Machine Learning Prediction and Recommendation Framework to Support Introductory Programming Course

https://doi.org/10.3991/ijet.v16i17.18995

Ijaz Khan[1](✉), Abdul Rahim Ahmad[1], Nafaa Jabeur[2], Mohammed Najah Mahdi[1]
[1] Universiti Tenaga Nasional, Kajang, Malaysia
[2] German University of Technology, Muscat, Oman
ijaz@buc.edu.om

Abstract—The new students struggle to understand the introductory programming courses, due to its intricate nature, which results in higher dropout and increased failure rates. Despite implementing productive methodologies, the instructor struggles to identify the students with distinctive levels of skills. The modern institutes are looking for technology-equipped practices to classify the students and prepare personalized consultation procedures for each class. This paper applies decision tree-based machine learning classifiers to develop a prediction model competent to forecast the outcome of the introductory programming students at an early stage of the semester. The model is then transformed into an adaptive consultation framework which generates three types of colored signals; red, yellow, and green which illustrates whether the student is performing low, average, or high respectively. This provides an opportunity for the instructor to set precautionary measures for low performing students and set complicated tasks that help the highly skilled students to improve their skills further. The experiments compare a set of decision tree-based classifiers and conclude J48 as an efficient model in classifying students in all classes with high accuracy, sensitivity, and F-measure. Even though the aim of the research is to focus on introductory programming courses, however, the framework is flexible and can be implemented in other courses.

Keywords—student performance prediction, data mining, machine learning, decision tree, introductory programming

1 Introduction

Introductory programming courses form the basis for the students in computer science major. The main objective of these courses is to empower the students with the fundamental skills required to develop computer programs that can solve real-world problems. These courses are positioned at the early stages of the study plan and thus appear a nightmare for the students. Programming requires realistically particular skills that students may struggle to obtain with the traditional learning methods. Consequently, novice programming students face diverse types of complexities which result

in high dropout and failure rates [1, 2]. In this context, the instructors may not be able to achieve the prime objectives of the course with the implementation of traditional teaching methodologies and may face various challenges related to the course's nature, the students' characteristics, and the adopted teaching methods [3].

The students often struggle with introductory programming courses, therefore, the instructor implements constructive teaching methods throughout the semester[4]. These methods may appear beneficial for the weak students, however, the students with exceptional programming skills have to slow their pace and go along with the instructor. Therefore, a major problem the instructor faces is to label the students in accordance with their learning pace. Once classified, the instructor can prepare procedures to support the slow learners so their risk of failing the course minimize. Similarly, the instructor can further augment the capabilities of fast learners with supplementary material. This classification of students is helpful not only for the students but also for the instructor.

Numerous studies have been conducted to support introductory programming courses. Several studies aim to investigate the diverse range of mistakes the students make[5], and their behavior towards the error messages[6]. Several researchers have implemented application software in which the students evaluate their own problem solving skills by interacting with a graphical interface[7]. One of the prime objectives of the existing studies is to identify the student's features which affect their learning [8]. For instance, several studies found that mathematical ability and exposure to mathematics courses are important predictors of the performance of introductory programming courses[9, 10]. Numerous authors apply machine learning classifiers to automatically identify the students at risk of failures, for instance, Liao et al. [11], use support vector machines to automatically identify students at risk of failure based on data collected from instructors when they make use of the Peer Instruction, an active-learning methodology, pedagogy. Quille et al. [12] propose a prediction model to predict student success early in an introductory programming module. Similarly, several types of research make use of students academic, demographic, and social features to develop prediction models which can forecast student's outcome based on several prediction features at a specific stage of the semester [13, 14].

Introductory programming courses form the basis of the computer-related majors and their indistinct nature endorses the need of models to classify the students and deal with each class of student with distinctive procedures. The previous research lacks a model which not only classifies the programming students but also adapt the model into a framework easily understandable by the instructor. In fact in programming courses, the instructor must have an opportunity to identify the students with low, medium and excellent programming skills and provide adaptive consultation to each group of students. This leads to the need of a model which can effectively identify the poor performing as well as the excellent programmers at the early stages of the semester to offer an opportunity to the instructor to design procedures for each class of students. The strength of this research is to develop classification models and then transform the appropriate model into an easily explainable framework. The instructor gets an opportunity, at an early stage of the semester, to design precautionary procedures for the

students likely to end up with poor outcome, preventive procedures for average performing students and persuasive procedures for the excellent programmers. This research apply decision tree based classifiers to develop models research draws attention to the usefulness of decision tree classifier with regard to its easy interpretation. The rest of the paper is organized as; section 2 provides literature review of Learning Analytics, Educational Data Mining and the popular classes of machine learning classifiers and provides an outline of the student performance prediction models. Section 3 provides materials and methods and section 4 provides the methodology and the experimental evaluation. Section 5 concludes the paper.

2 Literature review

Learning Analytics (LA) and Educational Data Mining (EDM) appears supportive for higher education institutions. The aim of both LA and EDM is to understand and optimize the learning environment, however, EDM primarily focuses on automated discovery, whereas LA focuses on human judgment [15]. EDM starts with a dataset and discover the hidden pattern. Contrary, LA usually starts with hypotheses and carries out research to assess the learning theories about the learning environment of the learners [16]. The data mining algorithms apply eminent techniques to the data and extracts momentous information [17]. Monitoring student academic performance is one of the key applications of Learning Analytics. The notion observes student's academic performance and generates reports which describe the current status of the students and draw attention to the shortfalls in their academic activities. The performance prediction forecasts the final result of students based on several features that illuminate their current academic status. The significance of monitoring is enlightened by the fact that the students with inadequate academic outcomes may face severe consequences. For instance, they might lose their governmental sponsorship at several institutes. However, upon their successful identification, these students can be targeted for additional tutoring or counseling services. The early identification provides an opportunity to the student to adopt precautionary measures.

Machine Learning refers to learning from past experience to improve future performance automatically without any external assistance from humans. The supervised classifier takes the input dataset (training dataset) and constructs a classification model. The model constitutes of classification rules as it is discovered from the training dataset. The testing phase executes the model with a subset of dataset (testing dataset) taken from the training dataset to assess the performance of the model. The model gains knowledge from the training dataset and is evaluated with the testing dataset. Henceforth, it is ready to classify the instances from the validation dataset. Naïve Bayes, artificial neural networks, support vector machines, k-nearest neighbor, and decision tree are some of the most widely used categories of supervised learning classifiers. On the other hand in unsupervised learning the data instances do not have pre-determined classes.

Decision tree is one of the most widely used machine learning classifier. The decision tree follows a recursive technique to build a tree. Each decision node splits the

attributes into different branches based on the number of distinct values from that feature. Deciding an appropriate feature to be placed at the root node or any other internal nodes is a complicated step. Numerous techniques have been used to select the root node and subsequent decision nodes. Measuring Entropy is one of the key techniques which measure the randomness in the information being processed. The Information Gain (IG), another related statistical property, measures how well a given feature separates the training instances following the class labels. Therefore, the two terms have an inverse relationship and constructing a decision tree is all about discovering a feature that returns maximum IG and the minimum entropy.

A drawback appearing in IG is that it prefers the features with a large number of distinct values as the root node. This drawback is eradicated in Gain Ratio which is a modification of Information Gain. ID3 and C4.5 are machine learning classifiers constructing decision tree with Information Gain and Gain Ratio techniques respectively.

Machine Learning classification models are used in the pedagogical environment to forecast the learner's expected academic outcome. Numerous models have been proposed under different educational contexts to address student performance prediction. There are several models which are based on supervised classifiers [18-20]. This section provides an overview of the models developed to classify and predict student academic performance with decision tree classifiers.

Ahadi et al. [21] used machine learning algorithms to label the students of introductory programming courses as either high or low performers. The author sets up threshold value and the student obtaining grades above and below are labeled as high and low performers respectively. Chen et al. [22] developed decision tree and linear regression-based prediction models with a variety of features extracted from the institution's autograding system. Formerly, the instruction teams allot teaching hours reactively to support the students who request for additional tutoring; conversely, the produced model helps to identify the struggling students automatically and allot additional teaching hours to the instructor. Hamsa et al. [23] applied decision tree and Fuzzy Genetic Algorithm to support instructors in identifying the students whose academic status is unsatisfactory. The instructor can set up practices to support these students. Pandey et al. [24]compared a set of machine learning classifiers over several datasets. After that, the three classifiers were combined to form a single ensemble model based on the voting scheme.

Kausar et al. [25] applied ensemble techniques over a dataset to examine the relationship between students' features such as quizzes, assignment, exams marks and the final results. The experimental evaluation concludes Random Forest and Stacking classifiers with achieving accuracies of 77% and 78% respectively prevailing K-NN and Naïve bayes classifiers. Saa et al. [26] conducted a survey to collect data about the students' demographic, academic, and social behavior and then developed decision tree-based models to discover the features which influence students' academic performance. Kiu et al. [27] inspected the correlation between social activities the final grades of the students. Even though the decision tree appeared more useful than other classifiers, however, the results show a weak correlation between the social activities of the student and their final grades. Kaunang et al. [28] produced several decision tree-based

models with the data collected through questionnaires containing student's demographics, academic and family background features. Table 1 provides an overview of several prediction models, the place of study, and their essential components such as dataset size, applied classifiers, and measured evaluation metrics.

The literature highlights the need of prediction model to classify the programming students. The prediction model is imperfect with no execution. Therefore, the prediction model must have a tendency to yield a classification framework easily understandable by the instructor. The goal of this research is to develop decision tree based classification models and then transform the appropriate model into an easy to understand framework. The identification of poor performing students necessitate added significance. The 3-class classification is meaningful, so the "medium" class can appear as a hyperplane between high and low performing students and this can distinguish the poor performing programmers with higher accuracy. The instructor gets an opportunity to design distinctive procedures for each class of students.

3 Material and methods

3.1 Experimentation tool

The experiments are performed in Waikato Environment for Knowledge Analysis (WEKA) [29]. To split the dataset into training and testing, we use 10-fold cross-validation. The training dataset is divided into 10 equal length intervals; 9 are used for training and the tenth is used for testing. This process is repeated and a new testing interval is taken afterward in every iteration.

We apply 2 decision tree classifiers; J48 and RepTree. J48 is the implementation of C4.5 in WEKA. Reduced Error Pruning Tree ("RepTree"), also an implementation of C4.5, builds the tree based on the information gain or reducing the variance.

Table 1. A brief Summary of the existing student performance prediction models

Reference	Decision Tree Classifiers	Dataset Size	Class variables	Terrain of study
[30]	Decision tree (with Gini Index) Decision Tree (with Information Gain) Decision tree (with accuracy)	210	5	Pakistan
[31]	C4.5 with Genetic Algorithm	5409	n/a	Philippine
[21]	J48 Random Forest	86, 210	2	Finland
[22]	C4.5	428	3	Canada
[28]	J48 Random Forest	249	3	Indonesia
[27]	J48	395	2 & 5 5	Malaysia
[32]	J48	206	4	Nigeria
[33]	J48	383	2	United Kingdom
[34]	ID3 C4.5	100	4	Nigeria

3.2 Evaluation Metrics

Confusion Matrix visualizes the performance of a classification model. Table-2 provides a standard visualization of a confusion matrix for a prediction model.

To assess the performance of prediction models, this research use accuracy, precision, sensitivity, F-measure, and Mathew Correlation Coefficient (MCC). Accuracy assesses the overall performance of the prediction model and is computed as the ratio of correctly identified instances (TP and TN) and the total number of instances in the dataset. Precision calculates out of all the positive instances that the model predicted correctly, how many are positive. Sensitivity or Recall is a measure of all positive instances and the number of positive instances the model predicted correctly. F-Measure takes both precision and sensitivity into account and calculates their weighted average. Mathew Correlation Coefficient (MCC) [35]is a reliable statistical rate that assess the performance of the classifier in terms of how well it classified the instances in the correct class. Table 3 provides a list of evaluation metrics and their formulae.

Table 2. An illustration of standard confusion matrix

		Predicted Results	
		Positive	***Negative***
Actual Values	***Positive***	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 3. The formulae of evaluation metrics

Metric	**Formula**
Accuracy	(TP+TN)/(TP+FN+FP+TN)
Precision	TP/(TP+FP)
Sensitivity	TP/(TP+FN)
F-Measure	2x ((Precision x Recall)/(Precision + Recall))
MCC	(TP.TN-FP.FN)/√((TN+FN).(TP+FP).(TN+FP).(TP+FN)))

4 Experiments and results

The methodology used in this research, as shown in figure 1, consists of 4 phases; Data preparation, Data Pre-Processing, experimental evaluation, and model implementation. The first phase aims at the collection of dataset and making it compatible for machine learning classifier processing. Data pre-processing enhances the quality of the dataset so as the machine learning classifier can produce better results. The experimental evaluation compares the model developed with different decision tree classifiers and concludes a single model which can provide an appropriate solution. The model implementation phase converts the final prediction model into a framework which the instructor can implement in the course to achieve the addressed academic goals.

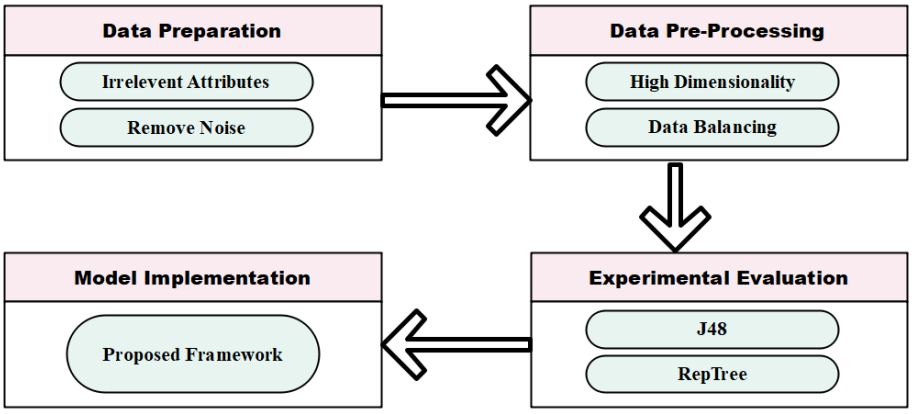

Fig. 1. The 4-phases methodologies followed in this research

4.1 Data preparation

The dataset source used in this research is taken from Al-Buraimi University College (BUC), an educational institution located in Oman. The data is the records of the students from 2 sections of an introductory programming course. There are a total of 50 instances in the training dataset with 9 features and one class feature. The prediction variable (Grade) has 3 classes; Low (marks less than 60%), Medium (equal to or more than 60% and less than 85%), and High (equal to or more than 85%). The objective of the prediction model is to predict the student's outcome soon after the midterm exam which takes place after the 6th week of a semester and 15 marks are allotted for it. Once, identified at this stage of the semester, the students still have 85% marks ahead to work for.

Data cleaning organize the dataset by dealing with missing value, data noise and resolving irregularity in the dataset. Initially, the irrelevant features, for instance, student ID, student name, and course code are erased from the dataset. These features do not play any role in the prediction and are rather associated with the student's privacy. Noisy data refers to the data which the classifier cannot understand and interpret correctly. This may include missing values where features in an instance may have irrelevant values or misleading values. The table 4 provides the features of the dataset and their brief description.

4.2 Data pre-processing

Removing high dimensionality and data balancing can further enhance the quality of the dataset.

Removing high dimensionality. The high dimensionality of the dataset points towards the high number of features in the dataset. Not all the features take part in the classification process; therefore, proper techniques must be used to eradicate the surplus features. To reduce the overlapping features, we applied a set of feature selection

algorithms; InfoGainAttributeEval, CorrelationAttributeEval, and CfsSubsetEval. Table 5 provides the output of the algorithms.

Table 4. The features and their essential details

Feature	Description	Scale of measuring
Gender	Gender of the student	Nominal (Male, Female)
Session	The shift in studying	Nominal (Morning, Evening)
Class_Duration	Duration of each class in hours	Nominal (one, one_half)
CGPA	Cumulative Grade Point Average	Real
Major	Major in Information Technology	Nominal (CS, IS, SE)
Degree	The certificate student is pursuing	Nominal (Bachelor, Diploma)
Year	The year of study	Nominal (Year-1 to Year-4)
Attendance	Percentage of class attendance at the time of data extraction	Real
Test1_Marks	Midterm exam marks (total 15 marks)	Real
Grade	Class variable.	Nominal (High, Medium, Low)

Table 5. The outcome of the feature selection algorithms

Feature	InfoGainAttributeEval	CorrelationAttributeEval	CfsSubsetEval
Test1_Marks	0.66463	0.3405	✓
CGPA	0.63985	0.3231	✓
Attendance	0.20942	0.1148	
Major	0.11002	0.1693	
Year	0.08614	0.1782	
Gender	0.07081	0.1823	✓
Class_Duration	0.04921	0.1278	
Session	0.04921	0.1278	
Degree	0.00174	0.0156	

Data Balancing. Classifiers trained with an imbalance dataset tend to predict the majority class (frequently occurring) more than the minority class (rarely occurring). In our dataset, the minority class has a very low number of instances (figure 2); therefore, we apply Synthetic Minority Over-sampling Technique (SMOTE) [36] to increase the number of instances in the minority class.

Figures 2 and 3 shows, the class distribution prior to and after SMOTE has been implemented respectively. Therefore, we have two datasets; without SMOTE (termed as original) and with SMOTE applied (termed as SMOTE) in the remainder of this paper.

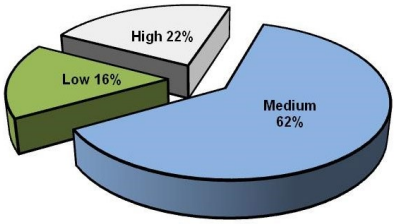

Fig. 2. Dataset before applying SMOTE

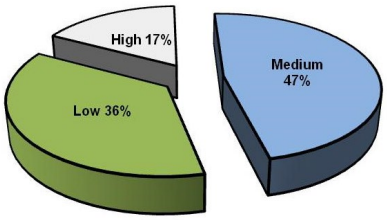

Fig. 3. Dataset after applying SMOTE.

4.3 Experimental evaluation

This section compares the performance of the produced models to conclude the model which better suits the addressed educational context. First, we applied decision tree classifiers, J48 and RepTree, over an imbalanced dataset of 50 students. These models are referred as J48 (original) and RepTree(original). In the second phase, we balanced the dataset using SMOTE and reapplied those same classifiers. This produces two more models referred as J48 (SMOTE) and RepTree (SMOTE). Table 6 provides the confusion matrices of the produced models. The confusion matrices can yield essential evaluation metrics. Table 7 provides the values to the metrics computed from the confusion matrix of each model.

Table 6. The confusion matrices of the classifiers

J48(original)				**Reptree (original)**			
High	***Medium***	***Low***	***Classified as***	***High***	***Medium***	***Low***	***Classified as***
9	2	0	High	5	6	0	High
2	28	1	Medium	3	27	1	Medium
0	2	6	Low	0	5	3	Low
J48(SMOTE)				**Reptree(SMOTE)**			
High	***Medium***	***Low***	***Classified as***	***High***	***Medium***	***Low***	***Classified as***
9	2	0	High	4	7	0	High
2	26	3	Medium	1	27	3	Medium
0	1	23	Low	0	0	24	Low

Accuracy. Figure 4 compares the accuracies of the developed models. Overall J48 is achieving the highest accuracy. However, a minor increase in the accuracy of J48 has appeared after sampling, whereas the accuracy of RepTree amplifies after sampling. Considering both the models of RepTree, table 7 demonstrates a rise in the sensitivity of "low" class (from 0.38 to 1.0) which boosts its accuracy. On the other hand, the sampling put a minor impact over J48 and a slight increase in the sensitivity of the "low" class is observed. This minor fluctuation in J48 and major inclination in RepTree is due to the added instances in the "low" class during sampling.

The accuracy does not appear useful due to its bias nature towards imbalanced datasets. This is visible in table 7, Reptree(original) achieves an accuracy of 70%, but the sensitivity for "low" class (0.38) is lower compared to other classes. However, sampling eradicates this behaviour. As a result, the J48 (SMOTE) model appears suitable when comparing the accuracy.

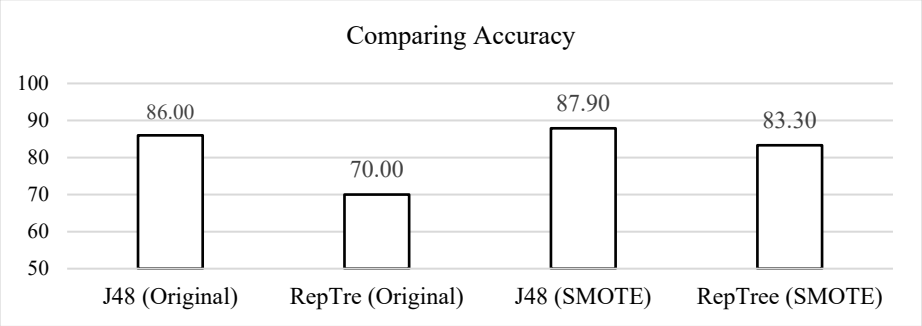

Fig. 4. A comparison of the accuracy metric of the classifiers

Sensitivity. Figure 5 compares the sensitivity of the models. It shows J48 produces a balanced sensitivity for all classes after the SMOTE. RepTree (with SMOTE) has perfect sensitivity (1.0) for "low" class, however, the sensitivity is very low (0.36) for "high" class. This behaviour makes it an unfavourable choice since it will not be able to identify the students with excellent performance as accurate as it will identify the minority class. Prior to SMOTE, RepTree has less than 0.50 sensitivity for "high" and "low" classes. Therefore, the evaluation based on the sensitivity also favours J48 (SMOTE) as a suitable choice, which shows the ability to identify the students in all classes.

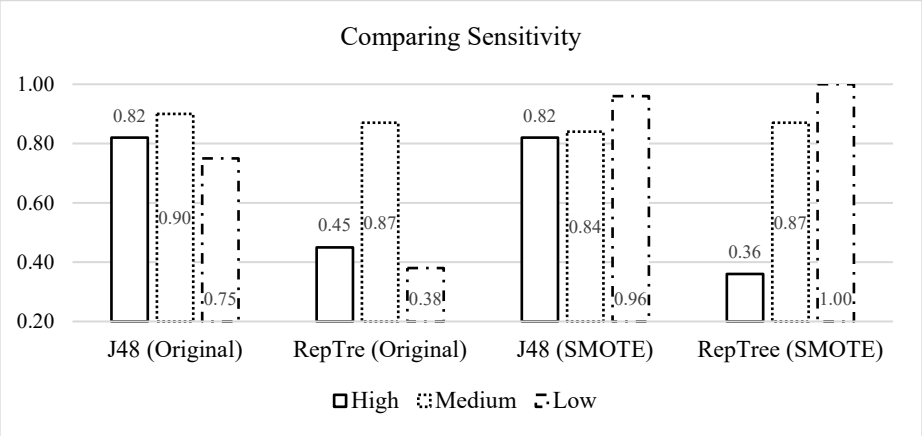

Fig. 5. Comparison of sensitivity for each prediction class of the classifiers

F-Measure. The F-measure metric further clarifies performance evaluation. Figure 6 shows the F-measure values of the prediction models. It illustrates an identical F-measure for most of the classes for J48 before and after sampling. However, this shows an increase in the F-measure values for all classes in RepTree after sampling. Comparing J48 and RepTree models after sampling illustrates J48 (SMOTE) showing a balanced F-measure for all the classes. In contrast RepTree(SMOTE) is showing very low F-measure value (0.5) for "high" class. This comparison concludes J48 as a better model than RepTree for both sampled and without sampled dataset.

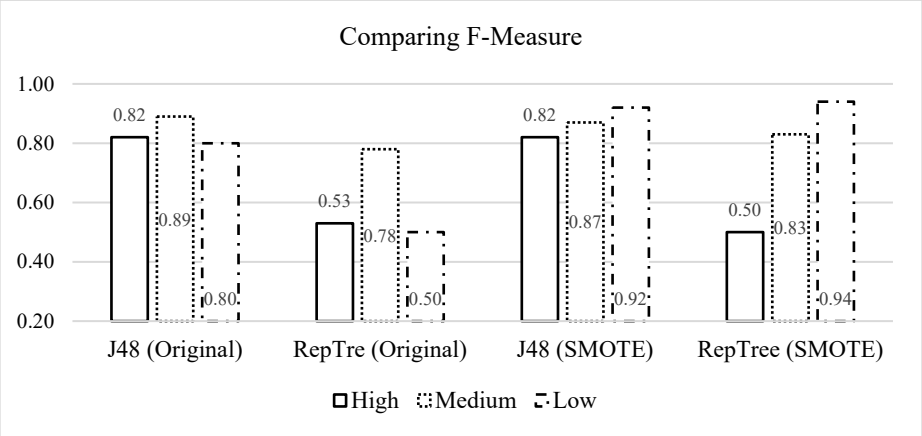

Fig. 6. Comparing the F-measures for each prediction class of the classifiers

Mathew Correlation Coefficient (MCC). MCC utilizes all of the four confusion matrix categories (TP, FN, FP, TN). Figure 7 provides a comparison of the MCC of the developed models. This comparison also suggests J48 (SMOTE) as the favourable model for the addressed context.

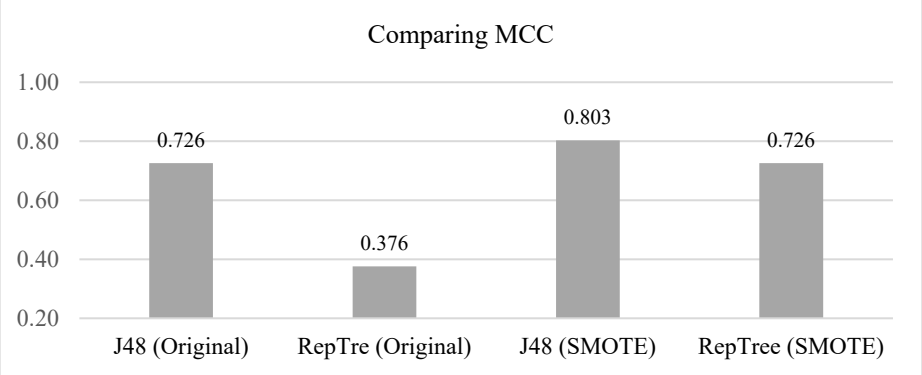

Fig. 7. MCC comparison of the prediction models

Table 7. The evaluation results of the classifiers

	TP	FN	FP	TN	Precision	Sensitivity	F-Measure	MCC	Accuracy
J48 (Original)									
High	9	2	2	37	0.82	0.82	0.82	0.767	86
Medium	28	3	4	15	0.88	0.90	0.89	0.7	
Low	6	2	1	41	0.86	0.75	0.80	0.767	
RepTree (Original)									
High	5	6	3	36	0.63	0.45	0.53	0.427	70
Medium	27	4	11	8	0.71	0.87	0.78	0.332	
Low	3	5	1	41	0.75	0.38	0.50	0.475	
J48 (SMOTE)									
High	9	2	2	53	0.8	0.82	0.82	0.782	87.88
Medium	26	5	3	32	0.9	0.84	0.87	0.757	
Low	23	1	3	39	0.9	0.96	0.92	0.873	
RepTree (SMOTE)									
High	4	7	1	54	0.80	0.36	0.50	0.487	83.33
Medium	27	4	7	28	0.79	0.87	0.83	0.67	
Low	24	0	3	39	0.89	1.00	0.94	0.909	

5 Model implementation

The previous subsection concludes J48 (SMOTE) as an appropriate model in the current context. Figure 8 shows the decision tree for the final prediction model (J48-SMOTE) as extracted from WEKA. Decision tree owns several conventional features making it a dominant choice for classification and prediction [13]. The key advantage of the decision tree is its ease in understanding and interpretation. Therefore, we interpret the produced output tree to prepare precautionary measures.

The prediction of the student is ineffective without placing precautionary measures. The aim of this paper is to implement the model within the programming course. In

light of this model, we can design precautionary measures for the students with inefficient programming skills. The tree shows that CGPA is at the root and splits at 1.73 to further branch the tree. Further, the Test1_Marks are split at 10. Accordingly, it is right to say that the CGPA below 1.73 could be labelled as "Low" and above 1.73 as "High". Similarly, Test1_Marks above 10 are labelled as "High" and below 10 as "Low". Student gender may play a vital role in student classification models; however, the students do not have control to modify this feature. Therefore, we exclude this feature from the proposed framework. We can interpret the decision tree in the form of a Consultation Framework as shown in figure 9.

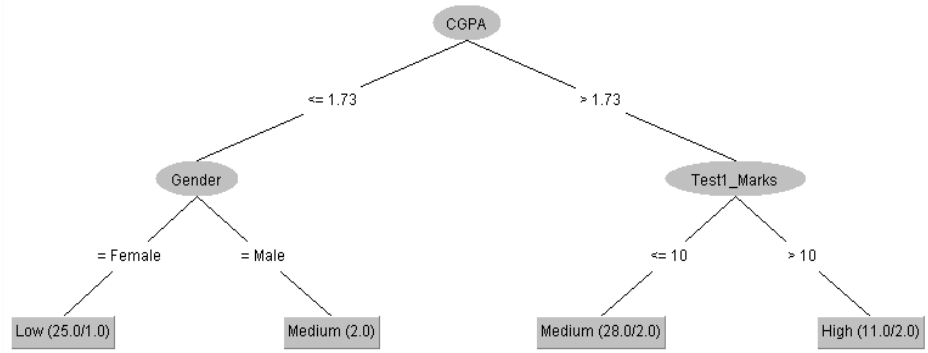

Fig. 8. The decision tree illustrations (J48 SMOTE), extracted from WEKA

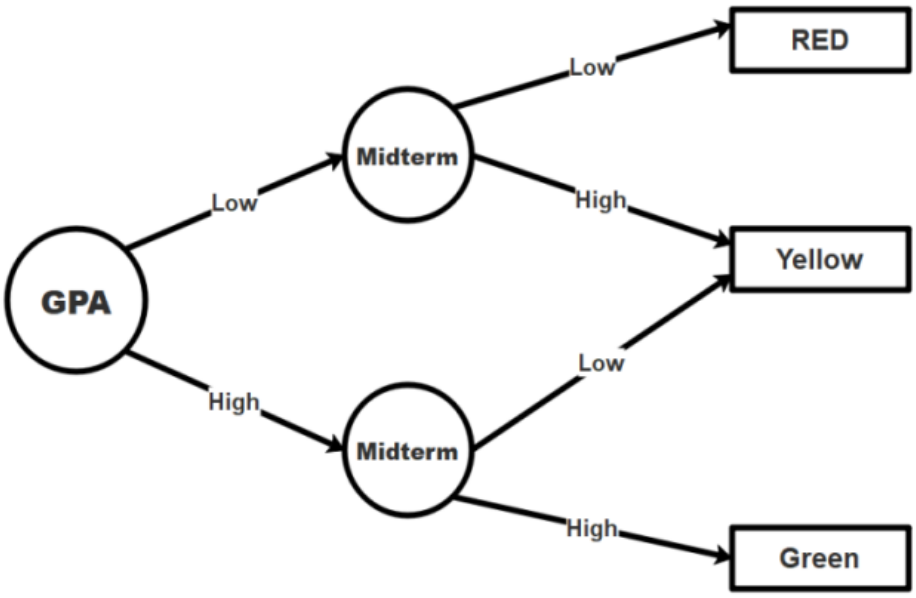

Fig. 9. The proposed prediction and consultation framework

Once, the midterm exam ends, the instructor assesses the student's performance from the perspective of the above framework. It will produce a coloured signal for each student similar to the traffic light signal, and the instructor prepares recommendations in accordance with each colour. The red signal indicates the student is in real danger of failing the course. These students will be forwarded for further advisory consultations. The instructor has to interview them and prepare precautionary procedures following the personalized context of each student. The instructor has to arrange extra classes to bring the students in the red zone on the proper track. The aim behind the extra class would be to revise the previous course contents and motivate the students for the remaining duration of the semester. The yellow signal, however, indicates the student with satisfactory performance in the course. However, proper consultation must be provided to these students to keep going on the correct academic track.

The Green signal indicates the student performing extraordinarily in the current programming course. Therefore, these students tend to perform far better and thus require special attention. The instructor can design problems with higher difficulty levels and motivate these students towards successful programmers.

6 Conclusions

This research aims to produce a prediction model that could correctly predict the performance of introductory programming students at an early stage of the semester. The data set is prepared with removing irrelevant and noisy features. The use of data mining techniques concludes a set of features significant for classification. Since the decision tree is one of the most widely used and easy to understand machine learning classifier, therefore, we produced models with J48 and RepTree decision tree classifiers with balanced as well as imbalanced dataset. The evaluation of the models shows that J48 decision tree model (with balanced dataset) is appropriate for the addressed context. The output of the J48 classifier is converted to an adaptive consultation framework that provides personalized actions for each class of students. The traffic signals indicators are used to notify the students regarding their current state and proper measures are suggested for each state.

7 References

[1] Alturki, R.A. (2016). Measuring and improving student performance in an introductory programming course. *Informatics in Education-An International Journal*, 2016. 15(2): p. 183-204. https://doi.org/10.15388/infedu.2016.10

[2] Kori, K.; Pedaste, M.; Tõnisson, E.; Palts, T.; Altin, H.; Rantsus, R.; Sell, R.; Murtazin, K., and Rüütmann, T. (2015). First-year dropout in ICT studies. *in 2015 IEEE Global Engineering Education Conference (EDUCON).* 2015. IEEE. https://doi.org/10.1109/educon.2015.7096008

[3] Alammary, A. (2019). Blended learning models for introductory programming courses: A systematic review. *PloS one*, 2019. 14(9): p. e0221765. https://doi.org/10.1371/journal.pone.0221765

[4] Zhang, X.; Zhang, C.; Stafford, T.F. and Zhang, P. (2019). Teaching introductory programming to IS students: The impact of teaching approaches on learning performance. *Journal of Information Systems Education*, 2019. 24(2): p. 6.
[5] Grover, S. and Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, 2013. 42(1): p. 38-43. https://doi.org/10.3102/0013189x12463051
[6] Munson, J.P. and Schilling, E.A. (2016). Analyzing novice programmers' response to compiler error messages. *Journal of Computing Sciences in Colleges*, 2016. 31(3): p. 53-61.
[7] Malik, S.I.; Mathew, R. and Hammood, M.M., *PROBSOL: A web-based application to develop problem-solving skills in introductory programming*, in *Smart Technologies and Innovation for a Sustainable Future*. 2019, Springer. p. 295-302. https://doi.org/10.1007/978-3-030-01659-3_34
[8] Watson, C.; Li, F.W. and Godwin, J.L. (2014). No tests required: comparing traditional and dynamic predictors of programming success. *in Proceedings of the 45th ACM technical symposium on Computer science education*. 2014. https://doi.org/10.1145/2538862.2538930
[9] White, G. and Sivitanides, M. (2003). An empirical investigation of the relationship between success in mathematics and visual programming courses. *Journal of information systems education*, 2003. 14(4): p. 409.
[10] Wilson, B.C. and Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. *Acm sigcse bulletin*, 2001. 33(1): p. 184-188. https://doi.org/10.1145/366413.364581
[11] Liao, S.N.; Zingaro, D.; Thai, K.; Alvarado, C.; Griswold, W.G., and Porter, L. (2019). A robust machine learning technique to predict low-performing students. *ACM Transactions on Computing Education (TOCE)*, 2019. 19(3): p. 1-19. https://doi.org/10.1145/3277569
[12] Quille, K. and Bergin, S. (2019). CS1: how will they do? How can we help? A decade of research and practice. *Computer Science Education*, 2019. 29(2-3): p. 254-282. https://doi.org/10.1080/08993408.2019.1612679
[13] Sunday, K.; Ocheja, P.; Hussain, S.; Oyelere, S.; Samson, B., and Agbo, F. (2020). Analyzing Student Performance in Programming Education Using Classification Techniques. *International Journal of Emerging Technologies in Learning (iJET)*, 2020. 15(2): p. 127-144. https://doi.org/10.3991/ijet.v15i02.11527
[14] Figueiredo, J.; Lopes, N. and García-Peñalvo, F.J. (2019). Predicting Student Failure in an Introductory Programming Course with Multiple Back-Propagation. *in Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality*. 2019. https://doi.org/10.1145/3362789.3362925
[15] Viberg, O.; Hatakka, M.; Bälter, O. and Mavroudi, A. (2018). The current landscape of learning analytics in higher education. *Computers in Human Behavior*, 2018. 89: p. 98-110. https://doi.org/10.1016/j.chb.2018.07.027
[16] Vahdat, M.; Ghio, A.; Oneto, L.; Anguita, D.; Funk, M., and Rauterberg, M. (2015). Advances in learning analytics and educational data mining. *Proc. of ESANN2015*, 2015: p. 297-306.
[17] Alabri, A.; Al-Khanjari, Z.; Jamoussi, Y. and Kraiem, N. (2019). Mining the Students' Chat Conversations in a Personalized e-Learning Environment. *International Journal of Emerging Technologies in Learning (iJET)*, 2019. 14(23): p. 98-124. https://doi.org/10.3991/ijet.v14i23.11031
[18] Alfere, S.S. and Maghari, A.Y. (2018). Prediction of Student's Performance Using Modified KNN Classifiers. *Prediction of Student's Performance Using Modified KNN Classifiers*, 2018.

[19] Mondal, A. and Mukherjee, J. (2018). An Approach to predict a student's academic performance using Recurrent Neural Network (RNN). *Int. J. Comput. Appl*, 2018. 181(6): p. 1-5. https://doi.org/10.5120/ijca2018917352

[20] Lagman, A.C.; Calleja, J.Q.; Fernando, C.G.; Gonzales, J.G.; Legaspi, J.B.; Ortega, J.H.J.C.; Ramos, R.F.; Solomo, M.V.S., and Santos, R.C. (2019). Embedding naïve Bayes algorithm data model in predicting student graduation. *in Proceedings of the 3rd International Conference on Telecommunications and Communication Engineering.* 2019. https://doi.org/10.1145/3369555.3369570

[21] Ahadi, A.; Lister, R.; Haapala, H. and Vihavainen, A. (2015). Exploring machine learning methods to automatically identify students in need of assistance. *in Proceedings of the eleventh annual International Conference on International Computing Education Research.* 2015. ACM. https://doi.org/10.1145/2787622.2787717

[22] Chen, H., *Predicting student performance using data from an Auto-grading system*. 2018, University of Waterloo.

[23] Hamsa, H.; Indiradevi, S. and Kizhakkethottam, J.J. (2016). Student academic performance prediction model using decision tree and fuzzy genetic algorithm. *Procedia Technology*, 2016. 25: p. 326-332. https://doi.org/10.1016/j.protcy.2016.08.114

[24] Pandey, M. and Taruna, S. (2016). Towards the integration of multiple classifier pertaining to the Student's performance prediction. *Perspectives in Science*, 2016. 8: p. 364-366. https://doi.org/10.1016/j.pisc.2016.04.076

[25] Kausar, S.; Oyelere, S.; Salal, Y.; Hussain, S.; Cifci, M.; Hilcenko, S.; Iqbal, M.; Wenhao, Z., and Huahu, X. (2020). Mining Smart Learning Analytics Data Using Ensemble Classifiers. *International Journal of Emerging Technologies in Learning (iJET)*, 2020. 15(12): p. 81-102. https://doi.org/10.3991/ijet.v15i12.13455

[26] Saa, A.A. (2016). Educational data mining & students' performance prediction. *International Journal of Advanced Computer Science and Applications*, 2016. **7**(5): p. 212-220.

[27] Kiu, C.-C. (2018). Data Mining Analysis on Student's Academic Performance through Exploration of Student's Background and Social Activities. *in 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA).* 2018. IEEE. https://doi.org/10.1109/icaccaf.2018.8776809

[28] Kaunang, F.J. and Rotikan, R. (2018). Students' Academic Performance Prediction using Data Mining. *in 2018 Third International Conference on Informatics and Computing (ICIC).* 2018. IEEE. https://doi.org/10.1109/iac.2018.8780547

[29] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P., and Witten, I.H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 2009. 11(1): p. 10-18. https://doi.org/10.1145/1656274.1656278

[30] Asif, R.; Merceron, A.; Ali, S.A. and Haider, N.G. (2017). Analyzing undergraduate students' performance using educational data mining. *Computers & Education*, 2017. 113: p. 177-194. https://doi.org/10.1016/j.compedu.2017.05.007

[31] Orong, M.Y.; Caroro, R.A.; Durias, G.D.; Cabrera, J.A.; Lonzon, H., and Ricalde, G.T. (2020). A predictive analytics approach in determining the predictors of student attrition in the higher education institutions in the Philippines. *in Proceedings of the 3rd International Conference on Software Engineering and Information Management.* 2020. https://doi.org/10.1145/3378936.3378956

[32] Abdulazeez, Y. and Abdulwahab, L. (2018). Application of classification models to predict students' academic performance using classifiers ensemble and synthetic minority over sampling techniques. *Bayero Journal of Pure and Applied Sciences*, 2018. 11(2): p. 142-148. https://doi.org/10.4314/bajopas.v11i2.17

[33] Hussain, M.; Zhu, W.; Zhang, W. and Abidi, S.M.R. (2018). Student engagement predictions in an e-learning system and their impact on student course assessment scores. *Computational intelligence and neuroscience*, 2018. https://doi.org/10.4314/bajopas.v11i2.17

[34] Afeni, B.O.; Oloyede, I.A. and Okurinboye, D. (2019). Students' Performance Prediction Using Classsification Algorithms. *Journal of Advances in Mathematics and Computer Science*, 2019: p. 1-9. https://doi.org/10.9734/jamcs/2019/45438

[35] Matthews, B.W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 1975. **405**(2): p. 442-451. https://doi.org/10.1016/0005-2795(75)90109-9

[36] Chawla, N.V.; Bowyer, K.W.; Hall, L.O. and Kegelmeyer, W.P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 2002. **16**: p. 321-357. https://doi.org/10.1613/jair.953

8 Authors

Ijaz Muhammad Khan is a PhD student at the College of Graduate Studies Universiti Tenaga Nasional Kajang, Malaysia. He received his MPhil degree in Computer Science from Liverpool John Moores University in 2009. Currently, he has been working as a lecturer at Al-Buraimi University College, Oman for the last 8 years. His research interests include Artificial Intelligence, Machine learning, internet of things and social networking.

Dr. Abdul Rahim Ahmad is Associate Professor at College of Computing and Informatics, Universiti Tenaga Nasional, Kajang, Malaysia. His early career was as a lecturer in the Malaysian Polytechnic system. He later joined the satellite television company Measat Broadcast Network Systems (MBNS) broadcast automation team. In 1997, he joined Universiti Tenaga Nasional (UNITEN) as Senior Lecturer. He held various positions in the College of Computing and Informatics (CCI), Research Management Centre (RMC) and the Institute of Energy Policy and Research (IEPRe). He is currently an Associate Professor in CCI. He obtained his bachelor degree in Computer Science from University of Queensland in Australia, masters from Loughborough University, UK and Phds from Universiti Tteknologi Malaysia and University of Nantes, France. His academic and research are in Computer Systems, Networks and Artificial Intelligence. He was active as a committee member of IEEE Malaysia and IEEE Computer Society Malaysia.

Dr. Nafaâ Jabeur is Associate Professor and Director of Research at the GermanUniversity of Technology in Oman (GUtech). He received his PhD and MSc degrees in Computer Science from Laval University, Quebec, Canada in 2006 and 2001, respectively. He has over than fifteen years of experience in the industrial and academic sectors. Nafaâ has participated in several R&D projects, edited2 books, and authored more than 80 research papers in prestigious conferences and high ranked journals.His main research interests include Smart cities, Transportation, Blockchain, Drones, and Artificial Intelligence.

Mohammed Najah Mahdi received his BSc degree in Information Engineering, College of Engineering, Baghdad University, Iraq, in 2002 and MSc in Information

Technology from Faculty of Computer Science and Information Technology, University of Malaya (UM) in 2011. He later obtained his Ph.D. in Information and Communication Technology, in 2017. He is currently a Post-Doctoral Researcher at the University Tenaga Nasional (UNITEN) Malaysia. His research interests include faceted search, information overload, exploratory search and information retrieval.

Article submitted 2020-10-02. Resubmitted 2020-12-12. Final acceptance 2021-01-04. Final version published as submitted by the authors.