. 3.3 Mark	Numeric	0..2
ES 3.4	Exc. 3.4 Mark	Numeric	0..2
ES 3.5	Exc. 3.5 Mark	Numeric	0..3
ES 4.1	Exc. 4.1 Mark	Numeric	0..15
ES 4.2	Exc. 4.2 Mark	Numeric	0..10
ES 5.1	Exc. 5.1 Mark	Numeric	0..2
ES 5.2	Exc. 5.2 Mark	Numeric	0..10
ES 5.3	Exc. 5.3 Mark	Numeric	0..3
ES 6.1	Exc. 6.1 Mark	Numeric	0..25
ES 6.2	Exc. 6.2 Mark	Numeric	0..15
Final Grade	Total Final Mark	Numeric	0..100
Total	Final Course Grade	Nominal	G,F,W

Moreover, all marks were scaled out of 100. Additionally, decimal point marks were rounded to the nearest 1.

4.2 Target variable description

For the two datasets under consideration, the target variables were constructed by considering the final grade. More specifically, the students were grouped into three groups as follows:

1. Good (G) – the student will finish the course with a good grade ($70 - 100\%$);
2. Fair (F) – the student will finish the course with a fair grade ($51 - 69\%$);
3. Weak (W) – the student will finish the course with a weak grade ($\leq 50\%$).

In this case, the target group is the Weak students (W) who are predicted to receive a mark below 50%, meaning that they are at risk of failing the course. Figure 1 shows that Datasets 1 and 2 are characterized by being small sized and unbalanced respectively. These two issues have more of an impact on the classification problem. It can be seen that for the first dataset, the three classes are relatively evenly distributed, but each class consists of only a few students. On the other hand, the second dataset is not small sized but is strongly unbalanced, having only 8 Weak students out of 486 students.

To better visualize the three classes, we applied PCA to the datasets (both considered at Stage 50%) as shown in Figs. 2 and 3. Looking at these two figures, we note that it can be possible to draw a boundary that separates Weak Students from the rest of the students, whereas Fair and Good students are too close and not separable by a boundary. We will see in the next sections that the performance of the models is affected by this distribution and that most of the algorithms fail in distinguishing between Fair and Good students, especially for Dataset 1.

5 Performance evaluation metrics description

In general there are two standard approaches to choosing multiple class performance measures [3, 25]. One approach, namely *OVA (One-versus-all)*, is to reduce the problem of classifying among N classes into N binary problems. In this case, every class is discriminated from the other classes. In the second approach, called *AVA (All-versus-all)*, each class is compared to each other class. In other words, it is necessary to build a classifier for every pair of classes, i.e. building $\frac{N(N-1)}{2}$ classifiers, while discarding the rest of the classes.

Due to the size of our datasets, we chose to follow the first method as opposed to the second one. In fact, if we were to use the second approach for Dataset 1, we would need to train three binary models, one for each pair of classes (G,F), (F,W), and (G,W). In particular, the subset of

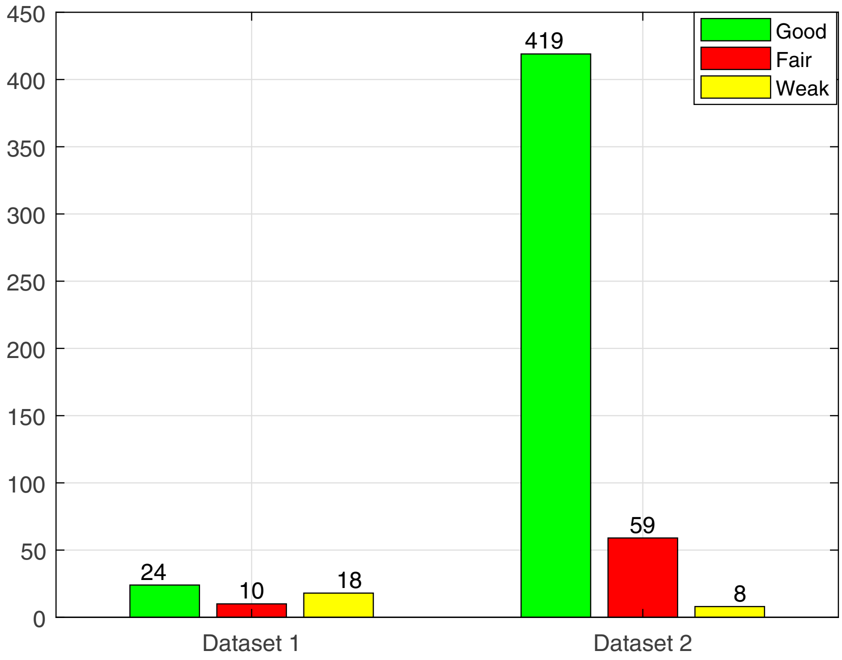


Fig. 1 Dataset 1 and Dataset 2-Target Variables

Table 2 Dataset 2 - Features

Feature	Description	Type	Value/s
Id	Student Id.	Nominal	std000,..,std485
Quiz01	Quiz1 Mark	Numeric	0..10
Assign.01	Assign.01 Mark	Numeric	0..8
Midterm	Midterm Mark	Numeric	0..20
Assign.02	Assign.02 Mark	Numeric	0..12
Assign.03	Assign.03 Mark	Numeric	0..25
Final Exam	Final Exam Mark	Numeric	0..35
Final Grade	Total Final Mark	Numeric	0..100
Total	Final Grade	Nominal	G,F,W

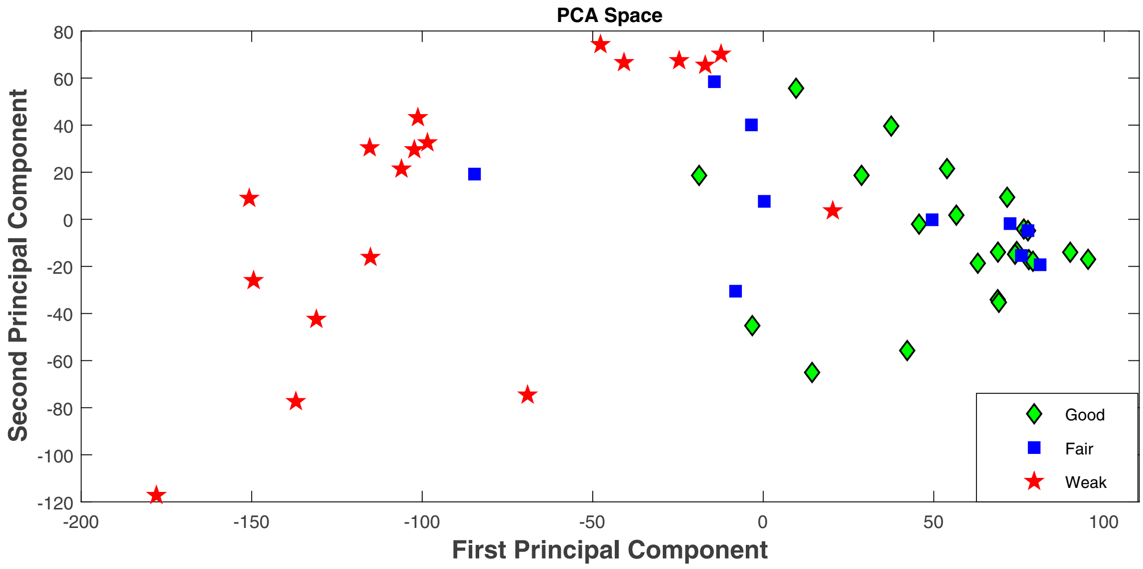


Fig. 2 Dataset 1 - multi-class target visualization

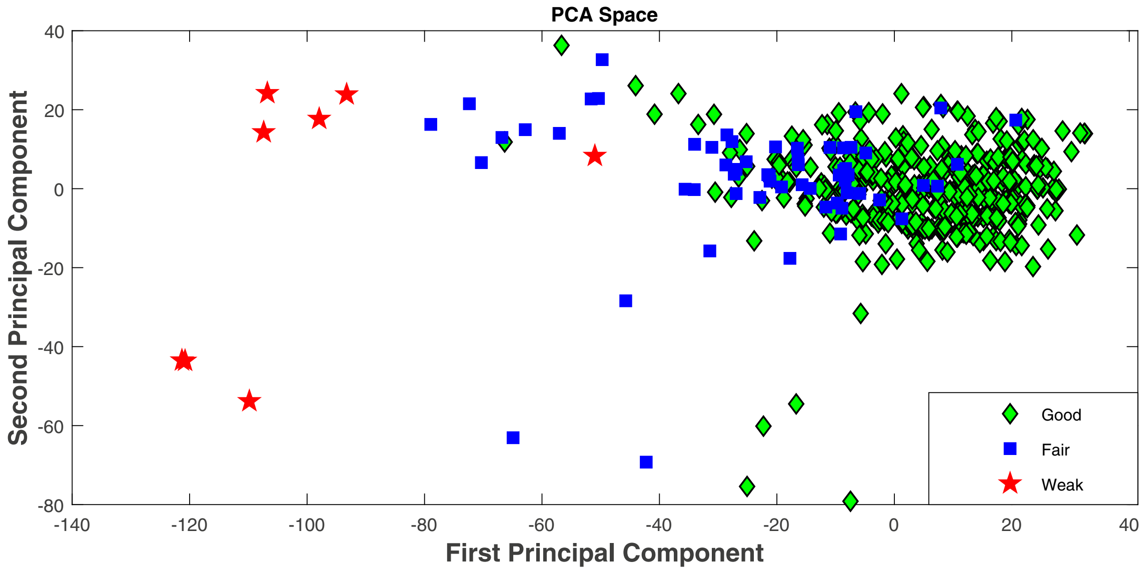


Fig. 3 Dataset 2 - multi-class target visualization

data for the (F,W) model would consist of only 28 students, which would be split into Training Sample (70%) and Test Sample (30%). This corresponds to training a model using 20 students and testing it using only 8 students. Due to the relatively small size of the (F,W) model, we determine that the AVA approach would not be suitabe for accurate prediction.

It is well-known that the Gini Index metric, as well as the other metrics (Accuracy, ROC curve etc.) can be generalized to the multi-class classification problem. In particular, *we choose the Gini Index metric instead of the Accuracy because the latter depends on the choice of a threshold whereas the Gini Index metric does not*. This makes it statistically more significant and robust than the accuracy, particularly given that it provides a measure of the statistical dispersion of the classes [27].

In particular, we implemented a generalization of Gini index metric: during the training phase, that computes the Gini Index of each one of the three binary classifications and *optimizes (i.e. maximizes) the average of the 3 performances*, i.e. the performances corresponding to classes G, F, W.

6 Methodology

For the multi-class classification problem we used several algorithms. More specifically we explored RF, SVM - RBF, k-NN, NB, LR, and NN with 1, 2 and 3 layers (i.e. 3 different NN models), for a total of eight classifiers per dataset.

In order to achieve better performances, we did not build only one individual model for each algorithm, instead we constructed baggings of classifiers. In fact, as explained in the previous section, bagging reduces the variance.

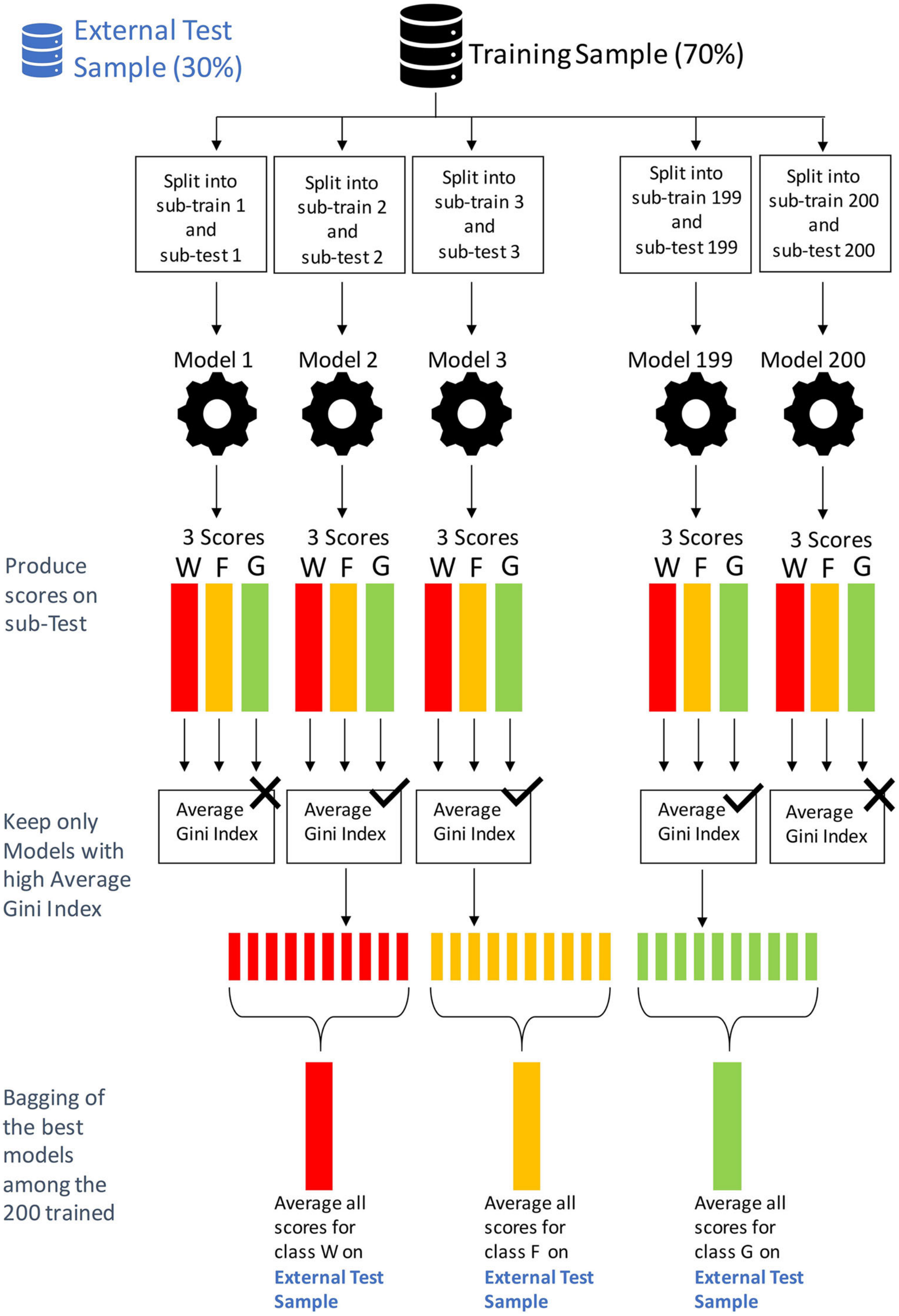
We built a bagging of models for each algorithm in the following way: we started by splitting each dataset into Training and Test samples in proportions 70%-30% then we used the training sample to build baggings of models. More precisely the Training sample was split 200 times into sub-Training and sub-Test samples randomly but forcing the percentages of Fair, Good and Weak students to be the same as the ones in the entire datasets.

The models resulting from the 200 splits were trained on the sub-Training samples and inferred on the corresponding sub-Test samples. If the Average Gini Index was above a certain fixed threshold (lowest acceptable Gini Index) then the model was kept otherwise it was discarded. For each algorithm we obtained in this way a set of models having the best performances, and we averaged their scores on the (external) Test sample, class by class. This procedure is explained in Fig. 4.

Once we had the eight baggings of models (one for each algorithm), we considered all the possible ensembles that could be constructed with them and compared their performances in terms of Gini Index, as explained in Section 5. Moreover, for each dataset, we computed the p-values corresponding to each one of the 256 possible ensembles and aimed to choose as the final ensemble the one that had best Gini Index and, at the same time, that was statistically significant.

The Gini Index, also commonly referred to as the Gini coefficient, can be seen geometrically as the area between the Lorenz curve [43] and the diagonal line representing perfect equality. The higher the Gini Index, the better the

Fig. 4 Bagging Ensemble Model Building Methodology


External Test Sample (30%)
Training Sample (70%)
Split into sub-train 1 and sub-test 1
Split into sub-train 2 and sub-test 2
Split into sub-train 3 and sub-test 3
Split into sub-train 199 and sub-test 199
Split into sub-train 200 and sub-test 200
Model 1
Model 2
Model 3
Model 199
Model 200
Produce scores on sub-Test
3 Scores
W F G
Keep only Models with high Average Gini Index
Average Gini Index
Bagging of the best models among the 200 trained
Average all scores for class W on External Test Sample
Average all scores for class F on External Test Sample
Average all scores for class G on External Test Sample

performance of the model. Formally the Gini index is defined as follows:

Let $F(z)$ be the cumulative distribution of z and let a and b be the highest and the lowest value of z respectively, then the we can calculate half of Gini's expected mean difference as:

$$2\int_a^b F(z)[1 - F(z)]dz \tag{1}$$

Alternatively, the Gini index can be calculated as $2 * \text{Area Under Curve} - 1$.

On the other hand, the statistical significance of our results is determined by computing the p-values. The general approach is to test the validity of a claim, called the *null hypothesis*, made about a population. An alternative hypothesis is the one you would believe if the null hypothesis is concluded to be untrue. A small p-value ($\leq$ 0.05) indicates strong evidence against the null hypothesis, so you reject the null hypothesis. For our purposes, the null hypothesis states that the Gini Indices were obtained by chance. We generated 1 million random scores from normal distribution and calculated the p-value. The ensemble learners selected have p-value ≤ 0.05, indicating that there is strong evidence against the null hypothesis. Therefore, choosing an ensemble model using a combination of Gini Index and p-value allows us to have a more statistically significant and robust model.

The classifiers were inferred on the test sample, giving as output three vectors of predictions to be analyzed. These three vectors express the chance that each student is classified as Weak, Fair and Good. In order t o build the confusion matrices, we fixed a threshold for each class, namely τ_F, τ_G, and τ_W. To determine each threshold, a one-vs-all method is considered for each class with the threshold being chosen as the score for which the point on the ROC curve is closest to the top-left corner (commonly referred to as the Youden Index) [20]. This is done in order to find the point that simultaneously maximizes the sensitivity and specificity.

For each student belonging to the Test sample, we defined the predicted class according to the following steps:

1. The 3 scores corresponding to the 3 classes were normalized in order to make them comparable.
2. For each class, if the probability is higher than the corresponding threshold then the target variable for the binary classification problem associated to that class is predicted to be 1, otherwise it's 0.
3. In this way we obtained a 3-column matrix taking values 1's and 0's. Comparing the 3 predictions, if a student has only one possible outcome (i.e. only one 1, and two 0's) then the student is predicted to belong to the corresponding class. Otherwise, if there is uncertainty about the prediction because there is more than one 1 predicted for the student, then the class with the highest score is chosen to be the predicted one.

For instance, consider the following example:

Example 1 Suppose we have trained a classifier using 70% of Dataset 1. When we infer the model on the test sample (remaining 30%, consisting of 15 students), we obtain 3 vectors of scores, one for each class and we can compute their Gini Indices, see Fig. 5.

In this example, the Gini Indices of Classes F, G, W are 97.2%, 76.8%, 98% respectively, hence the Averaged Gini Index is 90.7%.

We map the three scores linearly to the interval [0, 1], i.e. we normalize them to make them comparable. The normalized scores are represented in Table 3 in columns *score F*, *score G*, *score W*.

Column *Actual Class* corresponds to the actual target variable that we aim to predict. Treating each score as if it was the score associated to a binary classification problem, we need to set a threshold for each class such that if the score is greater than the threshold then the student belongs to such class otherwise he/she doesn't (i.e., he/she belongs to one of the other two classes). Therefore we set three thresholds τ_F, τ_G, and τ_W for Class, F, G, and W respectively. For instance, let $\tau_F = 0.267$, $\tau_G = 0.323$, and $\tau_W = 0.740$. For student 1 in Table 3, the chance to be classified as F is $0.365 \geq \tau_F$, whereas the probabilities to belong to classes G and W are less than τ_G and τ_W respectively. In conclusion, once the three thresholds are set, we can claim that student 1 is a Fair student.

Student 6 has score $F = 0.389 \geq \tau_F$ and score $G = 0.620 \geq \tau_G$ so he/she belongs either to Class F or to class G. Since the scores are normalized and are comparable, we set the predicted class to be the one corresponding to the highest score, hence we predict student ID=6 to belong to class G.

For student 2 (7 and 14) note that the three scores are all below the thresholds so the predicted class is the one corresponding to the greatest score, i.e. the student is predicted as Weak.

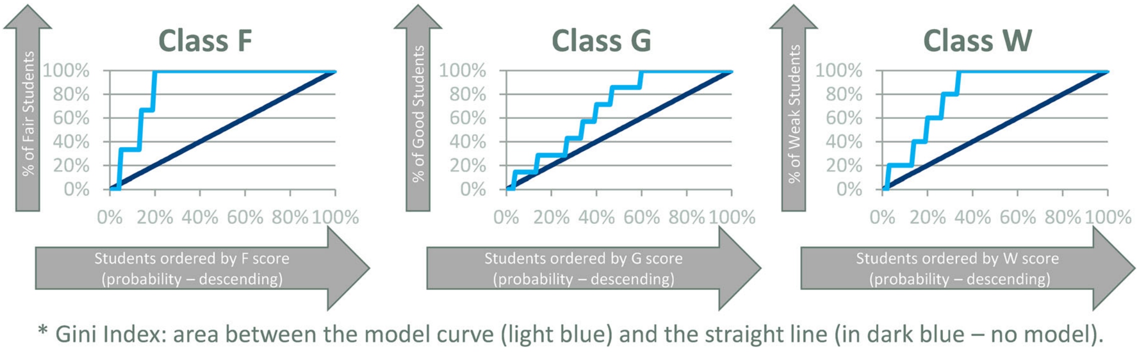


Fig. 5 Example - Averaged Gini Index Computation

Table 3 Example - Predicting Classes

ID	Actual Class	score F	score G	score W	Max Pred.	F	G	W	Predicted Class
1	F	0.365	0.1	0.707	W	1	0	0	F
2	F	0.015	0.25	0.647	W	0	0	0	W
3	G	0.828	0.232	0.337	F	1	0	0	F
4	G	0.085	0.13	0.758	W	0	0	1	W
5	W	0.663	0.038	0.853	W	1	0	1	W
6	G	0.389	0.62	0.142	G	1	1	0	G
7	G	0.234	0.078	0.723	W	0	0	0	W
8	W	0	0.054	0.793	W	0	0	1	W
9	W	0.009	0	0.944	W	0	0	1	W
10	G	0.33	0.797	0	G	1	1	0	G
11	F	0.266	0.818	0.01	G	0	1	0	G
12	G	0.33	0.797	0	G	1	1	0	G
13	G	0.248	0.58	0.22	G	0	1	0	G
14	W	0.18	0.167	0.648	W	0	0	0	W
15	W	0.061	0.186	0.745	W	0	0	1	W

The max probability associated to each student is expressed in column *Max Pred.*, and if we compare this column with column *Actual Class* we note that taking the max score as the predicted class would not have been a good strategy.

By setting the three thresholds τ_F, τ_G, and τ_W and considering the max score only in case of uncertainty we obtained for each student a predicted class, expressed in column *Predicted Class*. If we compare the actual class with the predicted class we can build the corresponding confusion matrix (Table 4).

The threshold for class W in dataset 1 is typically higher than that for the other two classes due to the combination of two reasons. The first is that the test sample is fairly small. The second is that the number of class W instances is also small. As such, based on the fact that the threshold is determined by finding the score that results in the closest point on the ROC curve to the top left corner, the threshold has to be high in order to make sure that the points are identified correctly. Therefore, since the number of class W points is low, missing one of them would result in a significant drop in specificity and sensitivity. Thus, the optimal threshold should be high to be able to identify and classify them correctly.

Table 4 Example - Confusion Matrix

	F	G	W
F	1	1	1
G	1	4	2
W	0	0	5

7 ML parameter tuning and application

We chose one algorithm for each area of ML aiming to cover all types of classification methods including tree-based (RF), vector-based (SVM-RBF), distance-based (k-NN), regression-based (LR), probabilistic (NB), and neural network-based (NN1, NN2, and NN3 with 5 neurons per layer). The corresponding bagging ensemble models consist of all possible combinations of the aforementioned base models. In Section 7.1, we explain how we train a NN. In the following sections, for each dataset, we show the impact of each variable on the performance of each classifier. As explained in Section 1, in order to understand which variables are the best predictors for each algorithm, we decided to average, for each variable, its importance on every model and this average is assigned to the variable and defined to be its *averaged importance*. In Section 8 we will show that the most important variables affect the performances of some classifiers.

7.1 Neural network tuning

Finding the optimal number of neurons for NN is still an open field of research and requires a lot of computational resources. The authors in [64] summarize some formulas for the computation of the optimal number of hidden neurons N_h:

- $N_h = \frac{\sqrt{1+8N_i}-1}{2}$
- $N_h = \sqrt{N_i N_o}$
- $N_h = \frac{4N_i^2+3}{N_i^2-8}$

where N_i is the number of input neurons (number of variables) and N_o is the number of output neurons (3

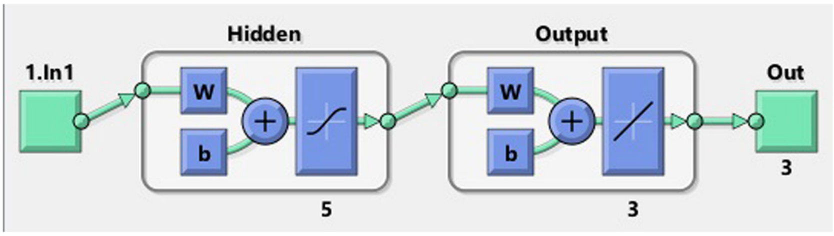


Fig. 6 NN with 1 hidden layer

classes). Applying the latter formulas to our datasets at the two different stages, we obtained a number of neurons between 2 and 6. Considering that we adopted the early stopping technique in order to prevent over-fitting and reduce variance, we decided to choose this number in the high range of the interval [2, 6] and set it to be equal to 5 instead of performing a full optimization (i.e., brute force searching).

The results obtained by using 1 hidden layer with 5 neurons were so promising that we decided to stress our hypothesis about early stopping and tried NN with 2 and 3 hidden layers with 5 neurons each, obtaining similar results.

The NN models we built are as in Figs. 6, 7 and 8.

The initialization of the weights of neural networks was implemented by using the Nguyen-Widrow Initialization Method [55] whose goal is to speed up the training process by choosing the initial weights instead of generating them randomly. Simply put, this method assigns to each hidden node its own interval at the start of the training phase. By doing so, during the training each hidden layer has to adjust its interval size and location less than if the initial weights are chosen randomly. Consequently, the computational cost is reduced.

Levenberg-Marquardt backpropagation was used to train the models: this algorithm was introduced for the first time by Levenberg and Marquardt in [47], and is derived from Newton's method that was designed for minimizing functions that are sums of squares of nonlinear functions [45]. This method is confirmed to be the best choice in various learning scenarios, both in terms of time spent and performance achieved, [15] . Moreover, the datasets were normalized in input by mapping linearly to $[-1, 1]$ (the activation function used in the input layer is the hyperbolic tangent) and in output to $[0, 1]$ (the activation function in the output layer is linear) in order to avoid saturation of neurons and make the training smoother and faster.

7.2 ML algorithms' parameter tuning

Hyper-parameter tuning has become an essential step to improve the performance of ML algorithms. This is due to the fact that each ML algorithm is governed by a set of parameters that dictate its predictive performance [39]. Several methods have been proposed in the literature to optimize and tune these parameters such as grid search algorithm, random search, evolutionary algorithms, and Bayesian optimization method [29, 39].

This work adopts the *grid search* method to perform hyper-parameter tuning. Grid search optimization method is a well-known optimization method often used to hyper tune the parameters of ML classification techniques. Simply put, it discretizes the values for the set of techniques' parameters [39]. For every possible combination of parameters, the corresponding classification models are trained and assessed. Mathematically speaking, this can be formulated as follows:

$$\max_{parm} f(parm) \tag{2}$$

where f is an objective function to be maximized (typically the accuracy of the model) and $parm$ is the set of parameters to be tuned. Despite the fact that this may seem computationally heavy, grid search method benefits from the ability to perform the optimization in parallel, which results in a lower computational complexity [39].

In contrast to traditional hyper-parameter tuning algorithms that perform the optimization with the objective of maximizing the accuracy of the ML model, this work

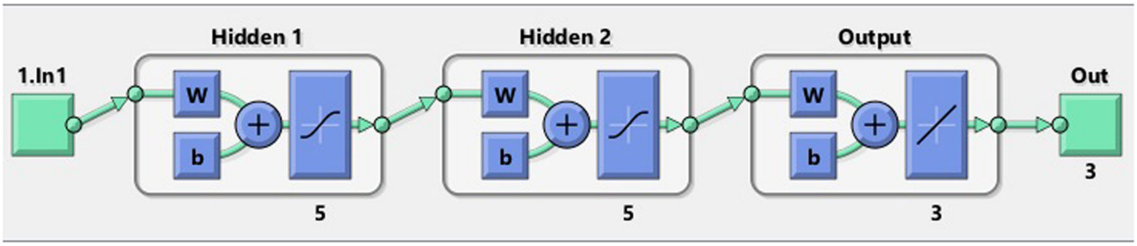


Fig. 7 NN with 2 hidden layers

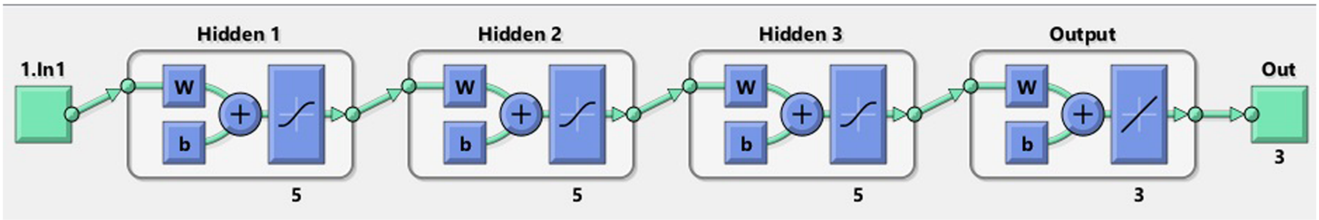


Fig. 8 NN with 3 hidden layers

tunes the parameters used for each model using the *grid search* optimization method to maximize the average Gini index (for more statistical significance and robustness [27]) over multiple splits [42]. More specifically, the objective function is:

$$\max_{parm} Average\ Gini\ Index = \max_{parm} \frac{1}{N} \sum_{i=1}^{N} Gini\ Index_i(parm) \tag{3}$$

where $parm$ is the set of parameters to be tuned for each ML algorithm and N is the number of different splits considered. For example, in the case of K-NN algorithm, $parm = \{K\}$ which is the number of neighbors used to determine the class of the data point.

R was used to implement the eight classifiers and the corresponding ensemble learners. As mentioned above, the eight classifiers considered in this work are SVM-RBF, LR, NB, k-NN, RF, NN1, NN2, and NN3. All the classifiers were trained using all the variables available. Moreover, the parameters of the algorithms were tuned by maximizing the Gini Index of each split. Furthermore, 200 different splits of data were used to reduce the bias of the models under consideration.

Table 5 summarizes the range of values for the parame-5ers of the different ML algorithms considered in this work.

Note the following:

- For the NB algorithm, density estimator used by the algorithm is represented using the *usekernel* parameter. In particular, *usekernel=false* means that the data distribution is assumed to be Gaussian. On the other hand, *usekernel* = *true* means that the data distribution is assumed to be non-Gaussian.
- The LR algorithm was not included in the table. This is due to the fact that it has no parameters to optimize. The sigmoid function, which is the default function, was used by the grid search method to maximize the Gini index.
- The NN method was not included in the table because it was explained in the previous Section 7.1.

The features are ordered according to their importance. This is done for the two datasets and for each of the algorithm used. This provides us with better insights about which features are important for each algorithm and each dataset. The importance of the features is determined using the CARET package available for R language [41]. Depending on the classification model adopted, the importance is calculated in one of multiple ways. For example, when using RF method, the prediction accuracy on the out-of-bag portion of the data is recorded. This is iteratively done after permuting each predictor variable. The difference between the two accuracy values is then averaged over all trees and normalized by the standard error [41]. In contrast, when the k-NN method is used, the difference between the class centroid and the overall centroid is used to measure the variable influence. Accordingly, the separation between the classes is larger whenever the difference between the class centroids is larger [41]. On the other hand, when using the NN method, the CARET package uses the same feature importance method proposed in Gevrey et al. which uses combinations of the absolute values of the weights

Table 5 Grid Search Parameter Tuning Range

Algorithm	Parameter Range in Dataset 1	Parameter Range in Dataset 2
SVM-RBF	C=[0.25, 0.5, 1] & sigma = [0.05-0.25]	C=[0.25, 0.5, 1] & sigma = [0.5-3.5]
NB	usekernel=[True,False]	usekernel=[True,False]
K-NN	k=[5,7,9,...,43]	k=[5,7,9,...,43]
RF	mtry=[2,3,...,12]	mtry=[2,3,4]

Table 6 Dataset 1 - Stage 20% - Features' importance for Different Base Classifiers

Ranking	RF	SVM-RBF	NN1	NN2	NN3	k-NN	LR	NB
1	ES2.2	ES2.2	ES2.2	ES2.2	ES2.2	ES2.2	ES1.1	ES2.2
2	ES3.3	ES3.3	ES3.5	ES3.3	ES3.3	ES3.3	ES1.2	ES3.3
3	ES2.1	ES2.1	ES3.3	ES3.5	ES3.5	ES2.1	ES3.5	ES2.1
4	ES1.1	ES3.5	ES3.2	ES2.1	ES2.1	ES3.5	ES3.3	ES3.5
5	ES3.5	ES1.2	ES1.1	ES3.2	ES3.2	ES3.4	ES3.4	ES3.4
6	ES1.2	ES3.4	ES2.1	ES1.1	ES1.1	ES1.2	ES3.2	ES1.2
7	ES3.4	ES1.1	ES1.2	ES3.4	ES3.4	ES1.1	ES3.1	ES1.1
8	ES3.1	ES3.1	ES3.4	ES1.2	ES1.2	ES3.1	ES2.2	ES3.2
9	ES3.2	ES3.2	ES3.1	ES3.1	ES3.1	ES3.2	ES2.1	ES3.1

[23]. This importance is reflected in the weights calculated for each feature for each classification model with more important features contributing more towards the prediction.

The final step consist of selecting the most suitable bagging ensemble learner for both datasets at the two course delivery stages.

7.3 Features importance: Dataset 1 - Stage 20%

- RF: The variables' importance in terms of predictivity is described in Table 6 that shows that the most relevant features are ES2.2 and ES3.3.
- SVM-RBF: The variables' importance for SVM is described in Table 6, that shows that the most relevant features are ES2.2 and ES3.3.
- NN1: For NN1, the variables' importance in terms of predicativity is described in Table 6 that shows that the most relevant features are ES2.2 and ES3.5.
- NN2: The most important variables for NN2 are ES2.2 and ES3.2, as shown in Table 6.
- NN3: The variables' importance in terms of predicativity is described in Table 6 that shows that the most relevant features are ES2.2 and ES3.2.
- k-NN: Table 6 shows that the most relevant features for k-NN are ES2.2 and ES3.3.
- LR: The variables' importance in terms of predicativity is described in Table 6 that shows that the most relevant features are ES1.1 and ES1.2.
- NB: Table 6 shows that the most relevant features are ES2.2 and ES3.3.

7.4 Features importance: Dataset 1 - Stage 50%

It is important to point out that, for Dataset 1 at stage 50%, features ES4.1 and ES4.2 are the most important for every classifier.

- RF: For RF, the variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.

Table 7 Dataset 1 - Stage 50% - Features' importance for Different Base Classifiers

Ranking	RF	SVM-RBF	NN1	NN2	NN3	k-NN	LR	NB
1	ES4.1	ES4.1	ES4.1	ES4.1	ES4.1	ES4.1	ES4.1	ES4.1
2	ES4.2	ES4.2	ES4.2	ES4.2	ES4.2	ES4.2	ES4.2	ES4.2
3	ES2.2	ES2.2	ES3.3	ES3.5	ES5.1	ES2.2	ES1.1	ES1.1
4	ES5.1	ES5.1	ES3.5	ES3.3	ES3.5	ES5.1	ES2.1	ES2.1
5	ES2.1	ES3.3	ES2.1	ES5.1	ES3.3	ES3.3	ES1.2	ES1.2
6	ES1.1	ES2.1	ES5.1	ES2.1	ES2.1	ES2.1	ES3.3	ES3.3
7	ES3.5	ES3.5	ES1.1	ES3.4	ES2.2	ES3.5	ES3.4	ES3.4
8	ES3.3	ES1.2	ES3.4	ES2.2	ES1.1	ES3.4	ES5.1	ES5.1
9	ES3.4	ES3.4	ES3.2	ES1.1	ES3.4	ES1.2	ES3.5	ES3.5
10	ES3.1	ES1.1	ES2.2	ES3.1	ES3.2	ES1.1	ES2.2	ES2.2
11	ES3.2	ES3.1	ES1.2	ES3.1	ES3.1	ES3.1	ES3.1	ES3.1
12	ES1.2	ES3.2	ES3.1	ES1.2	ES1.2	ES3.2	ES3.2	ES3.2

Table 8 Dataset 2 - Stage 20% - Features' importance

Ranking	Feature
1	Assignment01
2	Quiz01

- SVM-RBF: The variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.
- NN1: The variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.
- NN2: The variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.
- NN3: The variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.
- k-NN: Table 7 shows that the most relevant features for k-NN are ES4.1 and ES4.2.
- LR: Table 7 shows that the most relevant features for LR are ES4.1 and ES4.2.
- NB: The variables' importance in terms of predicativity is described in Table 7 that shows that the most relevant features are ES4.1 and ES4.2.

In general, the most important features for almost all the classifiers are ES4.1 and ES4.2. These features correspond to the *Evaluate* category as per Bloom's taxonomy which represents one of the highest level of comprehension of the course material from the educational point of view. Therefore, it makes sense for these features to be suitable indicators and predictors of student performance.

Table 9 Dataset 2 - Stage 50% - NN1, NN2, k-NN, and NB, Features' importance

Ranking	Feature
1	Assignment02
2	Assignment01
3	Midterm Exam
4	Quiz01

7.5 Features importance: Dataset 2 - Stage 20%

We have only two features for Dataset 2, Stage 20% and for all the classifiers, the list of features ordered by importance, see Table 8.

Since Dataset 2 at stage 20% has only two variables we can represent it graphically in order to have a better understanding of the situation and to explain why all the algorithms agree that Assignment01 is the most important predictor.

Figure 9 shows that it is straightforward to identify the categories of students by setting some thresholds on the Assignment01 feature. For instance, most of the Weak students have grade zero in Assignment01.

7.6 Features importance: Dataset 2 - Stage 50%

The variables' importance for NN1, NN2, k-NN, and NB is described in Table 9, whereas the variables' importance for NN3, LR, RF, and SVM is described in Table 10.

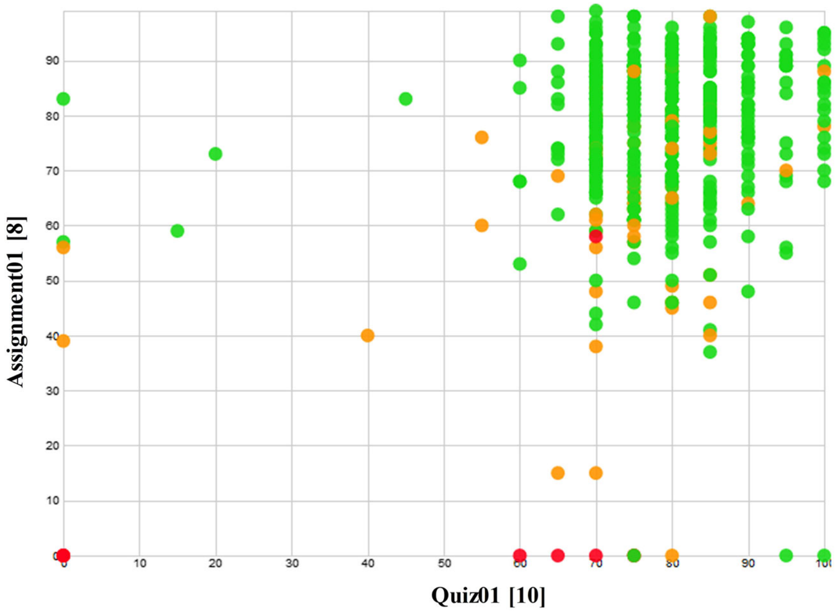


Fig. 9 Dataset 2 - Stage 20% - scatter plot

Table 10 Dataset 2 - Stage 50% - NN3, LR, RF, and SVM-RBF, Features' importance

Ranking	Feature
1	Assignment01
2	Assignment02
3	Midterm Exam
4	Quiz01

Based on the aforementioned results, it can be seen that assignments are better indicators of the student performance. This can be attributed to several factors. The first is the fact that assignments typically allow instructors to assess the three higher levels of cognition as per Bloom's taxonomy, namely analysis, synthesis, and evaluation [18]. As such, assignments provide a better indicator of the learning level that a student has achieved and consequently can give insights about his/her potential performance in the class overall. Another factor is that students tend to have more time to complete assignments. Moreover, they are often allowed to discuss issues and problems among themselves. Thus, students not performing well in the assignments may be indicative of them not fully comprehending the material. This can result in the students receiving a lower overall final course grade.

8 Experimental results and discussion

Matlab 2018 was used to build the Neural Networks classifiers, whereas all the other models were built using R.

All possible combinations of ensembles of eight baggings of models (256 in total) were computed for the initial Train-Test split and for 5 extra splits. For each dataset, the average of the performances, namely *averaged Gini Index*, on the 6 splits was used to select the most robust ensemble learner. In addition, we computed the p-values of all the ensembles for all the splits aiming to select the ensemble learner *with highest averaged Gini index that was also statistically significant on every split*. Note that the contribution of each feature is determined by the base learner model being used in the ensemble as per the ranking determined for each dataset at each stage. For example, if the RF learner is part of the ensemble being considered for Dataset 1 at 50% stage, the first split is done over feature ES 4.1, the second split is over feature ES 4.2, and so on.

In the following sections we will see the results obtained for the two datasets at each stage.

Table 11 Dataset 1 - Stage 20% Ensemble (NN2) Confusion Matrix $\tau_F = 0.158$, $\tau_G = 0.310$, $\tau_W = 0.682$

	F	G	W
F	1	1	1
G	1	4	2
W	0	0	5

8.1 Results: Dataset 1 - Stage 20%

If we based our choice only on the Gini index corresponding to the initial split, the ensemble learner we would have selected for Dataset 1 at Stage 20% would have been formed by NB, NN1, and SVM-RBF. Instead, the ensemble learner that appears to be the most stable on every split and with statistical significance is the one formed by a bagging of the NN2 model and the combination of the bagging of NN2 and NB as a bagging ensemble. Figure 10 shows the results obtained by inferring the ensemble on the initial test sample.

Classes G, F, W have Gini Indices equal to 46.4%, 38.9% and 94.0% respectively. Hence, the Averaged Gini Index is

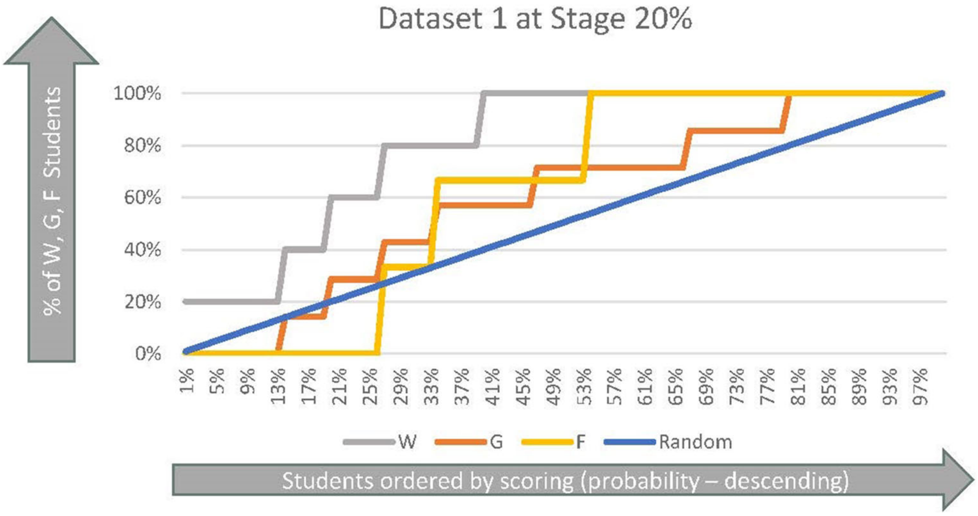


Fig. 10 Dataset 1 - Stage 20% - Ensemble Learner

Table 12 Dataset 1 - Stage 20% - Ensemble Performances

	Precision	Recall	F-measure	False Positive Rate
F	0.33	0.50	0.40	0.50
G	0.57	0.80	0.67	0.20
W	1.00	0.63	0.77	0.38
Avg	0.64	0.64	0.61	0.36

59.8%. On average, on Test sample and the 5 extra splits the Averaged Gini Index is 62.1%. The corresponding p-values are all less than 0.03.

The confusion matrix for the Test sample (consisting of 15 students), obtained as explained in Section 6, is shown in Table 11.

Table 12 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 66.7%. Although this may seem to be low, it actually outperforms all of the base learners used to create the bagging ensemble. Note that the low accuracy may be attributed to the fact that the dataset itself is small and hence did not have enough instances to learn from.

8.2 Results: Dataset 1 - Stage 50%

For Dataset 1 at Stage 50%, *none of the ensembles we constructed were statistically significant* even if their Averaged Gini Indices are on average higher than the ones obtained for Dataset 1 at Stage 20%. In fact, the performance for class F gets worse when we add the three variables. More precisely, when we add Features *ES4.1*, *ES4.2* and *ES5.1* to Dataset 1 at stage 20% obtaining Dataset 1 at stage 50%, they end up being the ones that have the main impact on the predictions. These variables help distinguish between W and G and in fact the performance corresponding to these two classes improve. However, since Fair students are closely correlated with the Good students class, the classifier becomes less confident in predicting the Fair students.

The best ensemble in terms of performance is the one obtained from a bagging of NB and k-NN. The Averaged Gini Index on 6 splits is 74.9% and on the initial test sample the Averaged Gini Index is 86.5%. Figure 11 shows the performance obtained on Split 1, having Averaged Gini Index equals 50%, with Gini Indices -22.2%, 76.8%, 86.0% respectively on Classes F, G and W. On a different split, the ensemble formed by a bagging of NB and k-NN on Dataset 1 at stage 20% gives Gini Indices 77.8%, 53.6% and 48.0% respectively on Classes F, G and W. This proves that the performance heavily depends on the split. In general, when we add the new three features (obtaining Dataset1 at stage 50%), the performance improves on classes G and W whereas it gets much worse for class F.

The confusion matrix obtained is shown in Table 13.

Table 14 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false

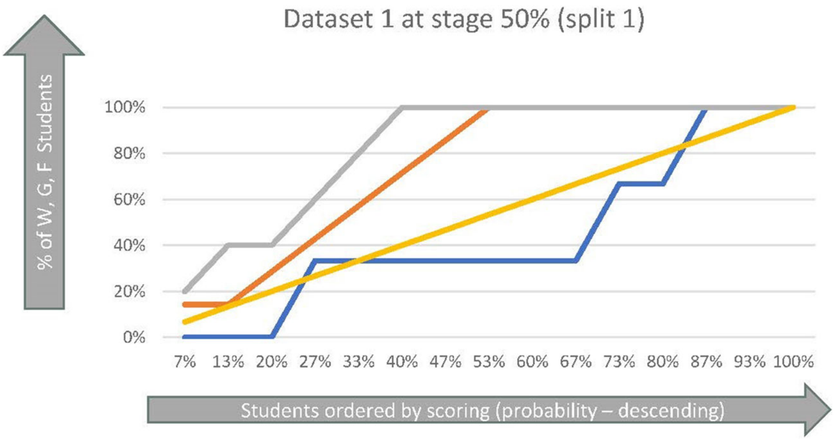


Fig. 11 Dataset 1 - Stage 50% - Ensemble Learner

Table 13 Dataset 1 - Stage 50% Ensemble (NB and k-NN) Confusion Matrix $\tau_F = 0.10$, $\tau_G = 0.29$, $\tau_W = 0.88$

	F	G	W
F	0	2	1
G	0	7	0
W	1	1	3

positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 66.7%. Again, the bagging ensemble outperforms all of the base learners used to create it despite it potentially being low. This is due to the fact that the dataset itself is small and hence did not have enough instances for the ensemble to learn from. Note that we cannot compute the F-measure for class F as Precision and Recall are zero.

It is worth noting that the low average Gini index can be attributed to 2 main reasons:

- This dataset is a small dataset.
- The Fair class is highly correlated with the Good students class. Hence, this is causing some confusion to the models being trained.

This is further highlighted by the large false positive rate obtained for the Fair class.

8.3 Results: Dataset 2 - Stage 20%

The ensemble learner selected for Dataset 2 at Stage 20% is formed by bagging of NB, k-NN, LR, NN2, and SVM-RBF. For instance, we show the results corresponding to the initial test sample. For each class, we normalized the scores obtained by the five baggings of models on the test sample in order to make these probabilities comparable, then we averaged them. The performances obtained are shown in Fig. 12.

Classes G, F, W have Gini Indices equal to 48.1%, 38.6% and 99.7% respectively. The confusion matrix associated is shown in Table 15.

Table 14 Dataset 1 - Stage 50% - Ensemble Performances

	Precision	Recall	F-measure	False Positive Rate
F	0.00	0.00	-	1.00
G	1.00	0.70	0.82	0.30
W	0.60	0.75	0.67	0.25
Avg	0.53	0.48	-	0.52

Furthermore, Table 16 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 88.2%, which is very good compared with respect to the performances obtained for Dataset 1. In a similar fashion to Dataset 1, the bagging ensemble outperforms the base learners in terms of classification accuracy.

8.4 Results: Dataset 2 - Stage 50%

The ensemble learner selected for Dataset 2 at Stage 50% is formed by a bagging of LR models only. The performances obtained on the initial test sample are shown in Fig. 13. On average, almost all the ensembles we constructed have very good performances and are statistically significant. The ensemble we selected is very robust on every split.

Classes G, F, W have Gini Indices equal to 92.3%, 90.7% and 99.3% respectively.

The confusion matrix obtained is shown in Table 17.

Table 18 illustrates the performances of the ensemble learner in terms of precision, recall, F-measure and false positive rate per class and on average. These quantities depend on the thresholds τ_F, τ_G and τ_W and the way we defined the predictions. The Accuracy is 93.1%. Again, the bagging ensemble at this stage also outperforms the base learners in terms of classification accuracy.

8.5 Performance comparison with base learners

Table 19 shows the classification accuracy of the different base learners in comparison with the average accuracy across the 256 splits of the bagging ensemble. It can be seen that the bagging ensemble on average outperforms all of the base learners at the two course delivery stages for both datasets. This is despite the fact that some of the splits may have had poor distribution which often leads to lower classification accuracy of the ensemble. This further highlights and emphasizes the effectiveness of the proposed ensemble in accurately predicting and identifying students who may need help.

8.6 Results summary

The performances obtained for Dataset 1 and Dataset 2 are very different. For Dataset 1, the models performances depend strongly on the splits. For instance, the same ensemble might perform very well on certain splits but have very low Averaged Gini Index on others, due to a negative Gini index on class F. Moreover, only 25% of the ensembles

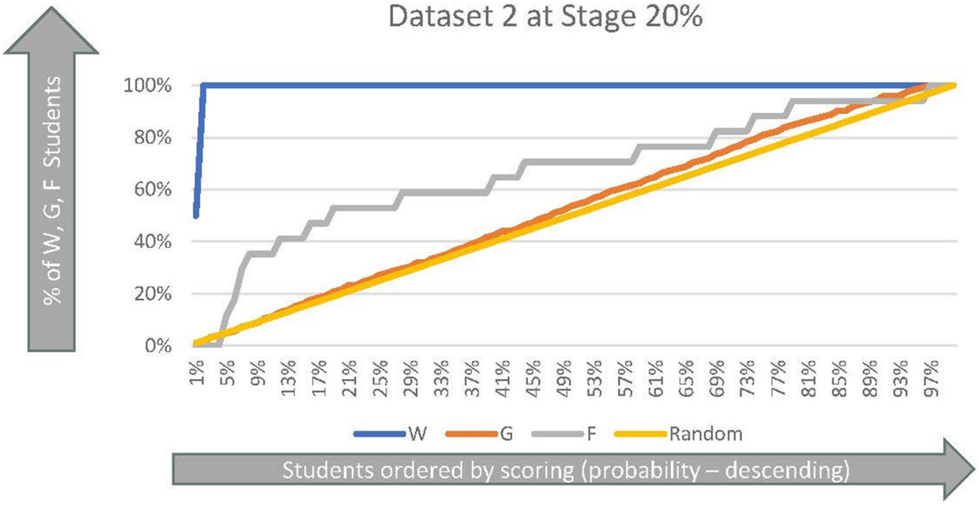


Fig. 12 Dataset 2 - Stage 20% - Ensemble Learner

for Dataset 1 at the 20% had averaged Gini Index above 50% and of all the ensembles only one of them is statistically significant, the one corresponding to a bagging of NN2 models.

Although the evidence shows that this ensemble performs decently on each split we have considered for our experiments, we cannot assume that this is true on every other possible split we might have chosen instead. The problem is so dependent on the split selected, that even the ensemble we chose results in lack of robustness and poor performances.

For Dataset 1 at stage 50%, the averaged Gini Index is in general higher than the averaged Gini Index obtained at stage 20% because the Gini Indices corresponding to classes G (Good students) and class W (Weak students) improve when we add the three features *ES4.1, ES4.2, ES5.1*. Since the Fair students class is highly correlated with the Good students class, the consequence is that when we add the best predictors, they predict incorrectly the Fair students. Consequently, the Gini Index for class F for each ensemble and for almost every split is negative or very low, leading to statistically insignificant results. In particular, there is not an ensemble among the 256 constructed such that the p-value corresponding to class F is lower than 0.03 on every split. Note that the ensemble chosen at the 50% stage is bagging of NB and k-NN. Although the ensemble was not statistically significant due to class F, it was statistically significant for the target class W.

For this reason, even though for completeness we are going to show the results for Dataset 1 at both stages, it is important to point out that if we were aiming to classify correctly the students for Dataset 1 and to use the classifier for applications in real world, we should not include the last three features, i.e. we should use Dataset 1 at stage 20%.

Dataset 2 was easier to deal with and also the choice of the best ensemble was straightforward. For Dataset 2 at stage 50%, 88% of the ensembles have averaged Gini Indices above 90%, and 96% of the ensembles were statistically significant.

For Dataset 2, the highest averaged Gini Index led us to choose:

– the ensemble of baggings of NB, k-NN, LR, and NN2 for the 20% stage.
– the ensemble consisting of bagging of LR for the 50% stage.

Note that in general, it is better to perform the prediction at the 50% stage rather than at the 20% stage. This is due to

Table 15 Dataset 2 - Stage 20% Ensemble (NB,k-NN,LR,NN2,SVM) Confusion Matrix $\tau_F = 0.12$, $\tau_G = 0.62$, $\tau_W = 0.40$

	F	G	W
F	5	11	1
G	5	120	0
W	0	0	2

Table 16 Dataset 2 - Stage 20% - Ensemble Performances

	Precision	Recall	F-measure	False Positive Rate
F	0.29	0.50	0.37	0.50
G	0.96	0.92	0.94	0.08
W	1.00	0.67	0.80	0.33
Avg	0.75	0.69	0.70	0.31

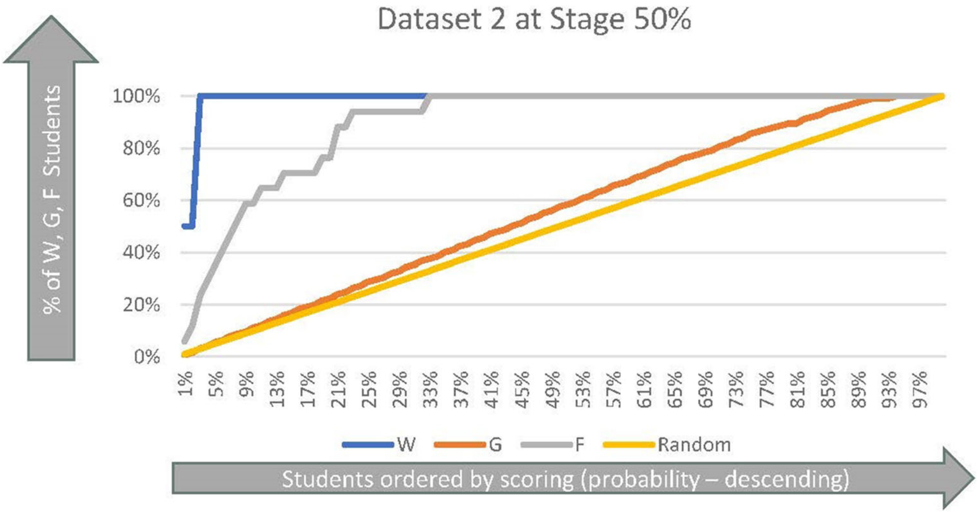


Fig. 13 Dataset 2 - Stage 50% - Ensemble Learner

Table 17 Dataset 2 - stage 50% Ensemble (LR) Confusion Matrix $\tau_F = 0.12$, $\tau_G = 0.62$, $\tau_W = 0.30$

	F	G	W
F	11	5	1
G	3	122	0
W	1	0	1

Table 18 Dataset 2 - Stage 50% - Ensemble Performances

	Precision	Recall	F-measure	False Positive Rate
F	0.65	0.73	0.69	0.27
G	0.98	0.96	0.97	0.04
W	0.50	0.50	0.50	0.50
Avg	0.71	0.73	0.72	0.27

the fact that more features are collected at the 50% stage, resulting in the learners being able to gain more information. Although this observation was not evident for dataset 1, this is due to the dataset being small with only a few instances of the F class that were at the border between the G and W classes. However, it was observed that the F-measure was high at both stages for the target class W.

For dataset 2, the results showed that indeed predicting at the 50% stage is better since the performance of the ensemble improved with the added number of features. However, the results at the 20% stage were still valuable as they helped provide vital insights at an extremely early stage of the course delivery as evident by the fact that the F-measure was close to 0.7 at that stage.

9 Conclusion, research limitations, and future work

In this paper, we investigated the problem of identifying student who may need help during course delivery time

Table 19 Performance of Bagging Ensemble and Base Learners

Accuracy Technique	Dataset 1		Dataset 2	
	Stage 20%	Stage 50%	Stage 20%	Stage 50%
RF	46.7%	66.6%	82.8%	89.0%
NN	66.7%	60%	86.2%	91.7%
K-NN	60%	66.6%	86.2%	89.0%
NB	53.3%	66.6%	85.5%	85.5%
LR	53.3%	53.3%	86.9%	90.3%
SVM	46.7%	33.3%	86.2%	90.3%
Ensemble	**66.7%**	**66.7%**	**88.2%**	**93.1%**

for an e-Learning environment. The goal was to predict the students' performance by classifying them into one of three possible classes, namely Good, Fair, and Weak. In particular, we tackled this multi-class classification problem for two educational datasets at two different course delivery stages, namely at the 20% and 50% mark. We trained eight baggings of models for each dataset and considered all the possible ensembles that could be generated by considering the scores produced by inferring them on a test sample.

We compared the performances, and concluded that the ensemble learners to be selected are formed by:

- a bagging of NN2 models for Dataset 1 at stage 20%.
- a bagging of NB and k-NN models for Dataset 1 at stage 50%.
- a bagging of NB, k-NN, LR, NN2, and SVM-RBF for Dataset 2 at stage 20%.
- a bagging of LR models for Dataset 2 at stage 50%.

whereas it was not possible to select a good ensemble for Dataset 1 at stage 50% as none of the ensembles was statistically significant.

The results are good for Dataset 2 both in terms of Averaged Gini Index and p-values, especially if we consider the issues encountered. In particular, the issues are mainly the size of Dataset 1 and the unbalanced nature of Dataset 2. In turn, this makes the multi-class classification problem more complex. This was evident by the fact that it was impossible to find a good classifier for Dataset 1 at stage 50% and that the performance obtained for Dataset 1 at stage 20% was poor.

Based on the aforementioned research limitations, below are some suggestions for our future work:

- The best way to face the dataset size issue would be to have more data available, by collecting training and testing datasets for every time the course is offered.
- We also suggest to perform several additional splits for Dataset 1 at Stage 20% to check the robustness of the model as well as the statistical significance.
- It might be worth trying to optimize the topology of the neural network with a dedicated algorithm. Even though our choice was based on recent literature it is unlikely that we reached the optimum. One could consider to try, for instance, all the possible combinations with 1, 2, and 3 layers and 1,..,20 neurons in each layer. If we considered all such combinations we would have had $20 + 20^2 + 20^3$ NNs to train. Of course it would be computationally not viable and would probably result in a massive over-fitting. However, there are several approaches proven to be effective in this kind of tasks, such as genetic optimization or pre-trained models capable to predict the optimal topology of a network for a given problem, considering parameters such as the dimension of the dataset and the intensity of the noise [19].

Datasets' Permissions

- Dataset 1: The dataset is publicly available at: https://sites.google.com/site/learninganalyticsforall/data-sets/epm-dataset. Use of this data set in publications was acknowledged by referencing [63].
- Dataset 2: All permissions to use this dataset were obtained through The University of Western Ontario's Research Ethics Office. This office approved the use of this dataset for research purposes.

Acknowledgments This study was funded by Ontario Graduate Scholarship (OGS) Program.

Compliance with Ethical Standards

Conflict of interests The authors declare that they have no conflict of interest.

Informed Consent This study does not involve any experiments on animals.

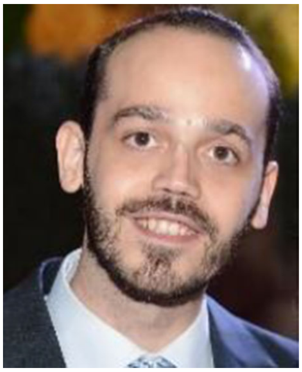
References

1. Abdul Aziz A, Ismail NH, Ahmad F (2013) Mining students' academic performance. Journal of Theoretical and Applied Information Technology 53(3):485–485
2. Ahmed ABED, Elaraby IS (2014) Data mining: a prediction for student's performance using classification method. World Journal of Computer Application and Technology 2(2):43–47
3. Aly M (2005) Survey on multiclass classification methods. Neural Network 19:1–9
4. Asogbon MG, Samuel OW, Omisore MO, Ojokoh BA (2016) A multi-class support vector machine approach for students academic performance prediction. Int J Multidisciplinary and Current Research 4
5. Athani SS, Kodli SA, Banavasi MN, Hiremath PS (2017) Student performance predictor using multiclass support vector classification algorithm. In: 2017 international conference on signal processing and communication (ICSPC). IEEE, pp 341–346
6. Baradwaj BK, Pal S (2012) Mining educational data to analyze students' performance. arXiv:12013417
7. Bhardwaj BK, Pal S (2012) Data mining: a prediction for performance improvement using classification. arXiv:12013418
8. Buffardi K, Edwards SH (2014) Introducing codeworkout: an adaptive and social learning environment. In: Proceedings of the 45th ACM technical symposium on computer science education, ACM, SIGCSE '14, pp 724–724. https://doi.org/10.1145/2538862.2544317
9. Bühlmann P (2012) Bagging, boosting and ensemble methods. In: Handbook of computational statistics. Springer, Berlin, pp 985–1022
10. Bühlmann P, Yu B et al (2002) Analyzing bagging. The Annals of Statistics 30(4):927–961

11. Chang YC, Kao WY, Chu CP, Chiu CH (2009) A learning style classification mechanism for e-learning. Computers & Education 53(2):273–285
12. Chen X, Vorvoreanu M, Madhavan K (2014) Mining social media data for understanding students' learning experiences. IEEE Transactions on Learning Technologies 7(3):246–259. https://doi.org/10.1109/TLT.2013.2296520
13. Daniel J, Vázquez Cano E, Gisbert Cervera M (2015) The future of moocs: adaptive learning or business model? International Journal of Educational Technology in Higher Education 12(1):64–73. https://doi.org/10.7238/rusc.v12i1.2475
14. Daradoumis T, Bassi R, Xhafa F, Caballe S (2013) A review on massive e-learning (mooc) design, delivery and assessment. In: 2013 eighth international conference on p2p, parallel, grid, cloud and internet computing, pp 208–213
15. Dhar V, Tickoo A, Koul R, Dubey B (2010) Comparative performance of some popular artificial neural network algorithms on benchmark and function approximation problems. Pramana 74(2):307–324
16. Essalmi F, Ayed LJB, Jemni M, Graf S, Kinshuk (2015) Generalized metrics for the analysis of e-learning personalization strategies. Computers in Human Behavior 48:310–322. https://doi.org/10.1016/j.chb.2014.12.050
17. Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. AI magazine 17(3):37–37
18. Feldman L (2006) Designing homework assignments: from theory to design. Age 4:1
19. Fiszelew A, Britos P, Ochoa A, Merlino H, Fernández E, García-Marínez R (2007) Finding optimal neural network architecture using genetic algorithms. Advances in Computer Science and Engineering Research in Computing Science 27:15–24
20. Fluss R, Faraggi D, Reiser B (2005) Estimation of the youden index and its associated cutoff point. Biometrical Journal: Journal of Mathematical Methods in Biosciences 47(4):458–472
21. Fok WW, He Y, Yeung HA, Law K, Cheung K, Ai Y, Ho P (2018) Prediction model for students' future development by deep learning and tensorflow artificial intelligence engine. In: 2018 4th international conference on information management (ICIM). IEEE, pp 103–106
22. Fujita H et al (2019) Neural-fuzzy with representative sets for prediction of student performance. Appl Intell 49(1):172–187
23. Gevrey M, Dimopoulos I, Lek S (2003) Review and comparison of methods to study the contribution of variables in artificial neural network models. Ecological Modelling 160(3):249–264
24. Guyon I, Lemaire V, Boullé M, Dror G, Vogel D (2010) Design and analysis of the kdd cup 2009: fast scoring on a large orange customer database. ACM SIGKDD Explorations Newsletter 11(2):68–76
25. Hand DJ, Till RJ (2001) A simple generalisation of the area under the roc curve for multiple class classification problems. Machine Learning 45(2):171–186
26. Hijazi ST, Naqvi S (2006) Factors affecting students' performance. Bangladesh E-Journal of Sociology 3(1)
27. Hosseinzadeh A, Izadi M, Verma A, Precup D, Buckeridge D (2013) Assessing the predictability of hospital readmission using machine learning. In: Twenty-fifth IAAI conference
28. Injadat M, Salo F, Nassif AB (2016) Data mining techniques in social media: a survey. Neurocomputing 214:654–670
29. Injadat M, Salo F, Nassif AB, Essex A, Shami A (2018) Bayesian optimization with machine learning algorithms towards anomaly detection. In: 2018 IEEE global communications conference (GLOBECOM), pp 1–6. https://doi.org/10.1109/GLOCOM.2018.8647714
30. Injadat M, Moubayed A, Nassif AB, Shami A (2020) Systematic ensemble model selection approach for educational data mining. Knowledge-Based Systems 200:105992. https://doi.org/10.1016/j.knosys.2020.105992. http://www.sciencedirect.com/science/article/pii/S0950705120302999
31. Jain A, Solanki S (2019) An efficient approach for multiclass student performance prediction based upon machine learning. In: 2019 International conference on communication and electronics systems (ICCES). IEEE, pp 1457–1462
32. Kaggle Inc (2019) Kaggle. https://www.kaggle.com/
33. Karaci A (2019) Intelligent tutoring system model based on fuzzy logic and constraint-based student model. Neural Computing and Applications 31(8):3619–3628. https://doi.org/10.1007/s00521-017-3311-2
34. Kaur G, Singh W (2016) Prediction of student performance using weka tool. An International Journal of Engineering Sciences 17:8–16
35. Kehrwald B (2008) Understanding social presence in text-based online learning environments. Distance Education 29(1):89–106. https://doi.org/10.1080/01587910802004860
36. Khan B, Khiyal MSH, Khattak MD (2015) Final grade prediction of secondary school student using decision tree. Int J Comput Appli 115(21)
37. Khribim MK, Jemni M, Nasraoui O (2008) Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval. In: 2008 eighth IEEE international conference on advanced learning technologies, pp 241–245. https://doi.org/10.1109/ICALT.2008.198
38. Klamma R, Chatti MA, Duval E, Hummel H, Hvannberg ET, Kravcik M, Law E, Naeve A, Scott P (2007) Social software for life-long learning. Journal of Educational Technology & Society 10(3):72–83
39. Koch P, Wujek B, Golovidov O, Gardner S (2017) Automated hyperparameter tuning for effective machine learning. In: Proceedings of the SAS global forum 2017 conference, pp 1–23
40. Kotsiantis S, Patriarcheas K, Xenos M (2010) A combinational incremental ensemble of classifiers as a technique for predicting students' performance in distance education. Knowl-Based Syst 23(6):529–535
41. Kuhn M et al (2008) Building predictive models in r using the caret package. Journal of statistical software 28(5):1–26
42. Lerman RI, Yitzhaki S (1984) A note on the calculation and interpretation of the gini index. Economics Letters 15(3-4):363–368
43. Lorenz MO (1905) Methods of measuring the concentration of wealth. Publications of the American statistical association 9(70):209–219
44. Luan J (2002) Data mining and its applications in higher education. New Directions for Institutional Research 2002(113):17–36. https://doi.org/10.1002/ir.35
45. Lv C, Xing Y, Zhang J, Na X, Li Y, Liu T, Cao D, Wang FY (2017) Levenberg–marquardt backpropagation training of multilayer neural networks for state estimation of a safety-critical cyber-physical system. IEEE Transactions on Industrial Informatics 14(8):3436–3446
46. Ma Y, Liu B, Wong CK, Yu PS, Lee SM (2000) Targeting the right students using data mining. In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 457–464
47. Marquardt DW (1963) An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics 11(2):431–441
48. Márquez-Vera C, Cano A, Romero C, Ventura S (2013) Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalanced data. Applied Intelligence 38(3):315–330

49. Moubayed A, Injadat M, Nassif AB, Lutfiyya H, Shami A (2018) E-learning: challenges and research opportunities using machine learning data analytics. IEEE Access 6:39117–39138. https://doi.org/10.1109/ACCESS.2018.2851790
50. Moubayed A, Injadat M, Shami A, Lutfiyya H (2018) DNS typo-squatting domain detection: a data analytics & machine learning based approach. In: 2018 IEEE global communications conference (GLOBECOM). IEEE, pp 1–7
51. Moubayed A, Injadat M, Shami A, Lutfiyya H (2018) Relationship between student engagement and performance in e-learning environment using association rules. In: 2018 IEEE world engineering education conference (EDUNINE), pp 1–6. https://doi.org/10.1109/EDUNINE.2018.8451005
52. Moubayed A, Aqeeli E, Shami A (2020) Ensemble-based feature selection and classification model for DNS typo-squatting detection. In: 33rd Canadian conference on electrical and computer engineering (CCECE'20). IEEE, pp 1–6
53. Moubayed A, Injadat M, Shami A, Lutfiyya H (2020) Student engagement level in e-learning environment. Clustering using k-means. American Journal of Distance Education. https://doi.org/10.1080/08923647.2020.1696140
54. Netflix Inc (2009) Netflix competition. https://www.netflixprize.com/
55. Nguyen D, Widrow B (1990) Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In: 1990 IJCNN international joint conference on neural networks. IEEE, pp 21–26
56. Pal S (2012) Mining educational data to reduce dropout rates of engineering students. Int J Inform Eng Electron Business 4(2):1
57. Prasad GNR, Babu AV (2013) Mining previous marks data to predict students performance in their final year examinations. Int J Eng Res Technol 2(2):1–4
58. Ramaswami M (2014) Validating predictive performance of classifier models for multiclass problem in educational data mining. International Journal of Computer Science Issues (IJCSI) 11(5):86
59. Rana S, Garg R (2016) Evaluation of students' performance of an institute using clustering algorithms. Int J Appl Eng Res 11(5):3605–3609
60. Romero C, Ventura S (2007) Educational data mining: a survey from 1995 to 2005. Expert systems with applications 33(1):135–146
61. Rosenberg MJ, Foshay R (2002) E-learning: strategies for delivering knowledge in the digital age. Performance Improvement 41(5):50–51. https://doi.org/10.1002/pfi.4140410512. https://onlinelibrary.wiley.com/doi/abs/10.1002/pfi.4140410512, https://onlinelibrary.wiley.com/doi/pdf/10.1002/pfi.4140410512
62. Saxena R (2015) Educational data mining: performance evaluation of decision tree and clustering techniques using weka platform. Int J Comput Sci Business Inform 15(2):26–37
63. Vahdat M, Oneto L, Anguita D, Funk M, Rauterberg M (2015) A learning analytics approach to correlate the academic achievements of students with interaction data from an educational simulator. In: Design for teaching and learning in a networked world. Springer International Publishing, Cham, pp 352–366
64. Vujicic T, Matijevic T, Ljucovic J, Balota A, Sevarac Z (2016) Comparative analysis of methods for determining number of hidden neurons in artificial neural network. In: Central European conference on information and intelligent systems, faculty of organization and informatics Varazdin, p 219
65. Wang X, Zhang Y, Yu S, Liu X, Yuan Y, Wang F (2017) E-learning recommendation framework based on deep learning. In: 2017 IEEE international conference on systems, man, and cybernetics (SMC), pp 455–460. https://doi.org/10.1109/SMC.2017.8122647
66. Yang L, Moubayed A, Hamieh I, Shami A (2019) Tree-based intelligent intrusion detection system in internet of vehicles. In: 2019 IEEE global communications conference (GLOBECOM)

MohammadNoor Injadat received the B.Sc. and M.Sc. degrees in Computer Science from Al Al-Bayt University and University Putra Malaysia in Jordan and Malaysia in 2000 and 2002, respectively. He obtained a Master of Engineering in Electrical and Computer Engineering from University of Western Ontario in 2015. He is currently working toward his Ph.D. degree in Software Engineering at the Department of Electrical and Computer Engineering, University of Western Ontario in Canada. His research interests include data mining, machine learning, social network analysis, e-learning analytics, and network security.



Abdallah Moubayed received his Ph.D. in Electrical & Computer Engineering from the University of Western Ontario in August 2018, his M.Sc. degree in Electrical Engineering from King Abdullah University of Science and Technology, Thuwal, Saudi Arabia in 2014, and his B.E. degree in Electrical Engineering from the Lebanese American University, Beirut, Lebanon in 2012. Currently, he is a Postdoctoral Associate in the Optimized Computing and Communications (OC2) lab at University of Western Ontario. His research interests include wireless communication, resource allocation, wireless network virtualization, performance & optimization modeling, machine learning & data analytics, computer network security, cloud computing, and e-learning.

Ali Bou Nassif received his Ph.D. degree in Electrical and Computer engineering from The University of Western Ontario, London, Ontario, Canada (2012). He is currently an Associate Professor and an Assistant Dean of the Graduate Studies, University of Sharjah, United Arab Emirates, and an Adjunct Research Professor with Western University. He has published more than 60 papers in international journals and conferences. His interests are Machine Learning and Soft Computing, Software Engineering, Cloud Computing and Service Oriented Architecture (SOA), and Mobile Computing. Ali is a registered professional engineer in Ontario, as well as a member of IEEE Computer Society.

Abdallah Shami is a Professor at the ECE department at Western University, Ontario, Canada. Dr. Shami is the Director of the Optimized Computing and Communications Laboratory at Western. He is currently an Associate Editor for IEEE Transactions on Mobile Computing, IEEE Network, and IEEE Communications Tutorials and Survey. He has chaired key symposia for IEEE GLOBECOM, IEEE ICC, IEEE ICNC, and ICCIT. He was the elected Chair of the IEEE Communications Society Technical Committee on Communications Software (2016–2017) and the IEEE London Ontario Section Chair (2016–2018).