

# Handling class imbalance in education using data-level and deep learning methods

Rithesh Kannan<sup>1</sup>, Hu Ng<sup>1</sup>, Timothy Tzen Vun Yap<sup>2</sup>, Lai Kuan Wong<sup>1</sup>, Fang Fang Chua<sup>1</sup>,  
Vik Tor Goh<sup>3</sup>, Yee Lien Lee<sup>3</sup>, Hwee Ling Wong<sup>3</sup>

<sup>1</sup>Faculty of Computing and Informatics, Multimedia University, Cyberjaya, Malaysia

<sup>2</sup>School of Mathematical and Computer Sciences, Heriot-Watt University Malaysia, Putrajaya, Malaysia

<sup>3</sup>Faculty of Engineering, Multimedia University, Cyberjaya, Malaysia

## Article Info

### Article history:

Received Mar 28, 2024

Revised Aug 13, 2024

Accepted Oct 1, 2024

### Keywords:

Academic at-risk

Class balancing

Educational data mining

Multi-classification

Resampling techniques

Synthetic datasets

## ABSTRACT

In the current field of education, universities must be highly competitive to thrive and grow. Education data mining has helped universities in bringing in new students and retaining old ones. However, there is a major issue in this task, which is the class imbalance between the successful students and at-risk students that causes inaccurate predictions. To address this issue, 12 methods from data-level sampling techniques and 2 methods from deep learning synthesizers were compared against each other and an ideal class balancing method for the dataset was identified. The evaluation was done using the light gradient boosting machine ensemble model, and the metrics included receiver operating characteristic curve, precision, recall and F1 score. The two best methods were Tomek links and neighbourhood cleaning rule from undersampling technique with a F1 score of 0.72 and 0.71 respectively. The results of this paper identified the best class balancing method between the two approaches and identified the limitations of the deep learning approach.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



## Corresponding Author:

Hu Ng

Faculty of Computing and Informatics, Multimedia University

Persiaran Multimedia, 63100 Cyberjaya, Selangor, Malaysia

Email: nghu@mmu.edu.my

## 1. INTRODUCTION

In present times, education institutes operate in a highly complex environment and are fiercely competitive with each other. In order to stand out, the institute must be able to consistently attract new students and retain old ones. With the rapid rise in technology development allowing equipment to become cheaper, more data on students can be collected and stored. By effectively analyzing and utilizing this data, an educational institute can leverage its advantages and outperform its competitors. Educational data mining (EDM) is the emerging field that is concerned with automating the process of analyzing student data and providing actionable insights for members of the staff to utilize.

Predicting student academic performance is a common goal in the field of EDM. This is an important task as a major contribution to a student's success is their academic performance in higher education institutes. Academic performance may consist of examination results, coursework, co-curricular achievement, whether the student has graduated on time, and so on. [1] used data mining to predict students' grades using their attendance, class test, assignment and midterm scores and achieved a high accuracy of 88.6% using deep learning algorithms. Students with good academic results are likely to face a greater amount of career choices and better job security, when compared to students with poor academic results. However, it must be noted that each student is unique, as they receive and process information differently

from one another. It is possible for students with poor academic results to improve with some interventions. Thus, it is a critical task to know the at-risk students as early as possible in the study year so that interventions can be applied quickly.

One major challenge faced in this task is the class imbalance between the successful students and the at-risk students. Generally, there are a lot more successful students compared to at-risk students in a program. Building a predictive model without addressing this issue can lead to meaningless results, as the model may show that it has a higher accuracy by simply predicting all students passed and are successful [2], [3]. Other researchers like Chawla *et al.* [4] have commonly used traditional oversampling techniques like synthetic minority oversampling technique (SMOTE) or undersampling techniques like Tomek links (TL) [5]. However, these techniques are not always applicable and have disadvantages, such as oversampling introducing additional noise and undersampling removing large samples of data to make the majority class samples equal to the minority class. The class imbalance issue is also not unique to the education domain, it can occur in many different domains including cybersecurity [6], air quality [7], and others.

In recent years, researchers have explored generating synthetic datasets to increase the amount of information and help improve the performance of the predictive models [8]. Moreover, through various deep learning synthesizers, it has been shown to be possible to generate synthetic datasets to solve the class imbalance issue [9]. The synthetic datasets are modelled on a similar data distribution to the real dataset by the synthesizers but can modify the target classes by making the classes balanced appropriately. However, as this is still a relatively new approach, there has not been much focus on this approach.

In this paper, multiple methods from the two approaches to class balancing, which are data-level sampling and deep learning synthesizers are compared and evaluated on a student education dataset. The multiclass evaluation is performed through the light gradient boosting machine (LightGBM) classifier using machine learning metrics such as receiver operating characteristic (ROC), precision, recall and F1 score. In addition, the datasets generated are compared visually and through data quality scores.

Data-level techniques are a group of techniques that involve modifying the actual dataset itself to make the class distribution balanced [10], [11]. There are three types under this approach, oversampling, undersampling and hybrid sampling. Oversampling techniques are generally concerned with increasing the minority class samples to make the class distribution like the majority class. This is done by sampling the majority class and replicating it or synthesizing new samples based on it for the minority class. The main disadvantage of this method is that it relies heavily on the original data quality, and generating too many samples may lead to overfitting. The four methods of this nature considered in this paper are random oversampling (ROS) [12], synthetic minority oversampling technique (SMOTE) [4], borderline synthetic minority oversampling technique (BSMOTE) [13] and adaptive synthetic technique (ADASYN) [14].

Undersampling techniques on the other hand, are concerned with reducing the majority class samples to make the class distribution equal to the minority class. This is done by randomly or strategically removing samples from the majority class. The main disadvantage of this method is the loss of data which could have been used to train the model. The four methods of this nature considered in this paper are random undersampling (RUS), TL [5], edited nearest-neighbor (ENN) [15] and neighbourhood cleaning rule (NCR) [16]. Finally, hybrid sampling techniques combine both oversampling and undersampling methods together to balance the minority and majority class distribution. Generally, they perform better than both oversampling and undersampling as they combine the advantages and limit the disadvantages of both techniques. The four methods of this nature considered in this paper are SMOTE-TL [17], SMOTE-ENN [17], SMOTE-RUS and SMOTE-NCR [18].

Pratama *et al.* [19] conducted several experiments to show the effect of data-level techniques on imbalanced classification and compared several resampling methods including SMOTE, BSMOTE and SMOTE-TL to find the best method. They also utilized several machine learning classifiers, like logistic regression (LR), k-nearest neighbors (K-NN), classification and regression trees (CART), random forest (RF), support vector machine (SVM), and the stacking ensemble method. The results showed that the hybrid sampling method, SMOTE-TL worked the best with the RF model, achieving 85.8% accuracy on a 10-fold cross-validation and a score of 0.89 Geometric mean, which was the best score among the models. The study could be improved by adding more undersampling and hybrid-sampling methods or focusing deeper on ensemble learning classifiers. More types of data could have also been added that have been adjusted with feature selection methods.

On the other hand, Buraimoh *et al.* [20] focused on the importance of dimensionality reduction and data-sampling methods to improve the performance of imbalanced models predicting student success. They utilized principal component analysis (PCA) for their dimensionality reduction along with comparing ROS, RUS, and SMOTE methods with six classifiers. They were SVM, K-NN, CART, gradient boosted tree (GBT), multilayer perceptron (MLP), and linear discriminant analysis (LDA). The results showed that

SMOTE with PCA was the best preprocessing method along with the SVM classifier as they achieved the highest accuracy of 0.93 and a Kappa accuracy of 0.89.

The other approach to class balancing is through using deep learning synthesizers to generate synthetic data that balance the target classes [21]. The two synthesizers used in this paper are Gaussian copula and generative adversarial network (GAN). Gaussian copulas are mathematical models used to map the marginal distribution of each variable in the given dataset to standard normal distribution. Essentially, they create a joint probability for two or more variables while still preserving their marginal distributions. The disadvantage of Gaussian copula is that it cannot capture tail dependence within the data distribution. Thus, it cannot replicate the real data distribution completely. GANs are a class of deep learning models that work by training two neural networks, called the generator and the discriminator. The generator network goal is to try and produce new synthetic data comparable to the real data and the discriminator's goal is to evaluate whether the input data is real or synthetic. GANs are typically used in computer vision to arbitrarily generate images to do data augmentation and increase the size of their datasets for a better prediction. However, researchers have also adapted them into learning from real tabular data and generating synthetic data with high fidelity [22], [23].

In terms of deep learning approaches, most use models to generate synthetic data that can balance the classes. However, the most popular use case for these methods are image data and video data. There were not many well performing models for text data until a researcher introduced one and improved upon previous methods by creating their model called conditional tabular generative adversarial network (CTGAN) [22]. They further ensured that the model could generate samples in a novel random manner to address the class imbalance issue. The CTGAN model is being continuously improved upon by other researchers, however the base model is still relatively suitable for most tasks. Researchers have also experimented with improving CTGAN models to create a generative adversarial network modelling inspired from naive Bayes and logistic regression's relationship (GANBLR) [24].

Besides GAN, another deep learning (DL) based popular model is variational autoencoder (VAE), which [25] proposed. It was an integrated framework that consisted of latent VAE with a deep neural network (DNN) to address and alleviate the class imbalance issue and provide early warning for at-risk students. VAE has certain advantages as it has a simpler loss function compared to GAN for example. The researchers utilized this to train multiple models quickly and got a high result of 80% F1 score.

## 2. METHOD

In this section, the overall methodology of this paper as shown in Figure 1, is explained. There are two separate research flows in the overall methodology, with the main difference being the stage at which the class balancing occurs. For data-level resampling, the class balancing occurs immediately after data splitting on the train set whereas for the deep learning method, the class balancing occurs after feature selection when the synthetic datasets are generated.

### 2.1. Dataset

The student academic dataset is collected from graduated students in the years of 2020 to 2021, across different programs from a private university in Malaysia. There are a total of 5,488 students and 158 features, including demographic and academic performance features. Among the 158 features, 23 are categorical, 132 are numerical and 3 are datetime features. Each semester has their own grade point average (GPA) and cumulative grade point average (CPGA) and are grouped together in Table 1.

### 2.2. Data pre-processing

The main step in preprocessing the data is handling missing data and removing any irrelevant data. In this paper, features with missing data above 80%, are dropped. Some of the columns have overlapping information such as *ACAD\_PLAN*, *TRANSCR\_DECR*, and *ACAD\_PROG\_DESCR*. The feature that provides the most information is kept and the remaining columns are dropped. All remaining missing values are replaced with median for quantitative features and mode for qualitative features.

### 2.3. Data splitting

The dataset is then divided into a train and test set with 70%-30% ratio respectively. The data-level class balancing methods are applied only to the train set as the test set must remain unchanged to obtain an unbiased estimate of the performance. Subsequent steps besides the data-level methods are applied to both sets of data.

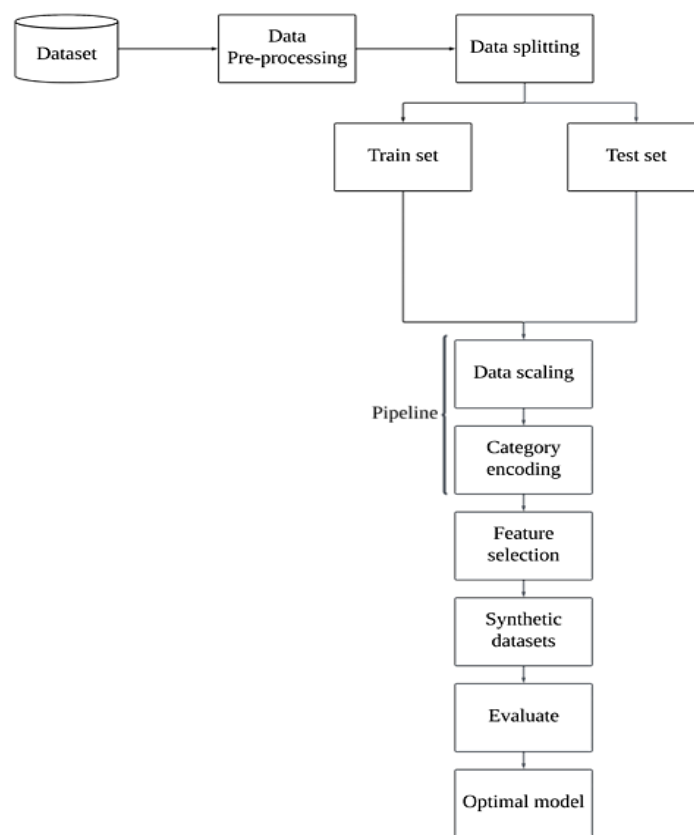
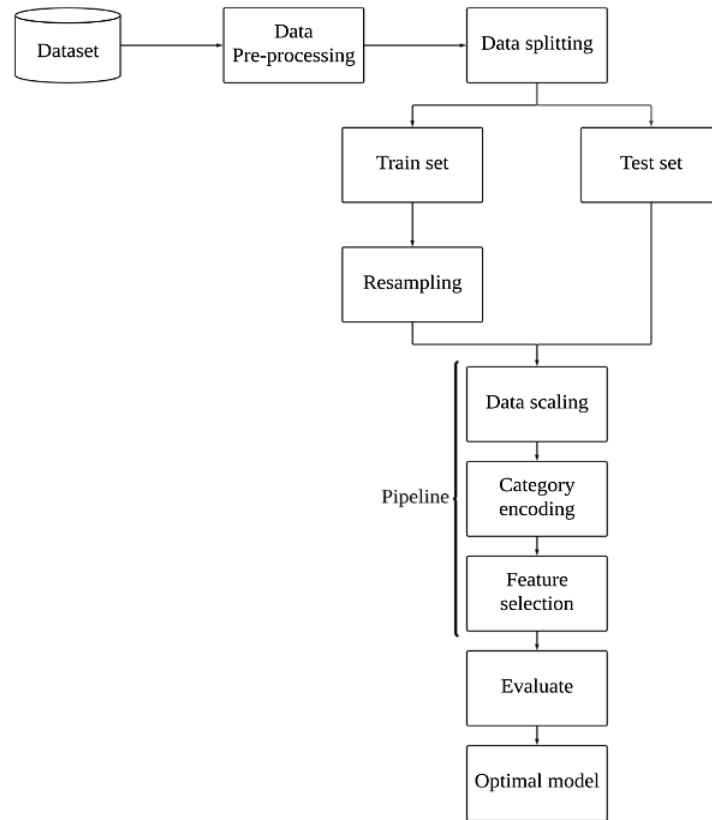


Figure 1. Research flows for (a) data-level resampling techniques and (b) deep learning synthetic models

Table 1. Features descriptions in the academic dataset

Name	Description
<i>GOT2</i>	Graduated on time or not
<i>NewID</i>	Unique identifier (ID) for students
<i>ACAD_CAREER</i>	Type of degree the student is taking
<i>PROG_STATUS</i>	End of program status of student
<i>PROG_ACTION</i>	End of program action for student
<i>ADMIT_TERM</i>	The term when the student is admitted
<i>BEGIN_DT</i>	Date when student is admitted
<i>END_DT</i>	Date when student is graduated
<i>STATUS_DT</i>	Date when student status is changed
<i>EXP_GRAD_TERM</i>	Expected term the student graduates
<i>CAMPUS</i>	Campus the student belongs to
<i>SAD_LOAD_DESCR</i>	Mode of study the student is in
<i>ACAD_PLAN</i>	Code for the specific study program the student is taking
<i>ACAD_PROG_DESCR</i>	Short form of the study program
<i>TRNSCR_DESCR</i>	Long, descriptive form of the study program
<i>ACAD_ORG</i>	Faculty the student belongs to
<i>DISABILITY</i>	Type of disability the student has
<i>NATIONALITY</i>	Nationality of the student
<i>RACE</i>	Race of the student
<i>SEX</i>	Sex of the student
<i>MUET</i>	Malaysian University English test (MUET) score for the student
<i>IELTS</i>	International English language testing system (IELTS) score for the student
<i>LOAN</i>	Loan belonging to the student
<i>SPONSOR</i>	Sponsorship belonging to the student
<i>SCHOLAR</i>	Scholarship belonging to the student
<i>TOT_CUMULATIVE</i>	Total cumulative credits
<i>N_FINAL_RSLT_DESCR</i>	Description of final program results
<i>N_HONOUR_DESCR</i>	Honors belonging to the student
<i>CREDITREQUIRED</i>	Total credits required to graduate
<i>INFO1</i>	Sijil Pelajaran Malaysia (SPM) and Sijil Tinggi Persekolahan Malaysia (STPM) grades of student
<i>Prog_Length (Term)</i>	Number of terms student has done
<i>T1 to T17: CUR_GPA</i>	Current GPA for the term
<i>T1 to T17: CUM_GPA</i>	Cumulative GPA (CPGA) for the term

## 2.4. Data scaling

For all quantitative features, the data is scaled to ensure no feature dominates the calculation in a predictive algorithm. For this paper, a robust scaler is applied as it scales the data to the interquartile range which allows the data to be robust to outliers. It is also suitable to handle skewed distributions as it is based on percentiles which are less affected by extreme values.

## 2.5. Category encoding

For this paper, a label encoder has been used to convert the categorical target feature into numerical. Ordinal encoder has been used to convert those ordinal categorical variables and those variables with only two unique values. For the remaining nominal variables, it is decided to use M-estimate encoder, as seen in [26]. The M-estimate or M-probability estimate encoder is generally used when the cardinality of features is high. It uses the target variable to encode the nominal features and uses a regularization variable to control the target leakage and reduce the overfitting. OneHotEncoder cannot be used as it causes the features to greatly increase in number. Examples for the mapping have been provided in Table 2. As there is a certain amount of randomness present in the M-estimate encoder, the mapping for the encoder cannot be clearly seen, which is a disadvantage of this method.

Table 2. Before and after category encoding

Name	Before encoding	After encoding
Label encoding	{NO GPA, PASS, PROBATION, TERMINATED, TERMINATED – REINSTATED}	{0, 1, 2, 3, 4}
Ordinal encoding	{N, Y}	{0, 1}

## 2.6. Feature selection

In this paper, recursive feature elimination (RFE), utilizing the LightGBM model is used to select the most important features from the dataset. RFE is widely used as one of the best feature selection methods and works by searching for the best subset of features by starting with all features in the training dataset and successfully removing the features until the desired number is reached. The reason why RFE was chosen is because it starts with all features and removes the less relevant ones, which helps conserve the most amount

of data. The LightGBM model was selected because it provides generally good performance and is memory efficient [27], which helps when multiple models are being run in RFE. They identified 43 features as providing the best performance in terms of F1 score. Beyond 43 features, the performance does not increase substantially, as can be seen in Figure 2.

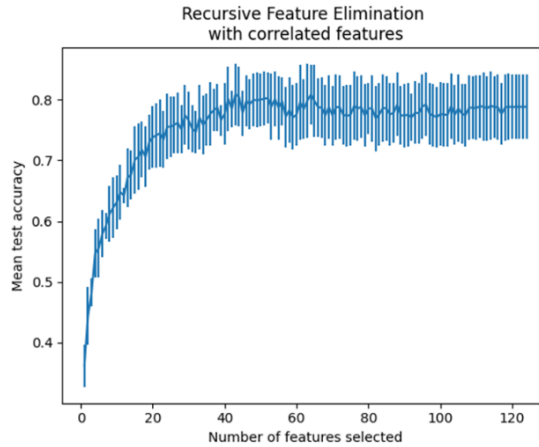


Figure 2. Comparing mean test accuracy of models against number of selected features in the models

**2.7. Class balancing**

When the chosen target feature, “*N\_FINAL\_RSLT\_DESCR*” otherwise known as the final status of the program is examined, a clear class imbalance can be seen. There is a significant amount of class imbalance present in this dataset as can be seen in Table 3 and Figure 3. The majority class, “*PASS*” occupies around 66% of all values whereas the minority class, “*TERMINATED – REINSTATED*” does not occupy even 1%.

Table 3. Distribution of classes in target feature

Original variable	Encoded variable	Count	Percentage (%)
NO GPA	0	929	17.33
PASS	1	3576	66.72
PROBATION	2	551	10.28
TERMINATED	3	273	5.09
TERMINATED – REINSTATED	4	31	0.58

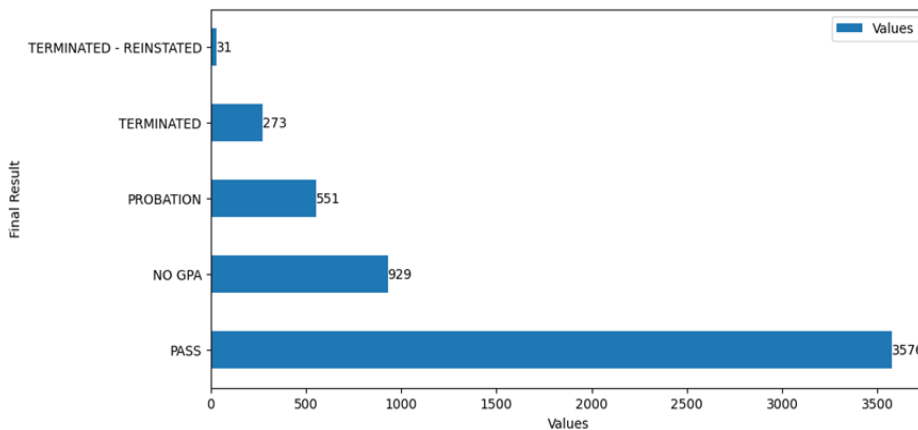


Figure 3. Visualization of class imbalance in target feature

The amount of class imbalance can be calculated using the imbalance ratio (IR) formula which is given in (1). For multiclass classification, the IR are samples from the greatest majority class over the lowest

minority class, which in this case is  $3576/31=115.35$ . When the dataset is split into train and test dataset, stratified splitting is used to ensure the same class distribution is present in both datasets. In (1),  $N_{maj}$  refers to the number of samples in the majority class and  $N_{min}$  refers to the number of samples in the minority class.

$$IR = \frac{N_{maj}}{N_{min}} \quad (1)$$

## 2.8. Performance metrics

To evaluate the models and choose the optimal one, certain metrics need to be chosen to compare the results. In this paper, common metrics from the machine learning field that are suited for imbalanced multiclass classification problems, have been chosen. These include threshold metrics such as precision, recall and F1 score. Precision score is a metric that calculates the ratio of true positive predicted to the amount of total positive included in the model. Recall score also known as true positive rate refers to the ratio of true positive predicted to the number of actual positive included in the model. F1 score can show the true model performance as it is calculated using harmonic means of both precision and recall for each class. This score shows poor results if the model is simply predicting the majority class. As this is a multiclass classification problem, the average for each metric is calculated using macro-average as it treats each class equally and if one class performs poorly the overall result becomes poor. This is useful for imbalanced datasets as it ensures that each class equally contributes to the result. Besides these, the ROC area under the curve (AUC) score is also included as it is not biased towards the majority or minority classes, which makes it useful in imbalanced classification. It calculates the trade-off between the true positive rate and false positive rates for a model. Again, as this is a multiclass classification, the ROC AUC score uses the one vs rest scheme which compares each class against all others together and averages the results.

## 2.9. Application of class balancing methods

In this Subsection, each of the class balancing methods are briefly explained, and their effect of the method on the dataset is shown in Tables 4 to Table 7. The different ways each method approaches class balancing and to what extent they consider classes as balanced can be seen as well. Each class balancing method has then been evaluated using a LightGBM classifier in section 3.

### 2.9.1. Oversampling techniques

Within the oversampling techniques, the simplest is ROS, where random samples among all classes besides the majority class are duplicated until the final amount from each class is equal to the majority class. In this paper, among the 5 target classes in the real dataset, '0' has 60, '1' has 2,503, '2' has 386, '3' has 191 and '4' has 22, ROS causes all classes to have 2,503 samples, as can be seen in Table 4. SMOTE is more complex than ROS as it generates synthetic samples of the minority class by considering the linear combinations of existing minority class neighbors. By default, it generates samples equal in number to the majority class. Thus, in this paper, samples from all other classes increase to 2,503.

BSMOTE improves upon the SMOTE algorithm by selecting the minority class samples on the border of the line of best fit. These are then used to synthesize new samples which helps improve the sample category distribution. For this paper, the number of samples from each class increases to 2,503. The same as the majority class samples. ADASYN is also an oversampling method that generates synthetic samples, but the difference between it and SMOTE is that ADASYN focuses on minority samples that are difficult to classify correctly, rather than oversampling all minority samples uniformly. It assigns weight to each minority instance based on its difficulty. For this paper, '0' increased from 650 to 2,500 samples, '1' remained the same at 2,503 samples, '2' increased from 386 to 2,488 samples, '3' increased from 191 to 2,511 samples and '4' increased from 22 to 2,507 samples, as seen in Table 4.

Table 4. Distribution of class after oversampling

Encoded variable	Imbalanced	ROS	SMOTE	BSMOTE	ADASYN
0	650	2503	2503	2503	2500
1	2503	2503	2503	2503	2503
2	386	2503	2503	2503	2488
3	191	2503	2503	2503	2511
4	22	2503	2503	2503	2507

### 2.9.2. Undersampling techniques

For undersampling techniques, like ROS, RUS is the simplest technique that randomly removes samples from all other classes besides the minority class until the number of samples is equal to the minority

class samples. For this paper, as the minority class is '4' with 22 samples, samples from all other classes are removed until there is 22 for each class. TL is an undersampling technique focused on removing those noisy instances that are on the border of the classification line. This is done by finding pairs of very close instances that belong to different classes. These pairs are known as Tomek links and removing the majority class instances of each pair increases the space between the classes which improves the classification process as the noisy and borderline instances are removed. In this paper, the '0' class decreased from 650 to 605 samples, '1' decreased from 2,503 to 2,449, '2' decreased from 386 to 355, '3' decreased from 191 to 180 and '4' remained the same at 22 samples, as seen in Table 5.

ENN is another method for eliminating noisy samples from classes other than the minority class. For every sample, ENN calculates its nearest neighbours 'by default, three', if the sample was not from the minority class and is misclassified by its neighbours, it is removed. If the sample was from a minority class however, and is misclassified by its neighbours, then the neighbour instances not from the minority class are removed. For this paper, '0' decreased from 650 to 396 samples, '1' decreased from 2,503 to 2,105, '2' decreased from 386 to 173, '3' decreased from 191 to 94 and '4' remained the same at 22 samples. Meanwhile, NCR is an improvement to ENN as it focuses less on improving the class distribution and more on increasing the unambiguity of the samples that are retained in all the classes except the minority class. NCR works by first selecting and removing all noisy samples in a similar manner to ENN. Then the number of samples for all classes except the minority class that are misclassified are removed, but only if the number of samples in those classes is larger than half the samples in the minority class. The NCR method changed the class distribution in the following manner: '0' decreased from 650 to 493 samples, '1' decreased from 2,503 to 2,357, '2' decreased from 386 to 247, '3' decreased from 191 to 142 and '4' remained the same at 22 samples.

Table 5. Distribution of class after undersampling

Encoded variable	Imbalanced	RUS	TL	ENN	NCR
0	650	22	605	396	493
1	2503	22	2449	2105	2357
2	386	22	355	173	247
3	191	22	180	94	142
4	22	22	22	22	22

### 2.9.3. Hybrid sampling techniques

Hybrid sampling mostly consists of combining both oversampling and undersampling techniques together. Since SMOTE is so popular [28], it is widely used in combination with other undersampling methods. SMOTE-RUS is one such method, where SMOTE and RUS are combined by first generating synthetic samples for the minority class and then eliminating samples from classes other than the minority class. According to [4], who first implemented SMOTE, the best performing method was when SMOTE was combined with RUS and not just when using SMOTE alone. For this paper, samples from all classes were equal to the majority class after SMOTE-RUS was applied. Like SMOTE-RUS, SMOTE-TL first performs oversampling on the minority class using SMOTE and then undersampling using Tomek Links. However, there are some slight differences in the class distribution after the method was applied when compared to SMOTE-RUS, due to RUS being a more simple and naïve method than TL. The class distribution was changed as follows: '0' increased from 650 to 2,501, '1' decreased from 2,503 to 2,501, '2' increased from 386 to 2,503, '3' increased from 191 to 2,503 and '4' increased from 22 to 2,503 samples.

A similar pattern to SMOTE-TL occurs when SMOTE-ENN is applied to the dataset, albeit with more differences in class distribution when compared to SMOTE-RUS. This is observed in Table 6 as class '0' samples increased from 650 to 2,405, '1' decreased from 2,503 to 1,990, '2' increased from 386 to 2,476, '3' increased from 191 to 2,500 and '4' increased from 22 to 2,503 samples. Whereas after SMOTE-NCR was applied '0' increased from 650 to 2,503 samples, '1' decreased from 2,503 to 2,158, '2' increased from 386 to 2,495, '3' increased from 191 to 2,501 and '4' increased from 22 to 2,503 samples. SMOTE-NCT is also used commonly in network intrusion detection [18].

Table 6. Distribution of class after hybrid sampling

Encoded variable	Imbalanced	SMOTE-RUS	SMOTE-TL	SMOTE-ENN	SMOTE-NCR
0	650	2503	2501	2405	2503
1	2503	2503	2501	1990	2158
2	386	2503	2503	2476	2495
3	191	2503	2503	2500	2501
4	22	22	2503	2503	2503



### 2.9.4. Deep learning synthesizers

The class distribution for the Gaussian copula generated dataset is not similar to the real dataset, as seen in Table 7. The minority class was changed from '4' to '3', but the result was a balanced distribution. On the other hand, for the CTGAN dataset, the synthetic data distribution follows the real dataset more than Gaussian copula. There was no change in the minority class but the overall class distribution for the synthetic data is more balanced than the real dataset.

Table 7. Distribution of class after deep learning synthesizers

Encoded variable	Imbalanced	Gaussian copula	CTGAN
0	650	662	527
1	2503	2758	2081
2	386	176	678
3	191	60	406
4	22	96	60

Figure 4 shows the differences in data distribution between Gaussian copula and CTGAN for one column which is the semester 9 CGPA (T9: *CUM\_GPA*). This column was chosen as it is the final semester grades for certain programs, and any useful information gleaned can be used in future works. The Gaussian copula chart shows that the synthetic data distribution does not follow the peaks and troughs of the real data distribution whereas the one made from CTGAN does. This informs that for this column, the Gaussian copula dataset does not follow the real data distribution as accurately as the CTGAN dataset.

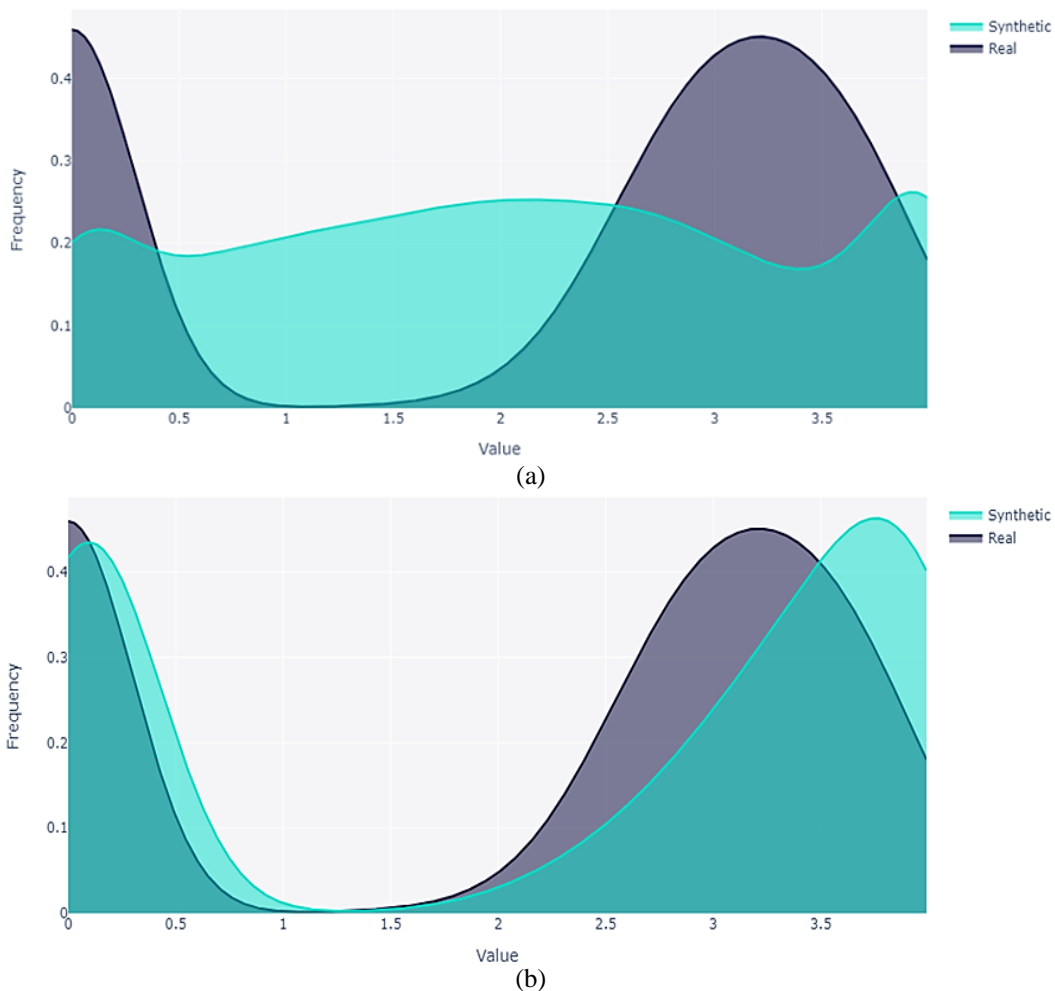


Figure 4. Comparing distribution of semester 9 CGPA between real dataset and (a) Gaussian copula and (b) CTGAN

### 3. RESULTS AND DISCUSSION

This section analyses the results of the comparison between the different types of class balancing methods. The evaluation is done in two steps. The first is by evaluating the dataset after class balancing methods have been applied using the popular classifier LightGBM [27]. LightGBM was chosen due to it being a histogram-based gradient boosting algorithm which leads to lower memory, faster training time and accuracy. It is generally much faster than other machine learning algorithms. The results from evaluating the dataset after different class balancing methods were applied are recorded in Table 8. The second step is evaluating only the synthetic datasets generated by the deep learning synthesizers using tabular data scores such as data quality, data coverage, column shapes score, and column pair trends score.

#### 3.1. Evaluation of class balancing methods

Whether regarding the F1 score or the ROC AUC score, all metrics for both Gaussian copula and CTGAN are low. This could be due to the low amount of student data in the original dataset, as it contains only approximately 5,000 rows, which may not be enough to generate a good quality synthetic dataset using deep learning models. When datasets become very large, the deep learning models has been shown to achieve competitive performance for other researchers [9], [29], [30]. The F1 score was around 0.25 for Gaussian copula and 0.29 for CTGAN.

Oversampling and hybrid sampling generally perform better than undersampling [31]–[33]. However, there are some cases where undersampling performs better than other techniques [7], [34]. These cases depend on a variety of factors, such as the size of the dataset, the type of predictive model used, the domain, the performance metrics, and so on. In this paper as well, TL and NCR from undersampling techniques outperformed the others. The reason they outperformed could be when those methods are used, all the noisy or ambiguous samples are removed, which allows the classifiers to better separate the individual classes. This is especially important for multiclass classification problems.

That said, not all undersampling techniques performed as well as Tomek link and NCR. Excluding the deep learning synthesizer results, the next worst result was from RUS, with an F1 score of 50%. This can be explained as RUS randomly eliminates large amounts of samples such that the number of samples in all classes becomes equal to the minority class samples which is 22. Reducing the number of samples from 3,652 samples to just 22 samples also lead to the loss of large amounts of information for the classifier, which may be the reason it performed so badly.

Table 8. Comparison of class balancing methods using LightGBM classifier

Class balancing approaches	Class balancing methods	Precision	Recall	F1	ROC
Imbalanced	Baseline	0.72	0.68	0.70	0.97
Oversampling	ROS	0.66	0.72	0.69	0.97
	SMOTE	0.67	0.72	0.69	0.97
	BSMOTE	0.67	0.72	0.69	0.95
	ADASYN	0.66	0.72	0.68	0.95
	RUS	0.51	0.63	0.50	0.85
Undersampling	TL	0.81	0.69	0.72	0.97
	ENN	0.76	0.65	0.68	0.96
	NCR	0.77	0.67	0.71	0.95
	SMOTE-RUS	0.67	0.72	0.69	0.97
Hybrid sampling	SMOTE-TL	0.67	0.71	0.69	0.96
	SMOTE-ENN	0.63	0.7	0.66	0.95
	SMOTE-NCR	0.65	0.71	0.67	0.96
	Gaussian copula	0.35	0.29	0.25	0.84
Synthesizers	CTGAN	0.31	0.31	0.29	0.75

#### 3.2. Evaluation of synthetic datasets

The synthesizers and the evaluation metrics used in this paper were from a package called synthetic data vault (SDV) [35]. Typically evaluating synthetic datasets can be done by inputting them into a classifier and comparing the results. However, they can also be evaluated based on their dataset structure including their data quality, column shapes, column pair trends and data coverage.

Data quality refers to the overall structure of synthetic data. It measures how closely the synthetic data matches with the real data. Data coverage refers to how much does the synthetic data follows the data distributions of the real data. It checks whether the synthetic data covers the real data's value range. Column shapes score refers to how closely each column of the synthetic data follows the real data and describes the overall column distribution change. The end score was calculated using the overall average for each column shape score. Similarly, the column pair trends score calculates how column pairs vary in relation to each

other. The data quality is calculated as the overall aggregation of the column shape score and the column pair trends score.

The overall quality of the dataset is the same for Gaussian copula and CTGAN models as seen in Table 9. The CTGAN model has better data coverage and better column pair trend score than Gaussian copula, likely due to being a more complicated and time intensive model compared to copula. This gave it more time to closely follow the shape of real data distribution and replicate the value ranges for the columns. Gaussian copula losses in column shapes but is better at replicating the relationship between columns in the real dataset than CTGAN.

Table 9. Comparison of synthetic dataset generated by deep learning models

Measure	Gaussian copula	CTGAN
Data quality	0.82	0.82
Column shapes	0.77	0.83
Column pair trends	0.87	0.83
Data coverage	0.96	0.99

#### 4. CONCLUSION

In conclusion, from the comparison of class balancing methods, it is observed that data-level sampling approaches perform much better than deep learning approaches. Tomek link and NCR methods from undersampling outperformed all other methods from oversampling and hybrid sampling, achieving an F1 score of 0.72 and 0.71. When comparing the quality of the synthetic datasets, it is observed that both synthesizers have the same overall data quality with CTGAN having better data coverage and higher column shape score and Gaussian copula having the better column pair trends score.

There are some limitations for this paper, which includes limiting the scope to data-level techniques against deep learning synthesizers. Adding methods from the algorithm-level and embedded level techniques for an updated comparison can be done as a future work. In addition, the educational dataset used was still relatively small for deep learning purposes which may have led to poor performance. Thus, focusing on gathering more data is critical. In addition, there can be more deep learning synthesizers to investigate for future work in class balancing, as this sub-field is still relatively new.

#### ACKNOWLEDGEMENTS

This research is supported by TM Research & Development Grant (TM R&D), MMUE/220028.




#### REFERENCES

- [1] M. M. Hussain, S. Akbar, S. A. Hassan, M. W. Aziz, and F. Urooj, "Prediction of student's academic performance through data mining approach," *Journal of Informatics and Web Engineering*, vol. 3, no. 1, pp. 241–251, Feb. 2024, doi: 10.33093/jiwe.2024.3.1.16.
- [2] P. Kumar, R. Bhatnagar, K. Gaur, and A. Bhatnagar, "Classification of imbalanced data: review of methods and applications," in *IOP Conference Series: Materials Science and Engineering*, Mar. 2021, vol. 1099, no. 1, pp. 1–8, doi: 10.1088/1757-899X/1099/1/012077.
- [3] F. Thabtah, S. Hammoud, F. Kamalov, and A. Gonsalves, "Data imbalance in classification: experimental evaluation," *Information Sciences*, vol. 513, pp. 429–441, Mar. 2020, doi: 10.1016/j.ins.2019.11.004.
- [4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
- [5] I. Tomek, "Two modifications of CNN," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, Nov. 1976, doi: 10.1109/TSMC.1976.4309452.
- [6] K. C. Khor, C. Y. Ting, and S. Phon-Amnuaisuk, "The effectiveness of sampling methods for the imbalanced network intrusion detection data set," *Advances in Intelligent Systems and Computing*, vol. 287, pp. 613–622, 2014, doi: 10.1007/978-3-319-07692-8\_58.
- [7] D. Javale, P. Pillai, P. Patel, and S. Jagtap, "A comparison of oversampling and undersampling methods for predicting air quality in metropolitan region," in *2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, May 2022, pp. 1615–1620, doi: 10.1109/ICAAIC53929.2022.9793084.
- [8] B. van Breugel, N. Seedat, F. Imrie, and M. van der Schaar, "Can you rely on your model evaluation? improving model evaluation with synthetic test data," in *Advances in Neural Information Processing Systems*, Dec. 2023, vol. 36, pp. 1889–1904.
- [9] V. Borisov, T. Leemann, K. Sebler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: a survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 7499–7519, Jun. 2024, doi: 10.1109/TNNLS.2022.3229161.
- [10] R. Mary Mathew and R. Gunasundari, "A review on handling multiclass imbalanced data classification in education domain," in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Mar. 2021, pp. 752–755, doi: 10.1109/ICACITE51222.2021.9404626.
- [11] L. Wang, M. Han, X. Li, N. Zhang, and H. Cheng, "Review of classification methods on unbalanced data sets," *IEEE Access*, vol. 9, pp. 64606–64628, 2021, doi: 10.1109/ACCESS.2021.3074243.
- [12] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proceedings of the 2000 International Conference on Artificial Intelligence*, 2000, pp. 111–117.




- [13] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning," in *Lecture Notes in Computer Science*, 2005, vol. 3644, no. PART I, pp. 878–887, doi: 10.1007/11538059\_91.
- [14] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, Jun. 2008, pp. 1322–1328, doi: 10.1109/IJCNN.2008.4633969.
- [15] F. J. Ferri, J. V. Albert, and E. Vidal, "Considerations about sample-size sensitivity of a family of edited nearest-neighbor rules," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 5, pp. 667–672, 1999, doi: 10.1109/3477.790454.
- [16] J. Laurikkala, "Improving identification of difficult small classes by balancing class distribution," in *Lecture Notes in Computer Science*, 2001, vol. 2101, pp. 63–66, doi: 10.1007/3-540-48229-6\_9.
- [17] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, Jun. 2004, doi: 10.1145/1007730.1007735.
- [18] Y. Sun and F. Liu, "SMOTE-NCL: a re-sampling method with filter for network intrusion detection," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct. 2016, pp. 1157–1161, doi: 10.1109/CompComm.2016.7924886.
- [19] I. Pratama, Y. Pristyanto, and P. T. Prasetyaningrum, "Imbalanced class handling and classification on educational dataset," in *2021 4th International Conference on Information and Communications Technology (ICOIACT)*, Aug. 2021, pp. 180–185, doi: 10.1109/ICOIACT53268.2021.9563968.
- [20] E. Buraimoh, R. Ajoodha, and K. Padayachee, "Importance of data re-sampling and dimensionality reduction in predicting students' success," in *2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, Jun. 2021, pp. 1–6, doi: 10.1109/ICECCE52056.2021.9514123.
- [21] A. Figueira and B. Vaz, "Survey on synthetic data generation, evaluation methods and GANs," *Mathematics*, vol. 10, no. 15, pp. 1–41, Aug. 2022, doi: 10.3390/math10152733.
- [22] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional GAN," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, vol. 32, pp. 1–11.
- [23] G. Douzas and F. Bacao, "Effective data generation for imbalanced learning using conditional generative adversarial networks," *Expert Systems with Applications*, vol. 91, pp. 464–471, Jan. 2018, doi: 10.1016/j.eswa.2017.09.030.
- [24] Y. Zhang, N. A. Zaidi, J. Zhou, and G. Li, "GANBLR: a tabular data generation model," in *2021 IEEE International Conference on Data Mining (ICDM)*, Dec. 2021, vol. 2021-Decem, pp. 181–190, doi: 10.1109/ICDM51629.2021.00103.
- [25] X. Du, J. Yang, and J.-L. Hung, "An integrated framework based on latent variational autoencoder for providing early warning of at-risk students," *IEEE Access*, vol. 8, pp. 10110–10122, 2020, doi: 10.1109/ACCESS.2020.2964845.
- [26] D. Micci-Barreca, "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems," *ACM SIGKDD Explorations Newsletter*, vol. 3, no. 1, pp. 27–32, Jul. 2001, doi: 10.1145/507533.507538.
- [27] G. Ke *et al.*, "LightGBM: a highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, 2017, vol. 30, pp. 1–9.
- [28] D. Bajer, B. Zonc, M. Dudjak, and G. Martinovic, "Performance analysis of SMOTE-based oversampling techniques when dealing with data imbalance," in *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jun. 2019, vol. 2019-June, pp. 265–271, doi: 10.1109/IWSSIP.2019.8787306.
- [29] D. McElfresh *et al.*, "When do neural nets outperform boosted trees on tabular data?," in *Advances in Neural Information Processing Systems*, Dec. 2023, vol. 36, pp. 1–34.
- [30] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?," *Advances in Neural Information Processing Systems*, vol. 35, pp. 507–520, Dec. 2022.
- [31] V. García, J. S. Sánchez, A. I. Marqués, R. Florencia, and G. Rivera, "Understanding the apparent superiority of over-sampling through an analysis of local information for class-imbalanced data," *Expert Systems with Applications*, vol. 158, Nov. 2020, doi: 10.1016/j.eswa.2019.113026.
- [32] R. Mohammed, J. Rawashdeh, and M. Abdullah, "Machine learning with oversampling and undersampling techniques: Overview study and experimental results," in *2020 11th International Conference on Information and Communication Systems (ICICS)*, Apr. 2020, pp. 243–248, doi: 10.1109/ICICS49469.2020.239556.
- [33] T. Wongvorachan, S. He, and O. Bulut, "A comparison of undersampling, oversampling, and SMOTE methods for dealing with imbalanced classification in educational data mining," *Information*, vol. 14, no. 1, pp. 1–15, Jan. 2023, doi: 10.3390/info14010054.
- [34] B. W. Yap, K. A. Rani, H. A. A. Rahman, S. Fong, Z. Khairudin, and N. N. Abdullah, "An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets," in *Lecture Notes in Electrical Engineering*, 2014, vol. 285, pp. 13–22, doi: 10.1007/978-981-4585-18-7\_2.
- [35] N. Patki, R. Wedge, and K. Veeramachaneni, "The synthetic data vault," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Oct. 2016, pp. 399–410, doi: 10.1109/DSAA.2016.49.

## BIOGRAPHIES OF AUTHORS






**Rithesh Kannan**    received the B.Sc. degree in computer science, majoring in data science, from Multimedia University (MMU), Malaysia, in July 2021. He is currently studying for his Master of Science (information technology) degree by research at MMU. He was placed on the Dean's list of Faculty of Computing and Informatics during his bachelor's studies for academic excellence. He presented his final year project (FYP) at the IEEE Taiwan conference and received third place for Best Paper in 2022. He can be contacted via email at kannanrithesh00@gmail.com.






**Hu Ng**    is currently a lecturer in the Faculty of Computing and Informatics at Multimedia University, Malaysia. He received his Doctor of Philosophy (Ph.D.) degree from Multimedia University, Malaysia, in January 2015. His Ph.D. study involves image processing, feature selection, and classification techniques applied in a gait recognition system. He has been registered as a professional technologist with the Malaysia Board of Technologists (MBOT) since 2018. His current research interests include biometrics, pattern recognition, machine learning, and data engineering. He can be contacted via email at [nghu@mmu.edu.my](mailto:nghu@mmu.edu.my).






**Timothy Tzen Vun Yap**    received the B.Eng. degree (Hons.) in electronics engineering, majoring in computer from Multimedia University, Malaysia, in 2002, and the M.Eng.Sc. and Ph.D. degrees from the System Identification and Control Group, Multimedia University, in 2006 and 2017, respectively. He is currently an assistant professor with the School of Mathematical and Computer Sciences at Heriot-Watt University, Malaysia. His current research interests include system identification, blockchain, data engineering, and machine learning. He can be contacted via email at [timothy.yap@hw.ac.uk](mailto:timothy.yap@hw.ac.uk).






**Lai Kuan Wong**    received her B.Sc. degree in computer science from Universiti Sains Malaysia (USM) and M.Sc. and Ph.D. degrees in computing from the School of Computing at the National University of Singapore (NUS). She is currently an associate professor at the Faculty of Computing and Informatics (FCI), Multimedia University (MMU), Malaysia. Her research interests include computer vision, computational photography, computational aesthetics, and medical imaging. She can be contacted via email at [lk Wong@mmu.edu.my](mailto:lk Wong@mmu.edu.my).






**Fang Fang Chua**    is currently working as an associate professor in the Faculty of Computing and Informatics at Multimedia University, Cyberjaya, Malaysia. She obtained a Bachelor of Information Technology (Hons) in software engineering from Multimedia University, Malaysia. She then received her master's degree from The University of Melbourne, Australia, and her Ph.D. from Multimedia University, Malaysia. Her research includes learning technologies, software engineering, business process management, adaptive systems, and analytics. She has published several research papers and has been invited as a reviewer and program committee member for various reputable international journals and conferences. She is involved in multiple research projects funded by JICA & SASTREPS, TM R&D, MOHE FRGS, and PRGS. She can be contacted via email at [ffchua@mmu.edu.my](mailto:ffchua@mmu.edu.my).






**Vik Tor Goh**    is an associate professor in the Faculty of Engineering, Multimedia University, Malaysia. He received his B.Eng. (Hons.) electronics majoring in Telecommunications and M.Eng.Sc. degrees from Multimedia University, Malaysia in 2002 and 2006, respectively. Following that, he obtained a Ph.D. in computer science from Queensland University of Technology, Australia in 2010 for his research on intrusion detection in encrypted networks. His current research interests include information security, indoor positioning technology and mobile services. In addition to being a qualified chartered engineer (C.Eng., IET), he is also certified as a CISSP, CEH, CCNA-CCAI and HCDA. He can be contacted via email at [vtgoh@mmu.edu.my](mailto:vtgoh@mmu.edu.my).



**Yee Lien Lee**    obtained B.Eng. and M.Eng degrees in 2002 and 2007 respectively. She is attached to the Faculty of Engineering, Multimedia University. She is currently part of the Center for Lifelong Education and Learning Innovation (LEARN). She can be contacted via email at [yllee@mmu.edu.my](mailto:yllee@mmu.edu.my).



**Hwee Ling Wong**    received the B. Eng. degree in electrical engineering, majoring in mechatronics and the M. Eng. degree focusing on robot vision and human-robot interaction from Universiti Teknologi Malaysia in 2002 and 2005. She has five years of industrial experience in semiconductor manufacturing, specifically in microcontroller product testing. She joined Multimedia University in 2010 and has been with the university ever since. Her research interests are image processing and visual-based automation. She can be contacted via email at [hlwong@mmu.edu.my](mailto:hlwong@mmu.edu.my).