VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DATA DRIVEN SYSTEM TO PREDICT THE ACADEMIC GRADES AND DROPOUT IN HIGHER EDUCATION

By

NGUYEN ANH TUAN

Supervisor: Dr. Nguyen Thi Thanh Sang

A thesis submitted to the School of Computer Science and Engineering in partial fulfillment of the requirements for the degree of Bachelor of Information Technology.

Ho Chi Minh City, Vietnam

2023

DATA DRIVEN SYSTEM TO PREDICT THE ACADEMIC GRADES AND DROPOUT IN HIGHER EDUCATION

APPROVED BY: ________________________________ ,

____

____

____

____

THESIS COMMITTEE

ACKNOWLEDGMENTS

Firstly, It is with deep gratitude and appreciation that I acknowledge the professional guidance of Dr. Nguyen Thi Thanh Sang. I received a lot of enthusiastic advice and support during my study, which inspired and motivated me to learn new things and prepare for pre-thesis and thesis also.

Secondly, I would like to thank Lecturers in International University and Authors with their research papers and contributions which I referenced and learned from. I really thank these people's assistance with great joy and a great deal of appreciation.

Afterward, I would like to show my appreciation to the International University - Vietnam National University of Ho Chi Minh City and the School of Computer Science and Engineering for establishing favorable circumstances for me to enroll in the course "Special Study of the Field" in the previous semester and take advantage of those brief for this thesis in the Fall semester of 2023 - 2014.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ANN:	Artificial Neural Network.
DT:	Decision Tree.
DNN:	Deep Neural Network.
EDM:	Educational Data Mining.
FP:	False Positive.
FN:	False Negative.
GD:	Gradient Descent.
IT:	Information Technology.
LSTM:	Long- Short Term Memory.
LMS:	Learning Management System.
MLP:	Multi-layer Perceptrons.
PLA:	Perceptron Learning Algorithm.
RF:	Random Forest.
ROS:	Random Oversampling Technique.
ReLU:	Rectified Linear Unit.
RNN:	Recurrent Neural Network.
SVM:	Support Vector Machine.
SMOTE:	Synthetic Minority Oversampling Technique.
TP:	True Positive.
TN:	True Negative.

ABSTRACT

Nowadays, With the digitization of the systems in the universities, academic institutions create a large volume of student-related data using computerized form. For the exploration of hidden information from this golden data, a lot of techniques and tools, introduced in recent years, help the data analysis process become easier than ever before. In higher education systems, applying a data driven system can help them take advantage of this data resource for the research of talent or final year students, and scientists in many departments to analyze behaviors, affecting factors… Instead of wasting the resources for storing useless data, This can be a new approach to have general and multi-dimensional insights to improve the education qualification day by day. The changes are based on the actual data of the university itself.

Along with the data driven system, Data mining, which is used as an analytical tool, becomes essential for these departments to transfer substantial amounts of data and manipulate it into useful knowledge. Therefore, educational data mining techniques were created for constructing predicting or classifier models built from the student historical records. In this context, automated systems supporting the lecturer are needed. One significant problem is not being able to predict a student's academic grade in a course, which would help them to achieve better results in the future course. Therefore, the main goals of this research are to explore and implement the efficiency of machine learning in the field of Educational data mining, especially in predicting aspects of student’s performance, including Grade prediction, Dropout or Course recommendation system in advance…

CHAPTER 1

INTRODUCTION

1.1. Background

With the rising development in massive data warehouses, the requirement for analyzing and extracting hidden information becomes a common question and a rich scholarly topic for many researchers. Building a data driven system has become an essential need in many corporations or companies in recent years. It is easier for technical or non-tech employers to access the needed data. Along with this system, the process of sorting through datasets to identify patterns and build the relationship to solve problems through data analysis is known as Data mining or Knowledge discovery. Educational Data Mining(EDM) is data mining used in the education fields. EDM applies various data mining techniques such as classification, regression, time series analysis and association rule mining… to analyze and evaluate several aspects of educational datasets collected from students, departments or higher educational institutions.

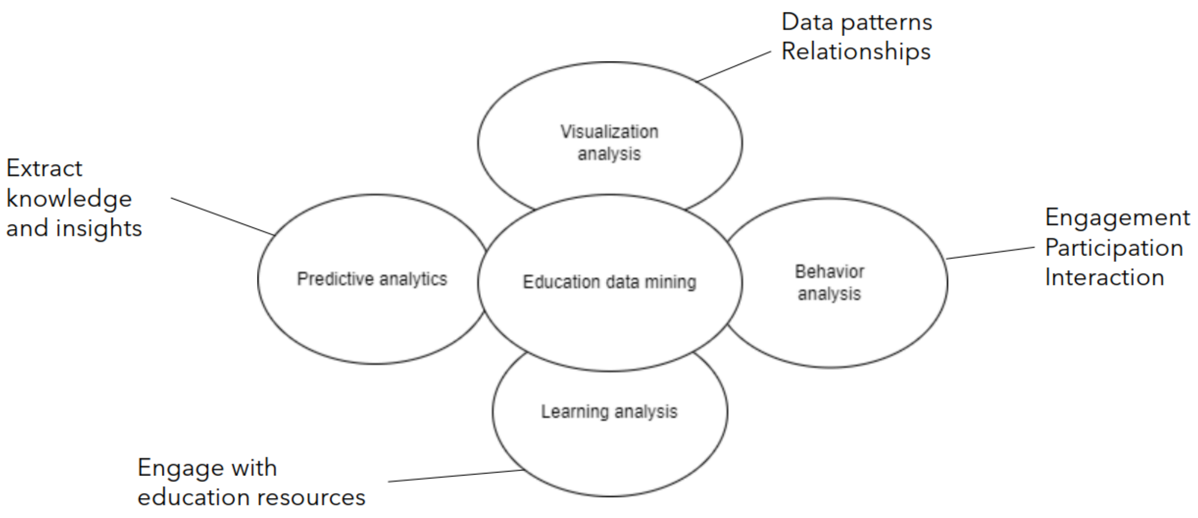


FIGURE 2.0.1: *Data Mining Scheme in Higher Education.*

Education plays an important role in the development of a country. Any educational institution makes an effort to create an optimal education environment which can help students and teachers to improve their performance in learning and teaching process. Therefore, many institutions have started to apply the technology to provide high-quality education for their partners to enhance their success rate. Nowadays, universities are generating massive amounts of data which can be used for data analytics for the decision-process making to improve the traditional teaching and learning process. One of the most important applications of educational data mining is to early predict student's performance and identify the students at risk of failure, or the leaving the course rate.

1.2. Scope and Objectives

In this study, a dataset collected from the University will be used to build a predictive model using various machine learning algorithms to analyze and evaluate student's data. Prior researches are successful to build accurate grade predictions based on the state of the art linear models, including decision tree(DT), random forest (RF), support vector classifier (SVC)...However, these linear models are sufficiently unable to capture the complexity of student's knowledge accumulation process throughout a sequence of semesters[1]. In this work, we keep an eye on two types of Bayesian Neural Network, namely, Multi-layer Perceptron(MLP) and Recurrent Neural Network(RNN). They are built based on a general assumption that the pre-requisite or outline course's knowledge can affect the achievement of future courses, so that these can be used to predict the academic grade of students. In Multilayer Perceptron, the hierarchical hidden layers, which help this to capture complex relationships or interactions between features, is the main distinction to deal with the non-linearity data. But, it also ignores the timestamp characteristic of data and can assume that it has the same predicting concepts with the linear models in many researches of this field. Therefore, we proposed another algorithm RNN for this course-specific model which not only applies for the result data, but also focuses on the knowledge's evolution cross sequence of semesters. The unbalanced dataset issue is one of the key elements influencing classifier performance. In the realm of EDM, it is a significant challenge that might result in inaccurate outcomes and poor performance[2]. Hence, an approach of this study is solving the imbalanced dataset problems using a resampling technique SMOTE to enhance the performance of the model and to achieve a better result. Another approach in this study, a driven system based on the multi-dimensional database can be built to provide a friendly user interface with visualizations, filter for query between features and apply the course-specific models to predict grade as well… This is not only an easier way to load and extract data into the needed form for the exploring and analyzing, but also help the lecturers and advisors have the general insight about the students, affecting factors…

1.3. Structure of thesis

This thesis report is divided into five parts including Introduction, Literature Review, Research Methodology, Experimental and Result, and Conclusion and Future Work.

CHAPTER 2

LITERATURE REVIEW

2.1. Artificial Neural Network

Artificial neural network (ANN) is a technology depending on the human brain structure. ANN, known as a computational model, comprises many processing elements that take inputs and deliver output based on the pre-defined activation functions. The perceptron is the simplest form of Artificial Neural Network with input layer and output layer. For more complex applications, multi-layer perceptrons (MLP) that have one or more hidden layers between input layer and output layer are used. Backpropagation is the most popular method used for training a neural network.

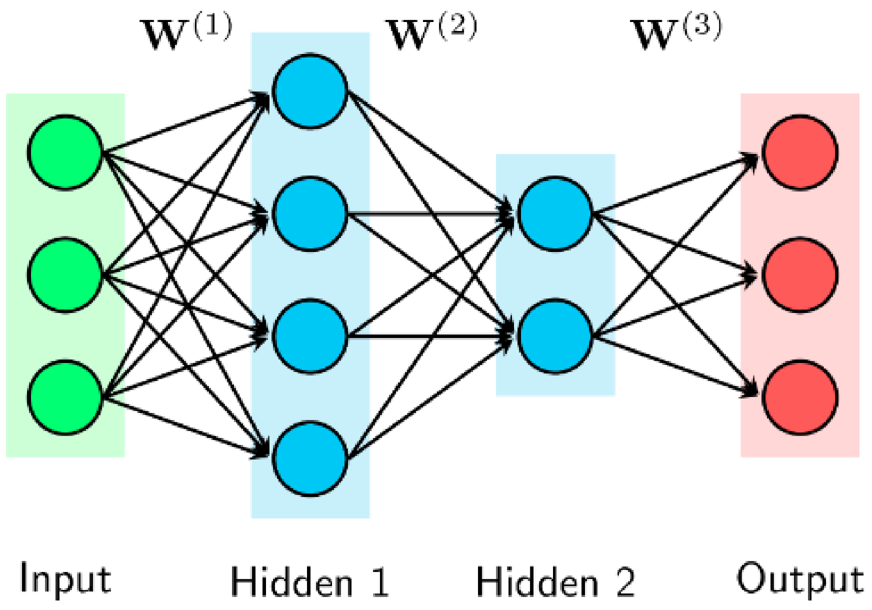


FIGURE 2.1. *Multi-layer Perceptrons with two hidden layers.*

2.1.1. Activation function

In training a neural network, it is an important thing to decide which activation function can be used in the hidden layer as well as at the output layer of the network. In an artificial neural network, activation functions are utilized to convert the input to output signals that are then sent as input to the next layer in the stack. In an artificial neural network, we calculate the sum of products of inputs and their corresponding weights and finally apply an activation function to it to get the output of that particular layer and supply it as the input to the next layer[3]. There are some commonly used activation functions including:

- **Sigmoid activation function** has the form: $\sigma(x) = \frac{1}{(1+e^{-x})}$. In particular, It will be 0 with the large negative number and become 1 with the large positive number. From the sigmoid function, we have the Tanh function: $tanh(x) = 2\sigma(2x) - 1$. The Tanh non- linearity receives the real numbers between [-1; 1].

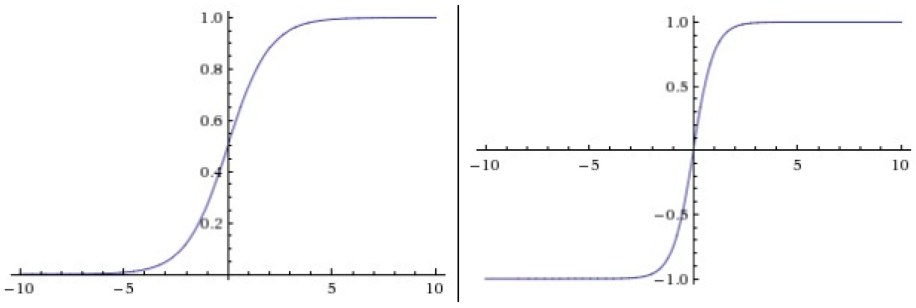


FIGURE 2.1.1. *Sigmoid (Left) and Tanh activation function (from: CS231n Convolutional Neural Network Visual Recognition).*

- **ReLU (Rectified Linear Unit):** has the form $f(x) = max(0, x)$. In the other word, the ReLU activation function becomes 0 when $x < 0$, and then linear with the slope 1 with $x > 0$.

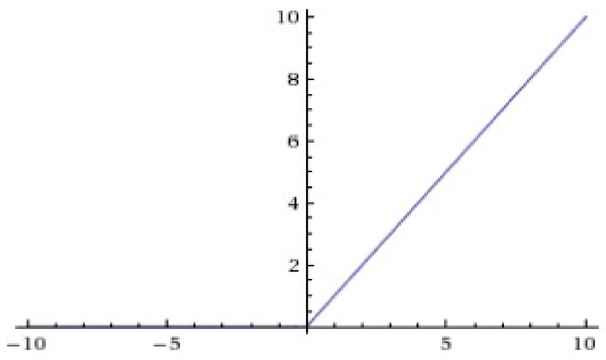


FIGURE 2.1.2. *ReLU activation function (from: CS231n Convolutional Neural Network Visual Recognition).*

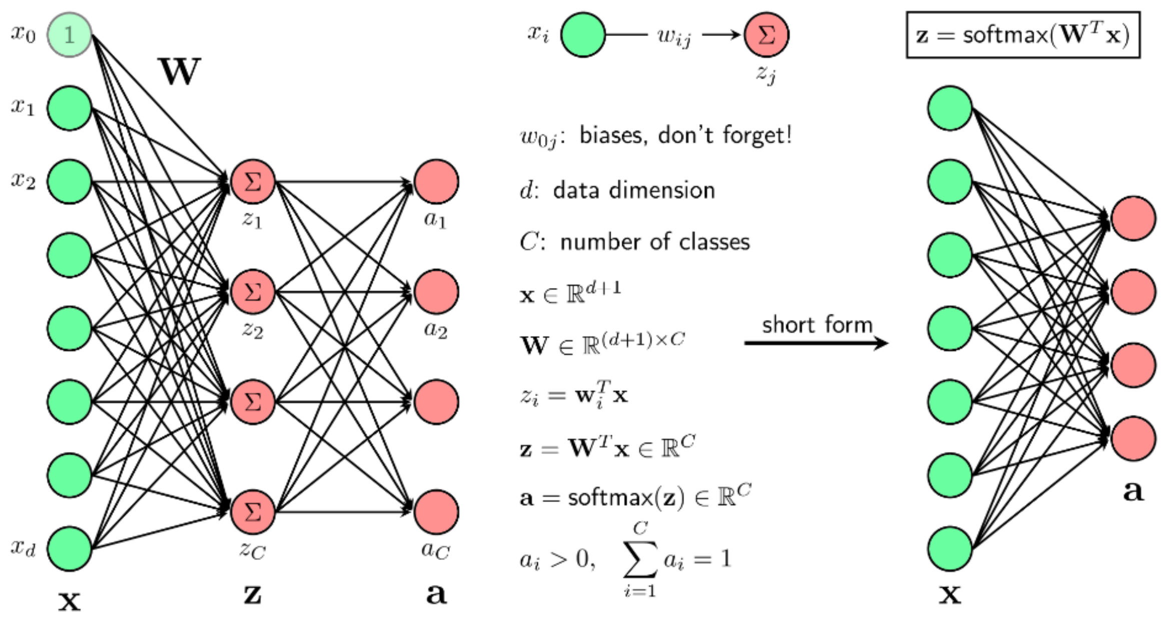
2.1.2. Multi-Layer Perceptron

Multi-Layer Perceptron(MLP) is the ANN with one or more hidden layers between the input layer and output layer. It is incredibly prevalent in industry applications as classification or recognition. Each hidden layer uses an activation function, often an ReLU, to map the weight to a standard state. For multiclass classification, a softmax non- linearity (eq. 2.1.2) can be used for the last activation function to normalize the output of a network to a probability distribution over the predicted classes.

$$a_i = \frac{exp(z_i)}{\sum_{j=1}^{C} exp(z_j)}, \forall\, i = 1, 2, 3\ldots C \; (2.1.2)$$

In this equation:

- C: number of classes.
- a_i : probability which an input is classified into class i.



x_0
1
$\mathbf{W}$
x_1
x_2
x_d
$\mathbf{x}$
z_1
z_2
z_C
$\mathbf{z}$
a_1
a_2
a_C
$\mathbf{a}$
x_i
w_{ij}
z_j
w_{0j}: biases, don't forget!
d: data dimension
C: number of classes
$\mathbf{x} \in \mathbb{R}^{d+1}$
$\mathbf{W} \in \mathbb{R}^{(d+1)\times C}$
$z_i = \mathbf{w}_i^T\mathbf{x}$
$\mathbf{z} = \mathbf{W}^T\mathbf{x} \in \mathbb{R}^C$
$\mathbf{a} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^C$
$a_i > 0, \quad \sum_{i=1}^{C} a_i = 1$
short form
$\mathbf{z} = \text{softmax}(\mathbf{W}^T\mathbf{x})$

FIGURE 2.1.3. *Softmax in the simple neural network (From: machinelearningcoban.com).*

The output of each node (except the input units) can be calculated by:

$$a_i^{(l)} = f(W_i^{(l)T} a^{(l-1)} + b_i^{(l)}) \quad (2.1.3)$$

And, in the vector form, we have:

$$a^{(l)} = f(W^{(l)T} a^{(l-1)} + b^{(l)})$$

- $f(.)$: is the non-linear activation function. The activation function $f(.)$ will apply for each element of the matrix or vector. Then, using element-wise technique to have the matrix, same size as the input.
- $W^{(l)}$: is the weight matrix or vector. This represents for the connections from layer $(l-1)^{th}$ to layer l^{th}. For example, $w_{ij}^{(l)}$ is the connection from node ith of layer $(l-1)$ to the node jth of layer l.
- $b^{(l)}$: is the vector of biases.
- $a^{(l)}$: is the output of the ith unit in the the lth layer.

According to the above, we apply the softmax function to the output layer's activation. Then, we minimize the cross entropy between the target output and the actual output. Cross Entropy can be a good choice because it can present the distance of two probability distributions.

$$H(p, q) = -\sum_{i=1}^{C} p_i \log q_i$$

Where:

- C: number of classes
- q: is the softmax output
- p: is the target output

The most common optimizer for MLP and complex MLP, namely Deep Neural Network (DNN) is Gradient Descent (GD). To apply the GD, we must calculate the gradient of loss function with the weight matrix $W^{(l)}$ and vector bias $b^{(l)}$ (with l is the layer). It is extremely complicated to calculate this value directly because the loss function does not depend directly on the coefficients. The most common method used is called Backpropagation which calculates the reverse gradient from the last layer to the first layer. There are different concepts of optimizer available such as Stochastic Gradient Descent, Mini-batch Gradient Descent… In practice, finding the number of hidden units and non-linear activations is often impossible. Instead, experiments demonstrate that Neural Networks with many hidden layers combined with nonlinear activation functions (as simple as ReLU) have better training data representation (in approximation). In addition, as the hidden layers increase, the optimized coefficients also increase and the model becomes more complex. This complex can affect two aspects. The first is speed of computation. Secondly, if the model is too complex, it can represent the training data very well, but the test dataset is not. This phenomenon can lead to overfitting.

2.1.3. Recurrent Neural Network (RNN)

A neural network model always has three main components, namely, input layer, hidden layer, and output layer which are independent with each other. As a result, this is not well-suited to apply for the sequential or continuous data, since the future prediction depends on historical forecast or position. Based on the condensed research of 1925, 1972, 1982, and 1986, a system that can handle very deep learning problems with 1000 subsequent layers in an RNN was developed in 1993. Recurrent neural networks, which are based on neural networks, are constructed with integrated internal memory that enable them to precisely predict what will happen next by remembering important information from the input. This is why they are extensively used in several deep learning domains for sequential data like time series, speech, text, financial data, audio, video, weather and much more. With the addition of recurrent edges spanning neighboring time steps, recurrent neural networks introduce a concept of time into the model. Recurrent neural networks are a strict superset of feedforward neural networks. Recurrent edges have the ability to generate cycles, including self-connections, even if RNNs may not have cycles among their conventional edges.[4]. As a result, the RNN has two inputs: the present and recent past (Fig.2.1.4). At time t, along with the current example $x_{(t)}$, nodes that receive input via the recurrent edges get the input activation from the hidden nodes $h_{(t-1)}$ in the prior state of the networks. On one hand, at the time t, the input $x_{(t)}$ will combine with the hidden layer $h_{(t-1)}$ by a function f_w to calculate the current $h_{(t)}$ and imply the output y_t from it . Hence, because of these recurrent connections, the output y_t at the time t can be influenced by the $x_{(t-1)}$ of the time $t - 1$.

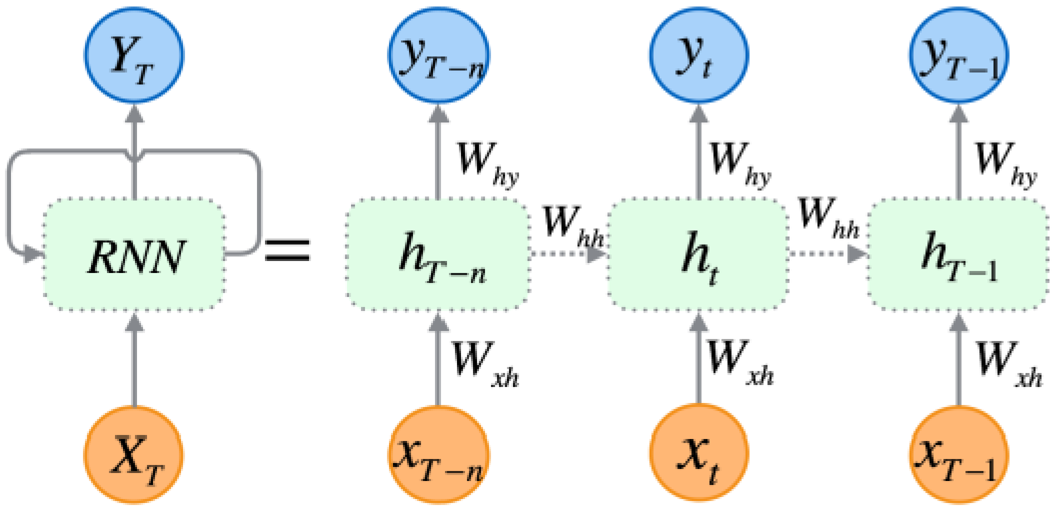


FIGURE 2.1.4. *The Folded and Unfolded Structure of RNN. (from: stackoverflow.com).*

This calculation process will be written as the mathematical form, as follow:

$$h_t = f_w(h_{t-1}, x_t) \qquad (2.1.4)$$

In practice, the f_w is often the function of $tanh$. The equation 2.1.4 will become:

$$h_t = tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h) \qquad (2.1.5)$$

$$y_t = softmax(W_{hy}h_t + b_y) \qquad (2.1.6)$$

Where:

- W_{hh} : is the matrix of weight between hidden layers at adjacent time steps, also known as W.
- W_{xh} : is the matrix of weight between hidden layers and the input x, also known as U.
- W_{hy}: is the matrix of weight of hidden-to-output connections, also known as V.

Training the learning model is done by specifying the loss function which estimates the difference between true label and the output, and minimizing it by using the backward pass or forward pass. Although training the RNN is similar to the traditional neural network, the difference is sharing parameters by all time steps in the networks. As a result, learning on the recurrent neural network can be especially hard due to the difficulty of learning long-range dependencies as described by Bengio et al in 1994 [4][5]. The well known issues of RNN, including vanishing and exploding gradient arising when propagating errors across several time steps. According to Bengio et al in 1994, the significant increase in the gradient's norm during training is referred to as the "exploding gradients" issue. The growth of the long-term components, which might develop exponentially more than the short-term ones, is the factor that causes such occurrences. The reverse tendency, known as the "vanishing gradient problem," occurs when long term components approach norm 0 exponentially quickly, making it difficult for the model to learn connection between events that occur over extended time periods[6][5].

From 1997, the most successful RNN architectures for sequential and time-series data were introduced, namely Long Short-Term Memory (LSTM) to overcome encountered problems in the earlier recurrent networks. Long Short-Term Memory by Hochreiter and Schmidhuber introduced a memory cell or block, unit computation replacement for the artificial neutrons in the hidden layers of neural networks[4]. In general, memory blocks, or a collection of recurrently connected sub-networks, define the majority of the LSTM architecture. The memory block's tasks involve using nonlinear gating units to regulate information flow and maintaining its state over time, as described et al [7]. As a result, the LSTM is capable of removing or adding the information to the memory block through a specific structure, namely gate which is a combination of hidden sigmoid activation function and the multiplication operator. As mentioned above, the sigmoid activation function outputs a probability value from 0 to 1 which decides how much information will pass through the gate. Fig 2.1.5 presents the architecture of LSTM consisting the input sample $x_{(t)}$, gates (input gate, output gate, and forget gate), the output $h_{(t)}$ and the activation functions. The output $h_{(t)}$ is connected recurrently back to the input to serve for the future calculation process. (All the σ notations below stand for the sigmoid activation function.)

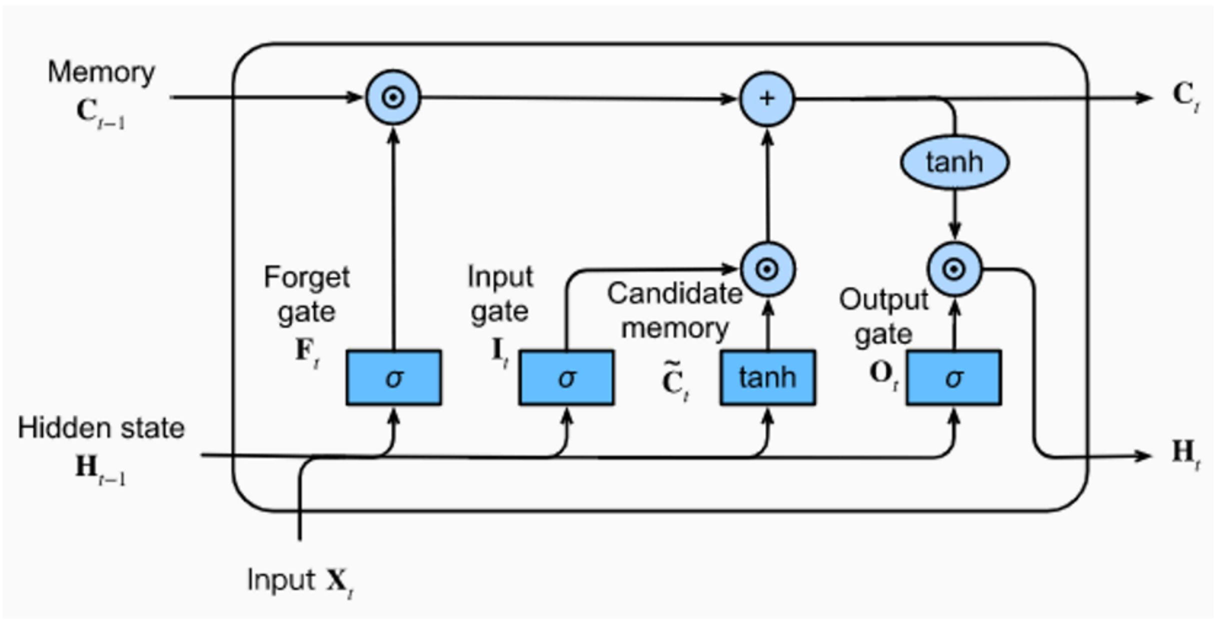


FIGURE 2.1.5. *The LSTM architecture (Based on: Toward Data Science).*

- **The Forget Gate:** In this step, the LSTM unit determines which information should be removed from its previous cell states $c_{(t-1)}$[7]. In time step t, the forget gate receiving the values of $h_{(t-1)}$, and the current sample $x_{(t)}$ returns an activation value from 0 to 1 for each previous cell state value $c_{(t-1)}$ via sigmoid activation function. If value of $c_{(t-1)}$ equals 1, all of the information in the previous time step is taken advantage of for the next time step, and otherwise.

$$f_{(t)} = \sigma(W_f [h_{(t-1)}, x_{(t)}] + b_f)$$

- **The Input Gate:** this gate is used for updating the input information based on the recurrent value $h_{(t-1)}$, the current sample $x_{(t)}$ and the previous cell state $c_{(t-1)}$. In general, this step decides what new information will be stored or updated in the cell state via sigmoid activation function, as activation value $i_{(t)}$ of the input gate.

$$i_{(t)} = \sigma(W_i [h_{(t-1)}, x_{(t)}] + b_i)$$

In previous steps, LSTM layers determine which information should be retained and removed in the network's cell state for the upcoming timestep. Next, the another hidden $tanh$ layer will create the vector of new candidates, $z_{(t)}$ that could potentially be added to the cell state[7]. Combination of these two value $i_{(t)}$ and $z_{(t)}$ outputs an update to the cell state. As a result, with the forget and update information in the earlier steps, we can update the old cell state $c_{(t-1)}$ to the new cell state $c_{(t)}$.

$$z_{(t)} = tanh(W_z [h_{(t-1)}, x_{(t)}] + b_z)$$

$$c_{(t)} = f_{(t)} * c_{(t-1)} + i_{(t)} * z_{(t)}$$

- **The Output Gate:** the output is calculated based on cell state vector, but will be a filtered version. Firstly, the sigmoid layer which is fed recurrent value $h_{(t-1)}$, the current $x_{(t)}$ determines which portion of cell state will be the value of output. It is multiplied by the cell state's $tanh$ activation value to output the parts we decided to.

$$o_{(t)} = \sigma(W_o [h_{(t-1)}, x_{(t)}] + b_o)$$

$$h_{(t)} = o_{(t)} * tanh(c_{(t)})$$

2.2. Decision Tree (DT)

Decision trees are a widely used classification method in data mining that are constructed based on the structure of the tree. Every internal node in a decision tree indicates a test on an attribute, every branch indicates the test's result, and every leaf node indicates a class. A decision tree model is used to classify a record by measuring the attributes test and determining a route from the root to the leaf, where the attribute represents the classification outcome[8]. In the recursive top-down, it compares the attributes between internal nodes in the decision tree, judges the downward branches according to the different attributes of the node, and returns the conclusion from the leaf of the decision tree[8]. Decision tree is a widely used learning system of classification in data mining. The process of developing a decision tree classification algorithm typically consists of two stages: decision tree construction and pruning. For constructing the decision tree, the key of this step is which logical judgment or attributes could be chosen or how to choose. According to research, a decision tree's prediction capacity tends to be stronger the smaller it is. Selecting the appropriate attribute-caused branch is the important step to build a tiny decision tree. The method of measuring impurity on the different subset instances determines which attributes are selected. Information gain, gain ratio, Gini index, distance measurement, X2 statistics, the weight of the evidence, the minimum description length, and other techniques are examples of impurity measuring techniques[8][9]. Next, Pruning is a technique to deal with noise and also help the tree be simplified and easy to understand. There are two basic pruning strategies including Forward-pruning (pruning before decision tree's growth process) and Post - pruning (pruning after decision's tree growth process).

2.3. Support Vector Machine (SVM)

Support vector machine is a framework of statistical learning theory which has been successfully applied in many fields of machine learning such as time series, face recognition or medical diagnosis. Assuming that there are two linearly separable classes which can be present in the multidimensional space, there exists at least one hyperplane dividing these classes. Although the perceptron learning algorithm (PLA) can be used to deal with this problem, the number of solutions can be more than 1, 2 or infinite. From those, which solution is suitable according to the specific standards and how to find that. In the first standard, we find a hyperplane that the distance from the closest data point of each class to it is equal. This distance is known as margin. Secondly, maximum margin can help us have a clearer division between two classes. Optimization problems in support vector machines(SVM) are to figure out the boundary with the maximum margin, sometimes namely, maximum margin classifier. Data points that are closer to the hyperplane than the others and influence the position and orientation, are known as support vectors. To build a SVM, we maximize the margin using these support vectors and delete ones that could affect the position of the hyperplane[10].

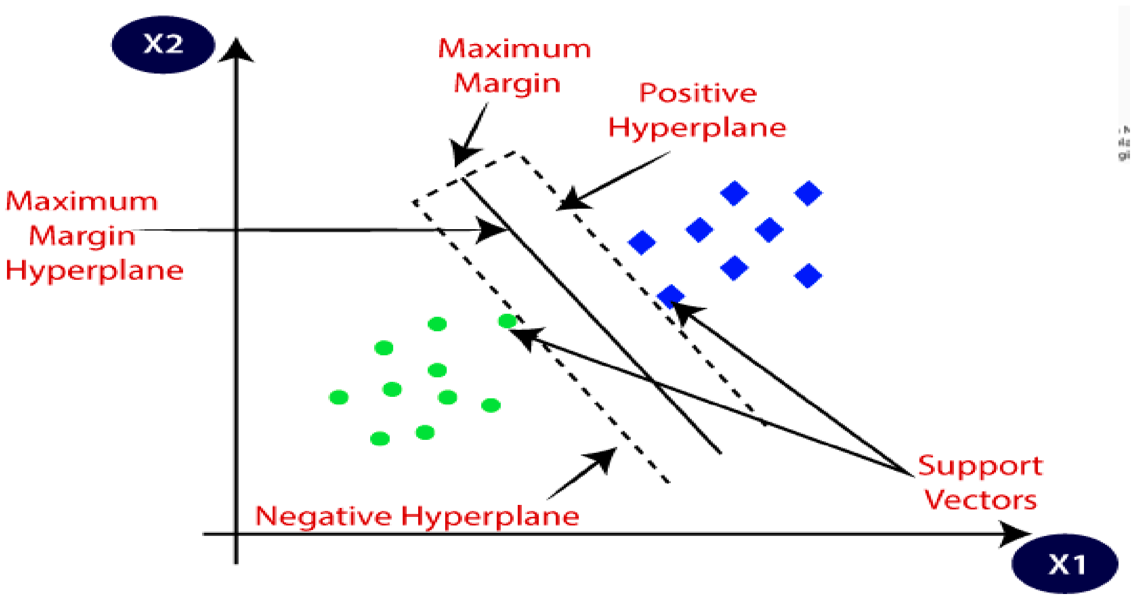


FIGURE 2.2.1. *Support Vector Machine (From: https://javatpoint.com)*

2.4. Class imbalanced problem

Class imbalance is a common problem in any practical cases. It occurs if one of the classes has a smaller amount than the other classes. This smaller class is known as minority and the other is called majority. When it comes to solving unbalanced data problems, two methods should be used: data-level solutions and algorithm-level solutions[11]. The distribution of the majority and minority classes is balanced at the data level by using techniques such as undersampling, oversampling, or combination of both methods. The algorithm-level solution is implemented via modifying classifier techniques or enhancing the efficiency of learning algorithms[12]. In data level, under sampling is a process where data on the majority class can be ignored to balance the distribution, while over sampling is created by adding some data to the minority class. The combination of oversampling and undersampling, that can be used to achieve the balance distribution between majority and minority class, is called combination or hybrid. In this study, we focus on the solutions at the data level including SMOTE (Synthetic Minority Oversampling Techniques) and ROS (Random Over Sampling).

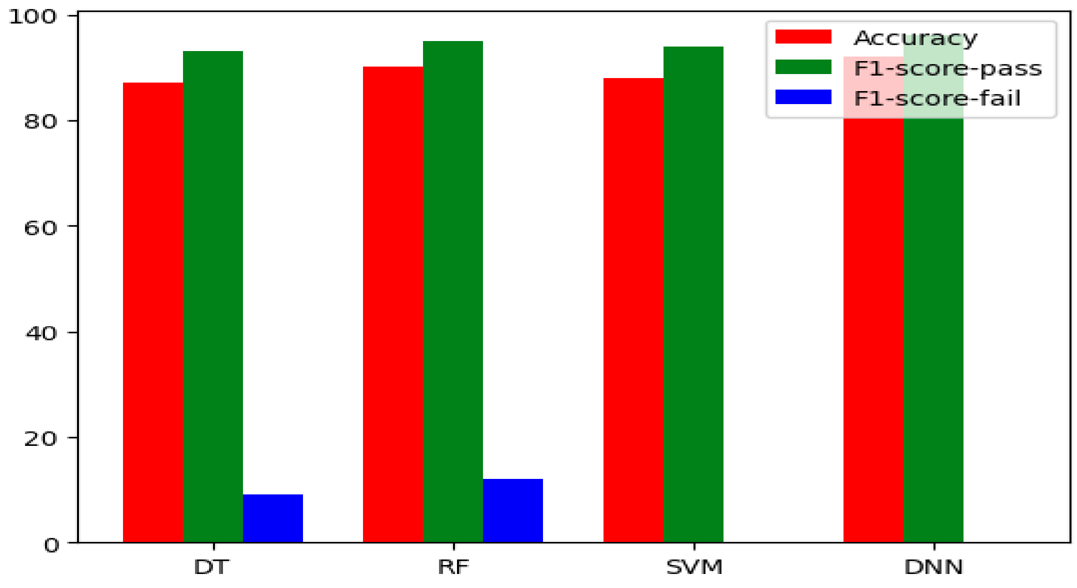


FIGURE 2.4.1. *Performance Evaluation on the Imbalanced Dataset(Based on Experiments of Pre-thesis).*

2.4.1. SMOTE

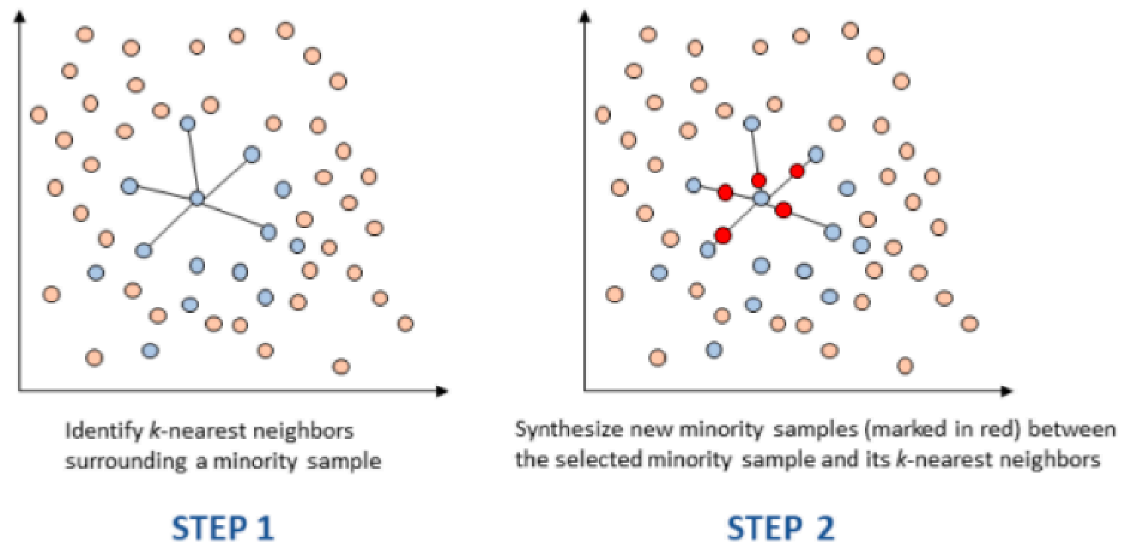
Synthetic Minority Oversampling Technique (SMOTE) is a technique based on the theoretical statistical field for increasing the data in your dataset to accomplish balanced goals. In SMOTE, extra data can be created based on the operation of the training data, rather than over-sampling with replacement. In general, new synthetic data will be created between the two minority data since SMOTE creates synthetic data depending on the distance between the minority data and the closest (nearest) minority data[11]. The formula of SMOTE can be expressed as following:

$$X_{new} = X_i + (\widehat{X_i} - X_i)\delta$$

Where:

- X_{new} : Synthetic data
- X_i : examples from the minority class
- $\widehat{X_i}$: one of the K- nearest neighbor from D_i
- δ : random number between 0 and 1

Although SMOTE is quite effective at increasing the classification accuracy of the minority class, there are still issues, among them the prevalence of overgeneralization. One of these problems is the performance of the classifier that will be affected since data from SMOTE can still spread to both minority and majority data.



Identify *k*-nearest neighbors surrounding a minority sample
STEP 1
Synthesize new minority samples (marked in red) between the selected minority sample and its *k*-nearest neighbors
STEP 2

FIGURE 2.4.2. *SMOTE resampling technique.*

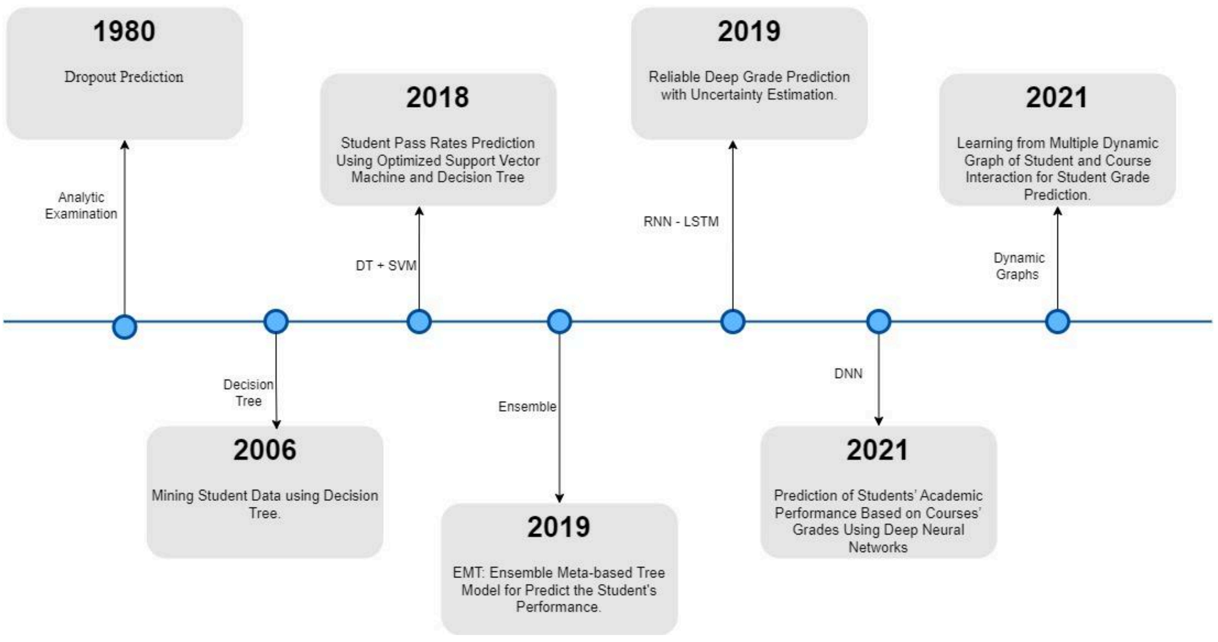
2.4.2. ROS

Random Over Sampling (ROS) is a popularly used technique to deal with the imbalance datasets. The main concept of ROS is to replicate instances labeled as minority class randomly. This process continues until the number of minorities matches with the majority. Similar to SMOTE, it helps the model improve the overall accuracy by providing the researcher more minority instances. However, same value instances by ROS can affect the model with the overfitting phenomenon. In addition, it will make training time increase, because of the larger dataset.

2.5. Related Work

From the past to now, education has taken an important position in the development of a nation and also a crucial tool for success in the life of a person. According to the Education Data Initiative's report, there are 40 million Americans who left the university's program in July of 2021, and just 864,824 of them re-enrolled. In 2019, Vietnam had 30% students per year leaving the program of university to transfer to low- level education, such as intermediate school,.. and the percentage of students graduating on time is only 50% and with an expired term is from 70% to 72% of all students enrolled. Since, school performance and pass rates are indicators of the standards of the teachers and the entire educational system[13]. Universities try to improve and provide a better learning process day by day. Therefore, how to prevent the dropout and improve the student's performance has now become a research hotpot in the field of educational data mining. There are many concepts of building a course-specific models for predicting the academic grade and dropout rate, such as using the demographic or academic features, using traditional supervised algorithms, deep learning algorithms, sequence- based model or even unsupervised learning. In the early 1980s, universities started to make surveys from the examination analytic to predict the dropout rate.

These researches were developed to predict models that can give them acceptable results based on not only the academic features, but also the demographic features. Along with the multi-dimensionality of predicting's context, the algorithms used for these also were improved to be suitable. In addition, in 2015, with the development of the Learning management system(LMS) and Massive open online course (MOOC), educational data tend to expand from normal academic features (final grades, partial grades…) to features related to the interaction of students with the course's materials or how much their attention on Learning management system. These availability and trustworthy characteristics of data motivates researchers to have multi-dimensions predicting models. We previously covered some important main ideas relevant to build a grade and dropout predicting model, hence, we will touch on some specific works briefly.



1980
Dropout Prediction
Analytic Examination
Decision Tree
2006
Mining Student Data using Decision Tree.
DT + SVM
2018
Student Pass Rates Prediction Using Optimized Support Vector Machine and Decision Tree
Ensemble
2019
EMT: Ensemble Meta-based Tree Model for Predict the Student's Performance.
RNN - LSTM
2019
Reliable Deep Grade Prediction with Uncertainty Estimation.
DNN
2021
Prediction of Students' Academic Performance Based on Courses' Grades Using Deep Neural Networks
Dynamic Graphs
2021
Learning from Multiple Dynamic Graph of Student and Course Interaction for Student Grade Prediction.

FIGURE 2.5. *Learning Models for Predicting Student's Grade and Performance.*

GPA and the earlier course's grades are the most common indicator to evaluate the achievement of students at the university level, such as [1], [2] and [13]. This approach is an easier accessibility solution for all the colleges when the only data they have is the prior course grades. Several researchers have used the earlier grades or academic achievements of students to predict the pass and fail rate of future courses. In [2], the authors compared the performance of models with traditional machine learning techniques such as Decision Tree (DT), Random Forest(RF), Support Vector Machine(SVM) and modern algorithm, deep neural network (DNN). The other mentioned problem in this research is imbalance of the dataset, which the authors carried out some experiments with various techniques, such as SMOTE , ROS, or ADASYN to deal with in the data level and have comparisons between them. Through these, [2] points out the effect of an imbalance dataset on the model performance and what is the suitable technique to deal with this. Similar to [2], the difference is that [1] transformed the earlier courses' grade which the methodology of [2] treats as the static vector to the matrix representing the temporal dynamic of student's knowledge evolution. Recurrent neural network was applied to achieve this problem. [1] concludes that this learning model has better capability at predicting the academic grade with time-series data than the state-of - the- art baseline. In [13], the authors observe that groups of students which have similar grades in the same course can help to enhance the grade predictive model. The [13] focuses on the similarity between courses and between students across the historical grade records as an indicator for student's profiles and course's properties. Firstly, Based on the dynamic graph technique, the researchers of [13] built three graphs like student - student and course - course from calculating the similarity and student-course graph of interaction to get the dynamic interaction between them. Then, similarly to [1], [13] also focuses on the evolving trend of students and courses across multiple semesters using Dynamic Graph based Time Edge Aware Model (DGTEAM) which utilizes and generates the influences of the other students and courses on the specific student. From this brief, many essential information such as, student's information, family's information or demographic have been taken advantage of to build multi-factional predictive models. To find key factors impacting the student's performance and pass rates in Mathematic and Portuguese, the contributors of [14] used two common algorithms Decision Tree(DT) and Support Vector Machine(SVM) with the optimization of feature dependencies and grid search algorithm. Features used for this research are information related to the demographic, student's habit and the academic achievement. Combination between student's life and academic attributes has been continued in the [15], such as behaviors in class, parent school satisfaction and students' status. Ensemble Meta-based Tree (EMT) which combines the best knowledge discovery techniques and boosting into a predictive model with the same training set is covered in this research. The final result states that this EMT algorithm can enhance the accuracy up to 98.5 % which is outperformance with the predictive models based on tree architecture. In addition, along with a complex of algorithms and attributes, there are many researches that provide for us the knowledge to deal with existing problems in educational data mining such as imbalance of data or lack of availability data in [2], [16]. As mentioned above, the researchers of [2] carried out some experiments of combination between data level resampling techniques and machine learning algorithms to deal with imbalanced datasets. It concludes that ROS and SMOTE can give us reliable results based on many evaluation measures like F1-score, accuracy or confusion matrix. In another concept, the authors of [16] proposed an advanced algorithm to generate data from the available student's instances, namely Conditional Generative Adversarial Network (CGAN) which a condition vector will be imputed to oversample with specific information. There are two components of this condition GANs, namely, Generator and Discriminator based on neural networks. The difficulty of traditional up-sampling approaches is the sequence data which the timestamp characteristic has to incorporate along with the up-sampling instance. Every quarter, the suggested SC-GAN is applied to create data in such a way that the prior behavior of the created instances is also included in the generation process of the subsequent quarter. By doing this, the network is able to understand how the up-sampled instances behave throughout the course of all the quarters [16].

CHAPTER 3

METHODOLOGY

In this study, our approach is predicting the student's academic grade and dropout rate of the specific subject based on the previous course's grades. For example, predicting the grade of an Object oriented programming course from the score of English, Physics, Calculus, Core programming… From this, at-risk students can be identified at the early stage of semester and have the remedial or good preparations for the upcoming courses. With the lecturers, It can help to evaluate student's available knowledge to deliver schedules or programs that can be easier to access or suitable for students. There are some steps of knowledge discovery processes which have been used for our study such as Data Collection, Data- preprocess, Data-mining process and Performance evaluation….

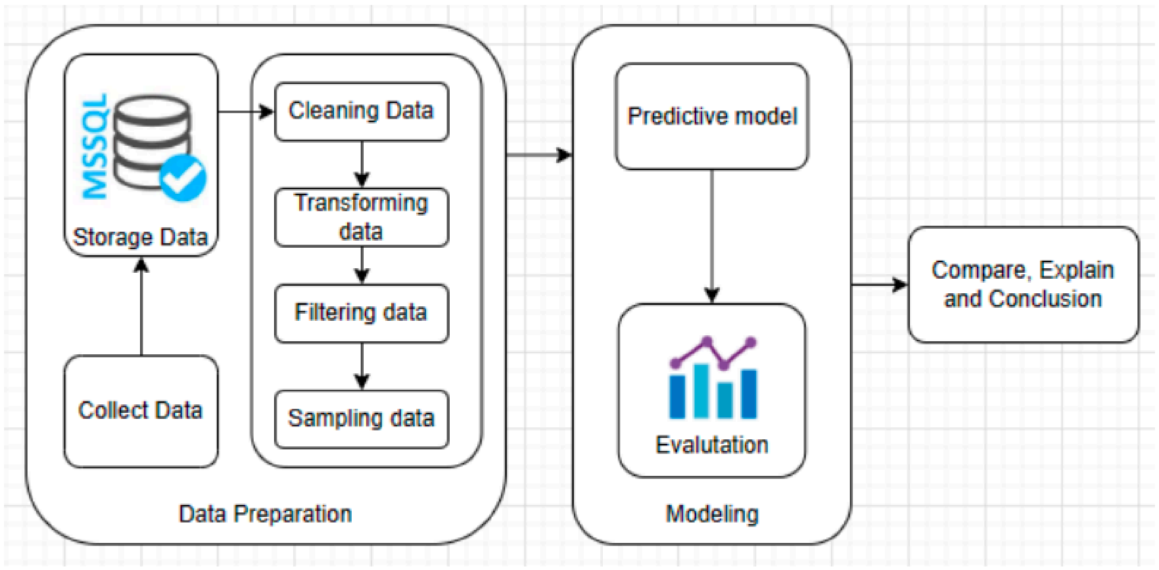


FIGURE 3.1. *Proposed Process to Build the Predictive Model.*

3.1. Data Collection

In this study, a real dataset of under-graduation students from the International University is used. Features of this dataset are academic grades of students in the previous courses. The

University gathered only academic features, and we tried to exploit these features to build a predictive model to be used by the university to reduce the number of failure rates. Academic features like previous results are available and popular data storage in most universities or institutions in various countries, hence, this work is considered crucial in order for them to utilize these[2]. To predict the result of the specific subject, we should get pre-requisite courses' grades or related outline courses for an upcoming course.

TABLE 3.1.1. *Dataset Description.*

Attribute	Type	Description or Values
S1	Numeric	Physics
S2	Numeric	Math
S3	Numeric	Core Programming
S4	Numeric	….
S5	Numeric	….
: :	Numeric	…..
Sn	Numeric	Name of required or outline subject
Target	Numeric	Predicted subject

3.2. Data preprocess

It is one of the most important steps in machine learning in which raw data can be converted to the suitable data format to deal with the errors of collected data. Having a good data-preprocess step can help for training models to achieve better results. There are some small processes in this step including data cleaning, data discretization, handling imbalanced data and feature scaling. These process will be described in detail for this study, as following:

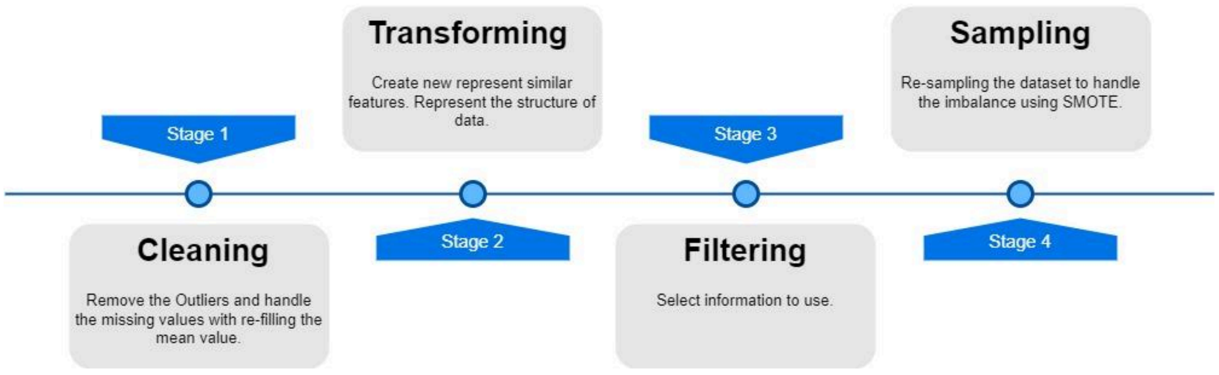

Transforming
Create new represent similar features. Represent the structure of data.
Sampling
Re-sampling the dataset to handle the imbalance using SMOTE.
Stage 1
Stage 3
Cleaning
Remove the Outliers and handle the missing values with re-filling the mean value.
Stage 2
Filtering
Select information to use.
Stage 4

FIGURE 3.2.1. *Data pre-processing.*

3.2.1. Data Cleaning

There are some missing and error values in our real data that will destroy the predicted performance. This student that has five missing course's grades will be removed from the dataset. Therefore, we will remove all the outliers that harm the dataset. In contrast, the acceptable missing value will be re-filled by the mean value of course.

3.2.2. Data Transforming

Today, many universities convert student's grades from numerical form to the letter form including A, B, C, D or F. If the dataset does not contain the letter grade, we will use this stage to convert the student's target grades in the numerical to the nominal values to present our dataset as the classification problem. There are three class labels for the grade level prediction, including **Good, Average and Fail** (Table 3.2.1).

And, for the dropout rate or at risk of leaving the course early, we will label based on the partial scores of a course. The partial scores represent the knowledge accumulation process of a student through class quizzes, assignments, midterm examination and final examination. In detail, a student, who cannot get average grades in these partial scores, will be labeled as the risk of dropout or leaving the course early. There are two class labels for the dropout rate, including **Risk and Not-risk** (Table 3.2.2).

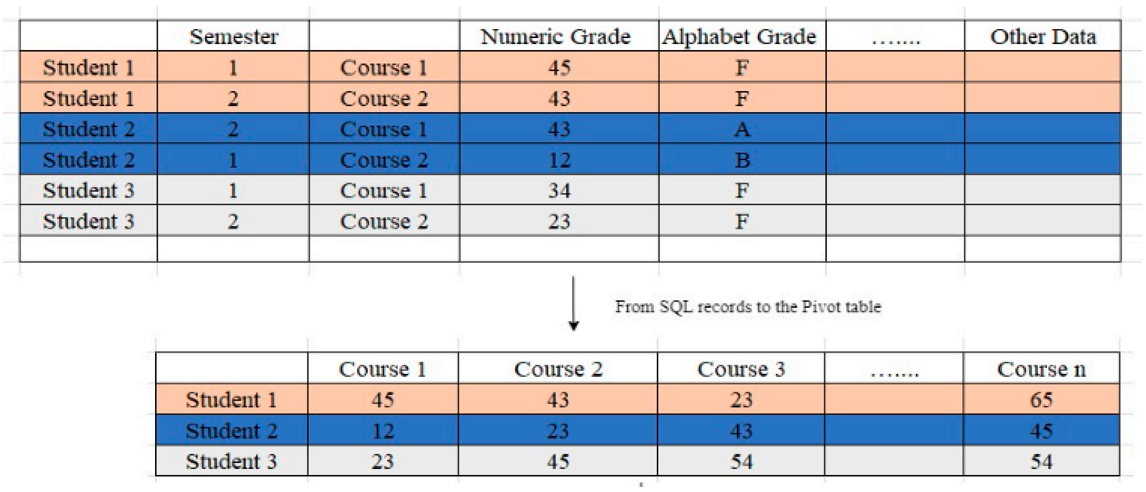
TABLE 3.2.1. *Grade Prediction Data Discretization Step.*

Nominal interval	Numerical value	Pass or Fail
Good	>= 80	Pass
Average	50 - 80	Pass
Fail	< 50	Fail

TABLE 3.2.2. *Dropout Prediction Data Discretization Step.*

Nominal interval	Numerical value	
Risk of Dropout	Partial Scores < 50	
Not Risk	Partial Scores > 50	

The different learning models require various types of input patterns. Therefore, transforming data to the required pattern is a re-prerequisite task in the process of building or training any data mining models. Historical records of students are often stored as a joined table of many - to - many relationships between students and courses (Fig. 3.2.2). In this concept of using the previous course's grades to predict the upcoming result, the inputs of traditional machine learning techniques are formed as the static vectors stored in the pivot table in which the rows represent the amount of records and each column is an instance for predicting. As mentioned above, these vectors cannot learn the latent temporal dynamic information of students. Hence, for this proposed model, grade of a semester will be encoded into the multiple hot encoding vectors in which course taking in semester t will be filled and the remaining is 0. The sequence of these vectors input to course-specific models to hint the knowledge evolving trend of students which is accumulated across multiple semesters, the detail will be mentioned in the later parts.



	Semester		Numeric Grade	Alphabet Grade	…….	Other Data
Student 1	1	Course 1	45	F		
Student 1	2	Course 2	43	F		
Student 2	2	Course 1	43	A		
Student 2	1	Course 2	12	B		
Student 3	1	Course 1	34	F		
Student 3	2	Course 2	23	F		

From SQL records to the Pivot table

	Course 1	Course 2	Course 3	…….	Course n
Student 1	45	43	23		65
Student 2	12	23	43		45
Student 3	23	45	54		54

FIGURE 3.2.2. *Data Transforming with the SQL joined table and pivot table.*

3.2.3. Handling Imbalanced Dataset

Findings from EDM-based innovations usually originate from a predictive model that forecasts categorical target outcomes (e.g., passing or failing a course) based on a collection of meaningful factors, including students' activity data, formative assessment scores, and effective time use[17][18]. However, class imbalance in the target variables is an common issue that has a detrimental effect on the accuracy of the predictive models in the EDM. For example, The ratio between drop-out students and those who complete their courses is imbalanced. To deal with this problem, researchers have processed and extracted information from data with a substantially skewed distribution using imbalance learning techniques[19]. Instead of ignoring the minority classes, these techniques can help models have the outcome variable with more realistic accuracy. In this study, we will apply a common technique to solve the imbalanced problem , namely, SMOTE, which is mentioned in the above chapter.

3.3. Model Validation

In machine learning, as the model is fitting with the training data, we could not say that this can perform accurately with the actual data. To achieve this, we need to make sure that the model has the correct pattern from the data, and it does not produce excessive noise.

Therefore, Cross-validation is a common technique for dealing with these problems. By using multiple validation datasets to evaluate the model, Cross - validation can help us to prevent overfitting phenomenon that the model can perform too well with the training dataset, but have poor performance with the unseen or newly data. There are various concepts of Cross - validation such as k-fold cross validation, leave-one-out cross validation, stratified cross validation and random hold-out method. A concept can be chosen based on the size of the dataset, or some specific requirements of modeling as well. For example, K-fold cross validation divides a data set into K nearly equal size, then (K − 1) folds are used for model training, and the remaining subset sample is used to test. In actuality, K can be chosen between 5 and 10. This procedure is occasionally repeated, and the average result of the testing set is used to determine the model's final outcome[2][20]. As a result, it is a helpful tool for limited datasets because it uses the complete dataset for testing and training. Additionally, another approach is the random hold-out method, which divides our dataset into two partials according to a predetermined ratio (e.g., 80% for training and 20% for testing). The test set is kept separately while the training set is utilized for developing a predictive model from start to finish. This makes it easier for us to evaluate the model's potential performance realistically when applied to new data.

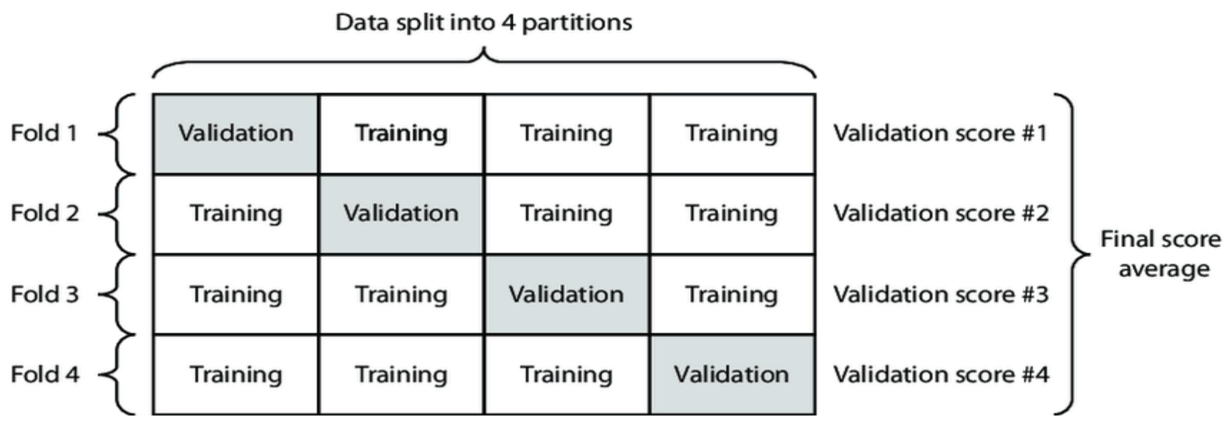


FIGURE 3.3.1. *K - fold cross validation method (From: researchgate.net).*

3.4. Machine Learning Model

As a result of previous study, we apply various traditional Machine Learning techniques, namely, Random Forest (RF), Decision Tree (DT) and Support Vector Machine (SVM) to create the predicting model. These machine learning methods are commonly used techniques to create a pass/ fail rate prediction in many research of Educational Data Mining. These predicting models have acceptable results with small/big datasets or imbalance/ balanced datasets and are evaluated as the suitable techniques for this problem according to some previous research in this field. In addition, a deep learning method called Deep Neural Network (DNN) will be applied to create the student performance predicting model. This technique is mentioned in the above chapter. This algorithm includes many layers and each layer is based on a processing unit called artificial neurons.

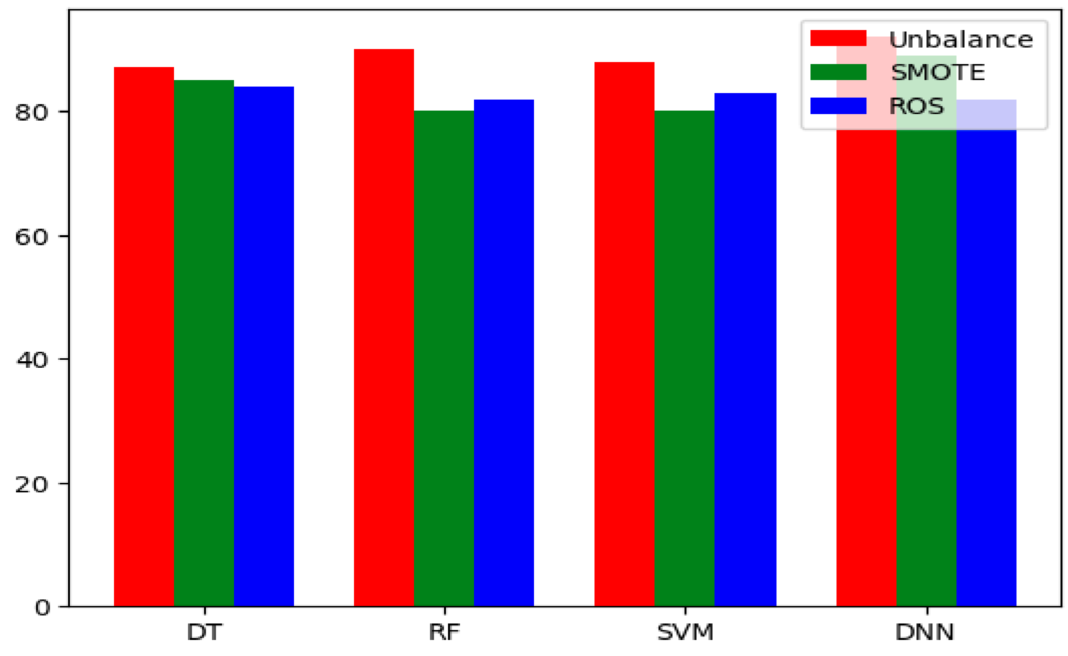


FIGURE 3.4.1. *Performance Evaluation based on Imbalanced and Balanced Datasets.*

(Based on the result of special study of field)

Based on this brief from the special study research, we expand the scope of this study to focus on the temporal and sequence characteristic of data. As mentioned above, the learning of students is a sequential process in which prerequisite or outlined knowledge can be delivered and updated over several semesters. On the other hand, enrolling in earlier courses can be stated as the preparation for the target one. And, the time distance between preparation and application tends to have a sizable impact on the student's result in the target course. For this assumption, we can observe a similar concept in the Natural Language Processing (NLP) that the context and meaning of sentences is analyzed and combined based on the position of the words. Consequently, we proposed a learning model based on the LSTM. In LSTM, the hidden state representing the student's knowledge evolution can be added and updated with the new information through the recurrent mechanism. To estimate the grade based on the course-specific LSTM model, we first extract the student's prior course grade from the tabular data (Fig 3.4.2) to the timestamp, in which semesters of prior courses are taken. Student can enroll more than one course in a semester, hence, the grade vector $x_{(t)}$ is represented as the multiple hot encoding vector (Fig.3.4.2). In this vector, courses taking in the semester t will be filled with their results and the remains will be 0. The input sequence $x_{(1)}, x_{(2)}..., x_{(t)}$ is fed to the model and the hidden state $h_{(t)}$ to the connected layers, in which the output is predicted grade y, as calculated at eq. 2.1.5 and eq. 2.1.6.

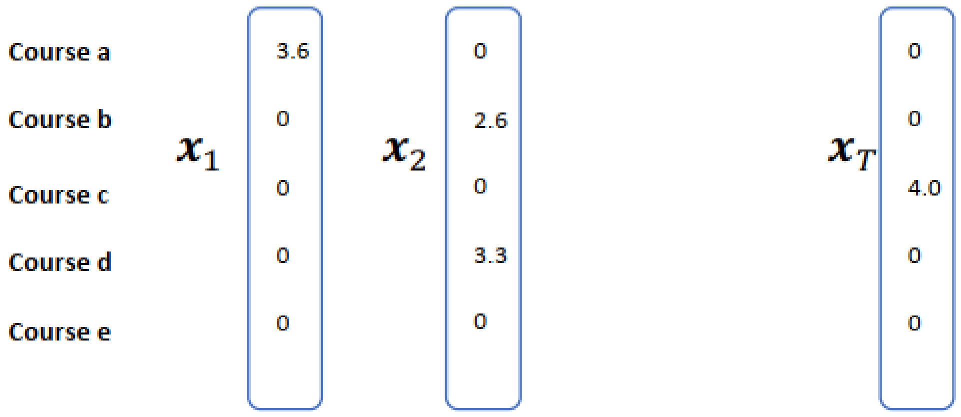


FIGURE 3.4.2. *The grade vector $x_{(t)}$ with multiple-hot encoding.*

Dropout Regularization is a simple technique that we apply for reducing the overfitting phenomenon. Because of containing multiple non-linear hidden layers, it is difficult for the model to learn the very complicated relationship between the inputs and outputs. The intuition behind this approach is that some units along with their connection will be ignored or dropped randomly. Applying the dropout technique to the neural network amounts to sampling the 'thinned' network from a complex network. Thinned network consists of all connections surviving after the dropout process. Therefore, during the training, a neural network can be judged as a collection of thinned networks with extensive weight sharing, where each thinned network gets trained very rarely, if at all[21]. After the training process, smaller networks with shared weight will be combined into a single neural network used for the test evaluation through a specific scaling technique .

3.5. Model Interpretability

When a model is used for decision making, it is necessary for the practitioners to be confident in the predictions in order to act upon them[1]. Most of the models often are evaluated based on the accuracy of the available validation dataset. In contrast, the real-world data that is significantly different may not be indicative of the model's goal via the metric of evaluation accuracy. Apart from these measurements, examining individual forecasts and their justifications is a valuable solution. Therefore, determining trust in individual predictions is important when the model used for decision making[22]. From this theory, when we predict the grade of a target course, the reason associated with this prediction can be a strong explanation to emphasize which trusting level this model can reach. In this particular case, the reason is the prior courses can lead to a lower grade or failure in the target course. To achieve this approach, we develop the explanation system for this proposal model in which we assume that all knowledge accumulated in the earlier stages has a sizable impact on the result of the target course. Based on the related work et al [1], we can implement the interpretability system by calculating the influence rate of the prior courses. The scenario of this system is the higher prior courses' grade, the better performance in target one. If we increase an earlier course's grade by a fixed value, we will calculate the difference between performance after and before. From this, we can define what course can affect the target result of students according to the order of influential or important rate. In practice, we can calculate as follow:

- Given the trained model M and a student s. The predicted grade of this student is denoted as $\widehat{y}_s$. Assume that p is the prior course and $\widehat{y(\neg p)}_s$ is the grade predicted by the model via increasing the earlier course's grade by a fixed value. The difference of both predictions can be understood the influential of course c of a student, denoted as $I(s, c)$:

$$I(s,\ c)\ = \widehat{y(\neg p)}_s\ - \widehat{y}_s \qquad (3.5.1)$$

$$\text{and} \quad I * (c)\ =\ \sum_{s \in S} I(s, c) \qquad (3.5.2)$$

S represents all of the students in the dataset. Not only providing an understandable explanation for the learning model, the applicability can be an important aspect which this implementation will serve. Depending on the result, the lecturers could deliver a better syllabus for a course and students could be advised to prepare and review the materials for these influenced courses to be successful in the target course [1].

CHAPTER 4
EXPERIMENTS AND RESULTS

4.1. Experiments

A model which can help the lecturers or students to predict the academic grade and student at risk of failure of a specific upcoming course depending on previous scores will be created. It helps the lecturer have a clear statistical picture about the knowledge of course's participants to improve their lessons and syllabus. In addition, students who are predicted a risk of failure early, can have more revision or remediation to improve their academic achievement for the final exam. In this study, proposal models are implemented with Keras and Tensorflow based on the Python programming language. Tensorflow is an end-to-end deep learning open source framework that was developed and released by Google from 2015. Tensorflow provides many types of abstraction levels for building and training models, or scalable productions and deployment as well. This symbolic math library is widely applied across multiple ranges of tasks of dataflow programming. Keras is a popular deep learning application programming interface (API) that is built on top of famous frameworks such as Tensorflow, Theano… This does not focus on the low -level computationals, instead it calls to other deeply mathematical frameworks as the backend. Keras was adopted and integrated into Tensorflow from 2017. This combination provides a user-friendly and extensible module for easily building and training models that users can access via the tf.keras.

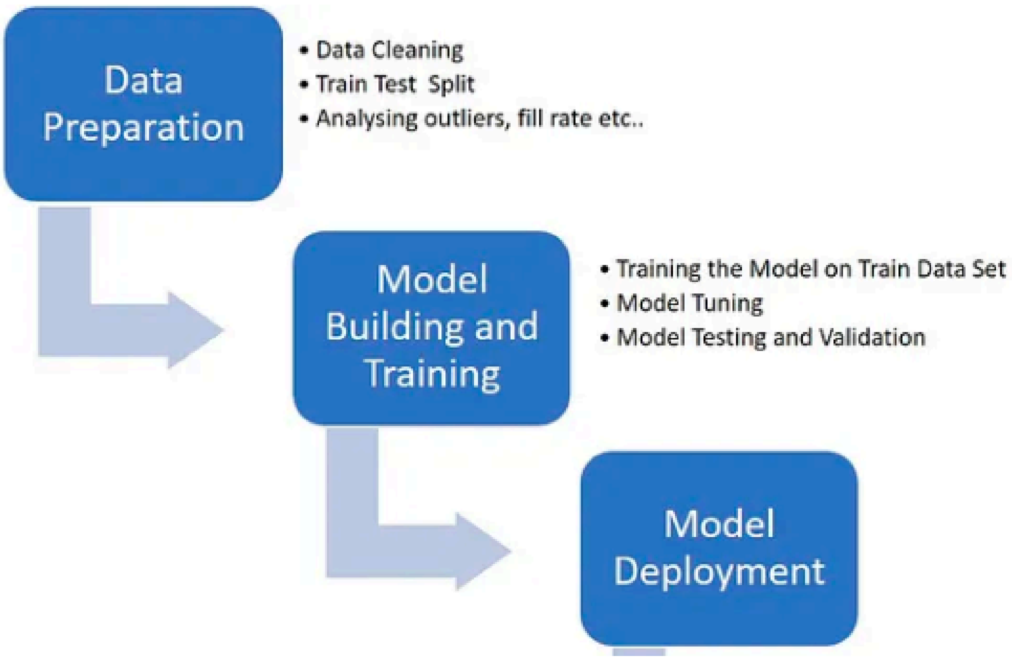
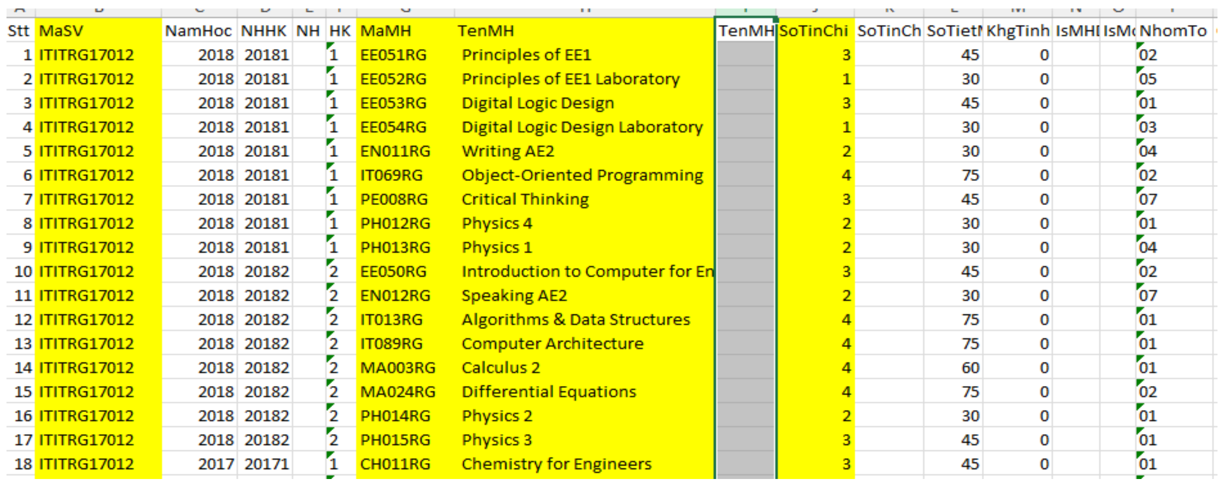

Data Preparation
• Data Cleaning
• Train Test Split
• Analysing outliers, fill rate etc..
Model Building and Training
• Training the Model on Train Data Set
• Model Tuning
• Model Testing and Validation
Model Deployment

FIGURE 4.1. *Processes of building predictive models.*

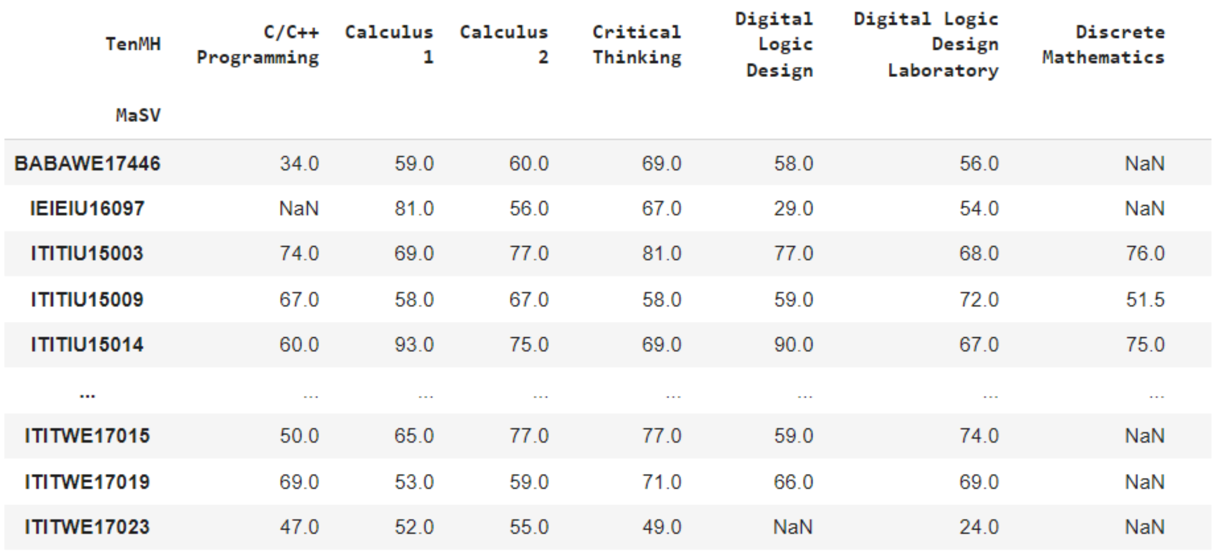
4.1.1. Dataset Description

In this research, we will use the data of student's scores from International University, this data is stored as SQL records (Fig 4.1.1). The row represents a student's finished course with student's ID, course's name, course's ID, score in 100, score in 4.0, average score,... Therefore, to use it as a dataset to train models, we convert data as records to a pivot table with course's name as column and score in 100 scale as value of column.



Stt	MaSV	NamHoc	NHHK	NH	HK	MaMH	TenMH	TenMH	SoTinChi	SoTinCh	SoTietI	KhgTinh	IsMHI	IsM(	NhomTo
1	ITITRG17012	2018	20181		1	EE051RG	Principles of EE1		3		45	0			02
2	ITITRG17012	2018	20181		1	EE052RG	Principles of EE1 Laboratory		1		30	0			05
3	ITITRG17012	2018	20181		1	EE053RG	Digital Logic Design		3		45	0			01
4	ITITRG17012	2018	20181		1	EE054RG	Digital Logic Design Laboratory		1		30	0			03
5	ITITRG17012	2018	20181		1	EN011RG	Writing AE2		2		30	0			04
6	ITITRG17012	2018	20181		1	IT069RG	Object-Oriented Programming		4		75	0			02
7	ITITRG17012	2018	20181		1	PE008RG	Critical Thinking		3		45	0			07
8	ITITRG17012	2018	20181		1	PH012RG	Physics 4		2		30	0			01
9	ITITRG17012	2018	20181		1	PH013RG	Physics 1		2		30	0			04
10	ITITRG17012	2018	20182		2	EE050RG	Introduction to Computer for En		3		45	0			02
11	ITITRG17012	2018	20182		2	EN012RG	Speaking AE2		2		30	0			07
12	ITITRG17012	2018	20182		2	IT013RG	Algorithms & Data Structures		4		75	0			01
13	ITITRG17012	2018	20182		2	IT089RG	Computer Architecture		4		75	0			01
14	ITITRG17012	2018	20182		2	MA003RG	Calculus 2		4		60	0			01
15	ITITRG17012	2018	20182		2	MA024RG	Differential Equations		4		75	0			02
16	ITITRG17012	2018	20182		2	PH014RG	Physics 2		2		30	0			01
17	ITITRG17012	2018	20182		2	PH015RG	Physics 3		3		45	0			01
18	ITITRG17012	2017	20171		1	CH011RG	Chemistry for Engineers		3		45	0			01

FIGURE 4.1.1. *The data is stored as SQL records.*



TenMH	C/C++ Programming	Calculus 1	Calculus 2	Critical Thinking	Digital Logic Design	Digital Logic Design Laboratory	Discrete Mathematics
MaSV							
BABAWE17446	34.0	59.0	60.0	69.0	58.0	56.0	NaN
IEIEIU16097	NaN	81.0	56.0	67.0	29.0	54.0	NaN
ITITIU15003	74.0	69.0	77.0	81.0	77.0	68.0	76.0
ITITIU15009	67.0	58.0	67.0	58.0	59.0	72.0	51.5
ITITIU15014	60.0	93.0	75.0	69.0	90.0	67.0	75.0
...	...	...	...	...	...	...	...
ITITWE17015	50.0	65.0	77.0	77.0	59.0	74.0	NaN
ITITWE17019	69.0	53.0	59.0	71.0	66.0	69.0	NaN
ITITWE17023	47.0	52.0	55.0	49.0	NaN	24.0	NaN

FIGURE 4.1.2. *Pivot Table.*

In this case, we choose the ‘Data Structures and Algorithms’ (DSA) as the target course to build course- specific models. As mentioned above, After that, 11 prerequisite, outline or related courses are used as the features for these models. A number of courses ,which a student finished, is not fully saved (e.g: student joins the DSA courses, but cannot find any records about required or outline subjects of this student). This dataset has 163 students joining the ‘Data Structures and Algorithms’ course and have acceptable or trustworthy records about related subjects.

TABLE 4.1.1. *Feature Description for predicting DSA course’s pass/fail rate.*

Features	Type
C/C++ Programming	Numeric
Calculus 1	Numeric
Calculus 2	Numeric
Critical Thinking	Numeric
Digital Logic Design	Numeric
Digital Logic Design Laboratory	Numeric
Discrete Mathematics	Numeric

Introduction to Computing	Numeric
Object-Oriented Programming	Numeric
Physics 1	Numeric
Writing AE1	Numeric

4.2. Results

There are some import tasks in this study, including: handling imbalanced dataset, implementing the course-specific model using the Long Short Term Memory (LSTM) and Applying this to the friendly user interface. Result of these tasks will be present in detailed, as following:

4.2.1. Result 1: Handling imbalanced dataset

In this task, the SMOTE technique mentioned above to deal with. According to the methodology, there are three labels of this predictive model and the amount of these has a significant difference. Therefore, applying a raw dataset can give us high accuracy, but most students tend to be predicted to the majority class. When we apply the SMOTE techniques to handle this problem, the number of students in each class will be equal, as 191 students for three levels: Fail(0.0), Average(1.0) and Good(2.0). Similarly, SMOTE will be applied for the dropout rate prediction and, the number of students are also equal for each class.

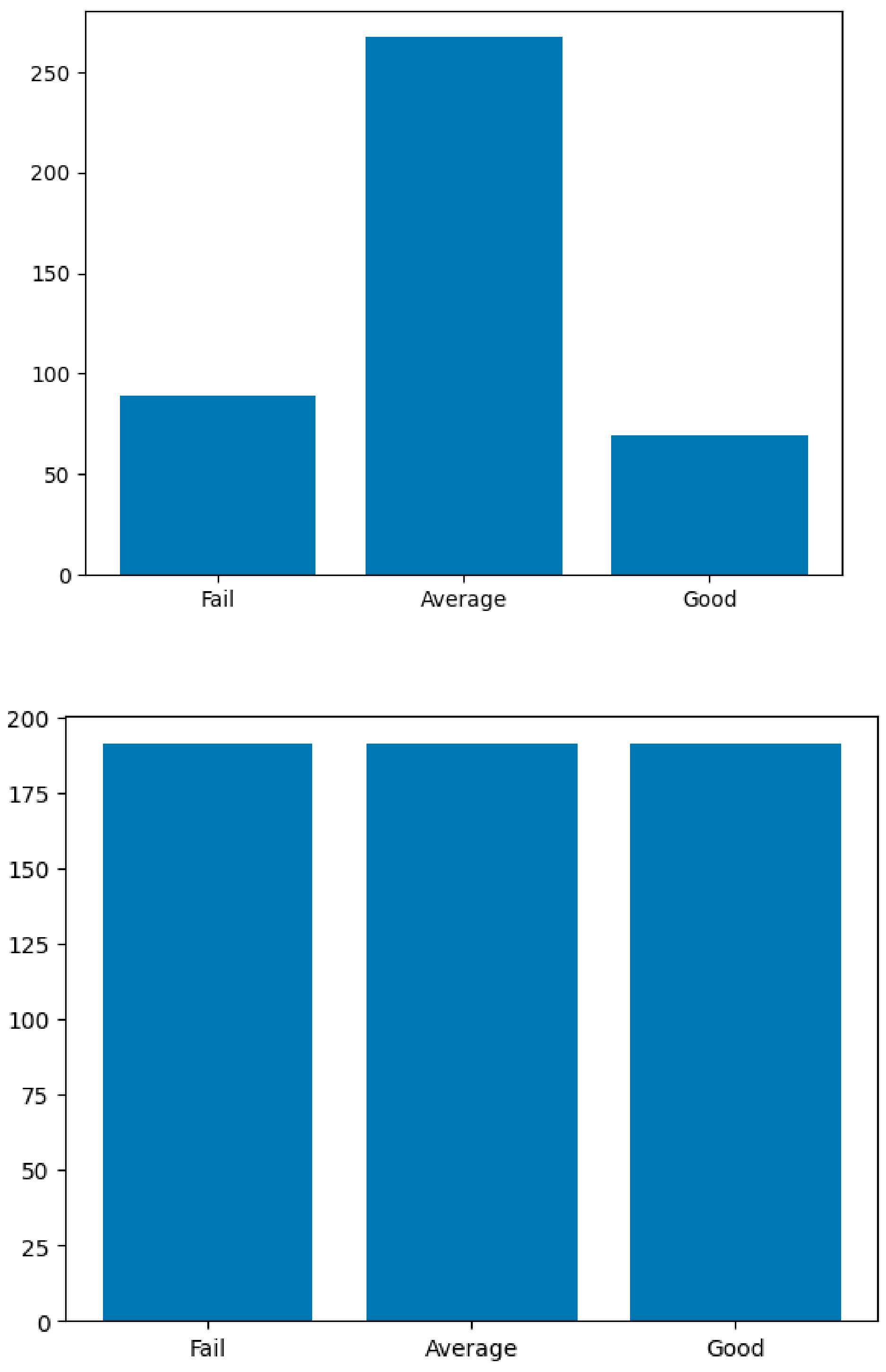
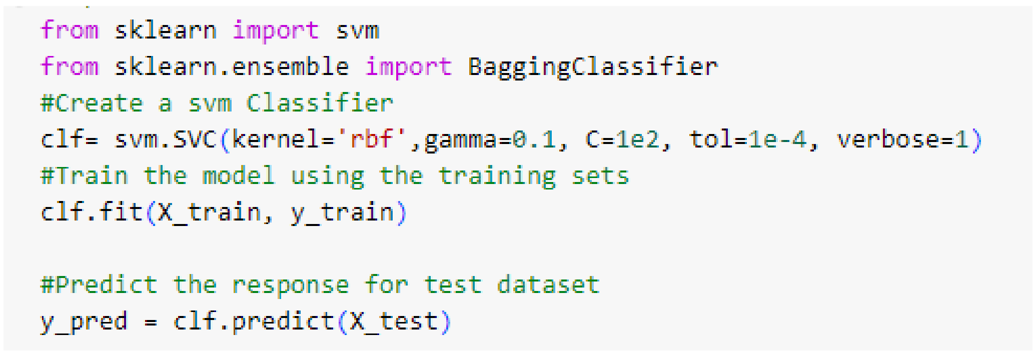


FIGURE 4.2.1. *Handle The Imbalanced Dataset problems for the grade prediction.*

4.2.2. Result 2: Training and Building Model for Dropout Prediction

For the dropout or leaving a course early, a traditional machine learning technique, namely Support Vector Machine will be implemented based on the tabular data of 11 previous pre-requisite and outline courses. After that, we will use various types of measures, including: Accuracy, F1-scores, Recall and the confusion matrix to preferences and have detailed analysis. The dataset will be splitted into two separated parts including 80% for the training process and 20% for the testing. The overall results, this predictive model can reach 81% of accuracy and 90.06% and 80.09% for the Precision and Recall. In the confusion matrix, there are 18 samples being False predictions and the remaining are true predictions. As mentioned above, with the brief experiment of previous pre - thesis work, a support vector machine can give us an acceptable and reliable result.



```python
from sklearn import svm
from sklearn.ensemble import BaggingClassifier
#Create a svm Classifier
clf= svm.SVC(kernel='rbf',gamma=0.1, C=1e2, tol=1e-4, verbose=1)
#Train the model using the training sets
clf.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

FIGURE 4.2.2. *Support Vector Machine Implementation .*

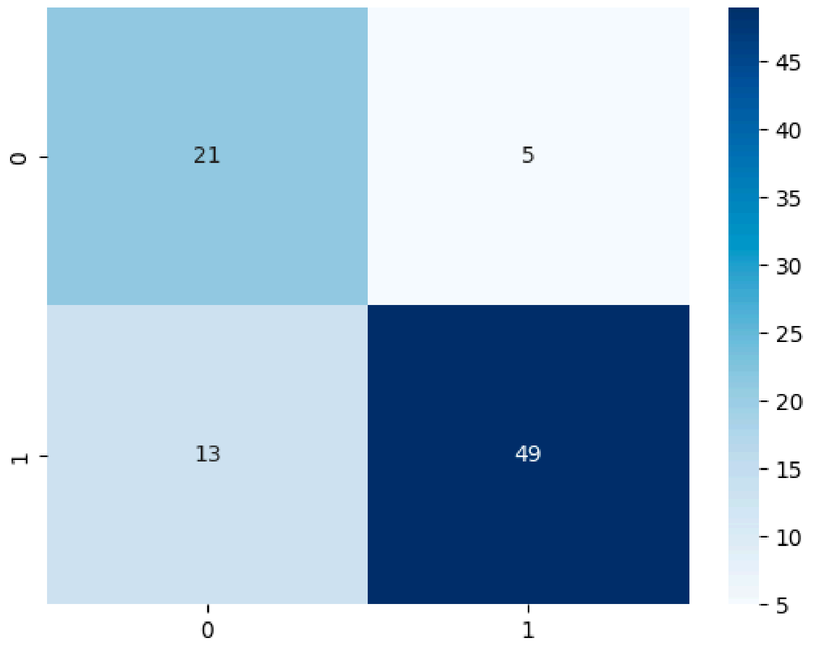
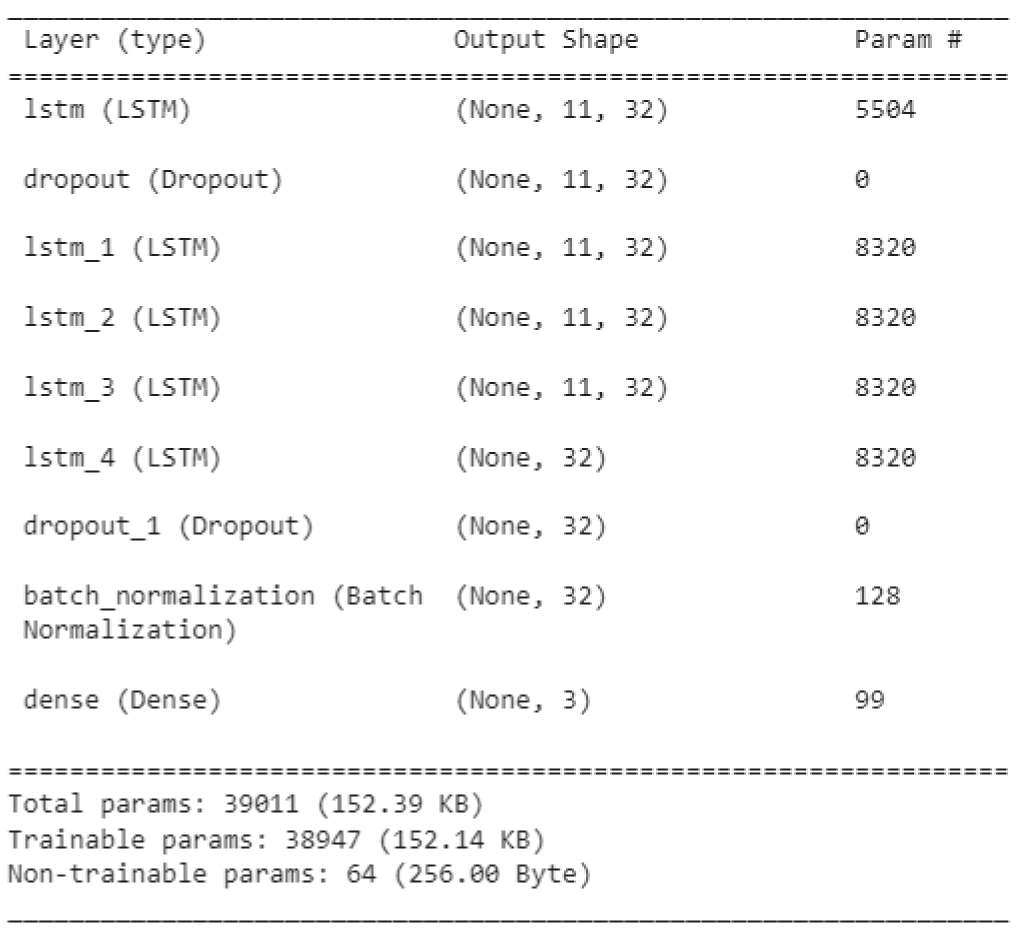


FIGURE 4.2.3. *Confusion Matrix for the Dropout prediction .*

4.2.3. Result 3: Training and Building Model for Grade Prediction

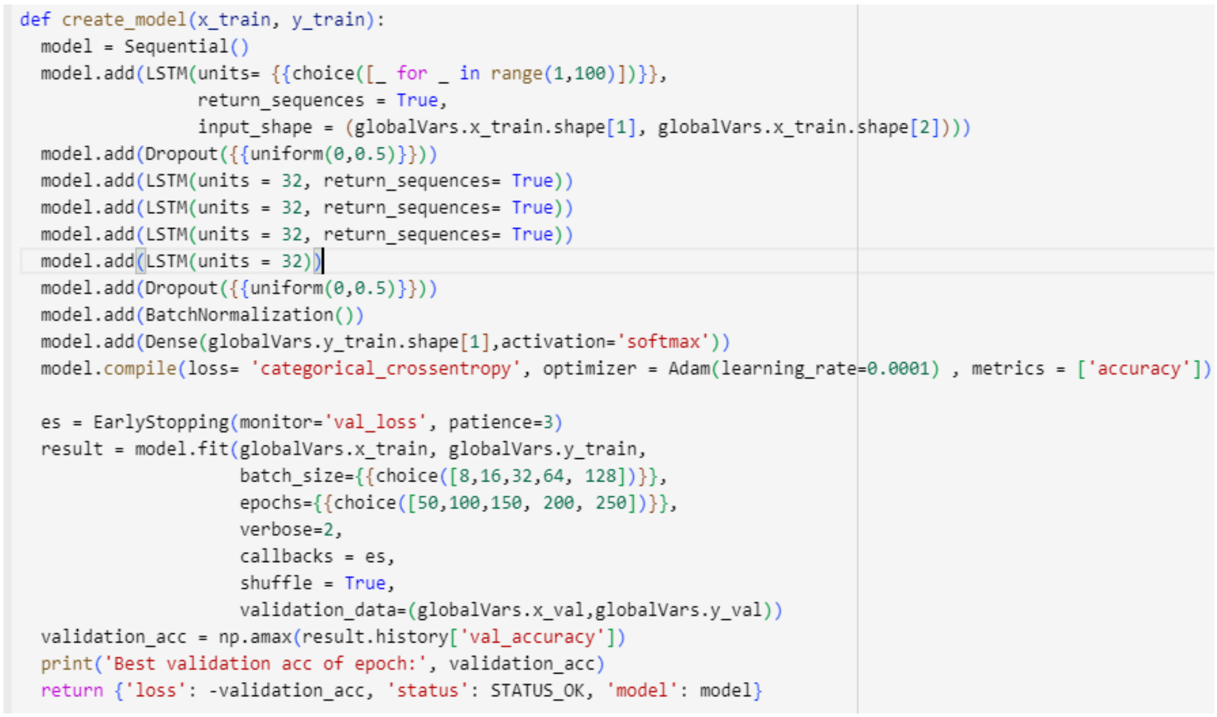
There are two evaluation measures, including: Accuracy and Confusion Matrix to evaluate or understand the performance of the models better. Following the information in chapter 3, accuracy might be a useful metric if the dataset has the same number of instances for each class. Otherwise, it cannot because the model tends to predict the value of the majority class following the information in chapter 3. Therefore, we use the accuracy as a preferences variable in this case and use the F1-score to deeply understand the model depending on analyzing the confusion matrix.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 11, 32)            5504

 dropout (Dropout)           (None, 11, 32)            0

 lstm_1 (LSTM)               (None, 11, 32)            8320

 lstm_2 (LSTM)               (None, 11, 32)            8320

 lstm_3 (LSTM)               (None, 11, 32)            8320

 lstm_4 (LSTM)               (None, 32)                8320

 dropout_1 (Dropout)         (None, 32)                0

 batch_normalization (Batch  (None, 32)                128
 Normalization)

 dense (Dense)               (None, 3)                 99

=================================================================
Total params: 39011 (152.39 KB)
Trainable params: 38947 (152.14 KB)
Non-trainable params: 64 (256.00 Byte)
_________________________________________________________________
```

FIGURE 4.2.4. *The Model Summary.*

Fig 4.2.2 describes the architecture of this course-specific model based on the Long - Short Term Memory (LSTM). In addition, This predictive model is built with the other techniques for the optimizations and preventing the overfitting phenomenon. Adam optimization algorithm which stands for the Adaptive Moment Estimation is an algorithm used for the minimize the loss function iteratively during the training neural network. Developed from 2014, Adam has become a go- to- choice optimization technique for many machine learning and deep learning researchers. An Essential task of Adam is choosing the initial learning rate for this algorithm, the default value is often 0.001 and popular in range 0 to 0.1. As previously stated, dropout reduces the neural network's capacity by ignoring some neutrons of a layer. Another regularization technique, called Early stopping, simply stops the training model as soon as there is no long update in loss of validation dataset. By doing this, the training process is able to terminate before overfitting manifests. Training or building an learning model is stated as the selection process of the collections variables and hyperparameters to reach an reliable and acceptable performance. Hyperparameter tuning is the process of carefully selecting these sets of hyperparameters either manually or automatically until the desired result is reached. A common hyperparameter tuning, called grid search, is used to find the ideal setting for our knowledge discovery model. The intuition behind this technique is to provide the range of values for the selected hyperparameters and do comprehensive search over all potential combinations. To get the desired outcome, this study's adjusted parameters for batch size, epoch, layer units, and dropout values. The detailed configuration of these parameters is as follow:



```
def create_model(x_train, y_train):
  model = Sequential()
  model.add(LSTM(units= {{choice([_ for _ in range(1,100)])}},
                return_sequences = True,
                input_shape = (globalVars.x_train.shape[1], globalVars.x_train.shape[2])))
  model.add(Dropout({{uniform(0,0.5)}}))
  model.add(LSTM(units = 32, return_sequences= True))
  model.add(LSTM(units = 32, return_sequences= True))
  model.add(LSTM(units = 32, return_sequences= True))
  model.add(LSTM(units = 32))
  model.add(Dropout({{uniform(0,0.5)}}))
  model.add(BatchNormalization())
  model.add(Dense(globalVars.y_train.shape[1],activation='softmax'))
  model.compile(loss= 'categorical_crossentropy', optimizer = Adam(learning_rate=0.0001) , metrics = ['accuracy'])

  es = EarlyStopping(monitor='val_loss', patience=3)
  result = model.fit(globalVars.x_train, globalVars.y_train,
                    batch_size={{choice([8,16,32,64, 128])}},
                    epochs={{choice([50,100,150, 200, 250])}},
                    verbose=2,
                    callbacks = es,
                    shuffle = True,
                    validation_data=(globalVars.x_val,globalVars.y_val))
  validation_acc = np.amax(result.history['val_accuracy'])
  print('Best validation acc of epoch:', validation_acc)
  return {'loss': -validation_acc, 'status': STATUS_OK, 'model': model}
```

FIGURE 4.2.3. *The Code for hyper parameter tuning using the hyperas in Python.*

TABLE 4.2.5. *The Parameters' configuration.*

	Parameters	**Value**
The First Layer	Units	32
	Dropout	0.142
The Second Layer	Units	32
	Return_sequence	True
The Third, Fourth Layer	Units	32
The Fifth Layer	Units	32
	Dropout	0.118
	Batch size	64
	Epoch	200
	Loss	categorical_crossentropy
	Activation	softmax
Adam Optimization	Learning Rate	0.0001
Early Stopping	Monitor	val_loss

	patience	3

Our dataset was divided with the ratio 70-15-15, 70 percent of the dataset for the training and the remaining for validation and testing. And, both training dataset and validation dataset are separated during the training model. Through the 175 epochs, our learning model's accuracy is about **84.12%** and the overall loss is **0.4858** (Fig 4.2.3). An observation during the training of this predictive model is sometimes the validation accuracy being greater than the training accuracy. From the Internet and many researches, we can explain this problem comes from using the dropout technique to reduce the complex course-specific model. As previously stated, dropout techniques ignore some units of layer to generate 'thinned' networks which is just valid on the training dataset. The model measurement on the validation dataset is based on the initial setting of the neural network in which the model is more robust and can lead to higher accuracy. Therefore, the phenomenon could be from the different behaviors between training time and validation time.

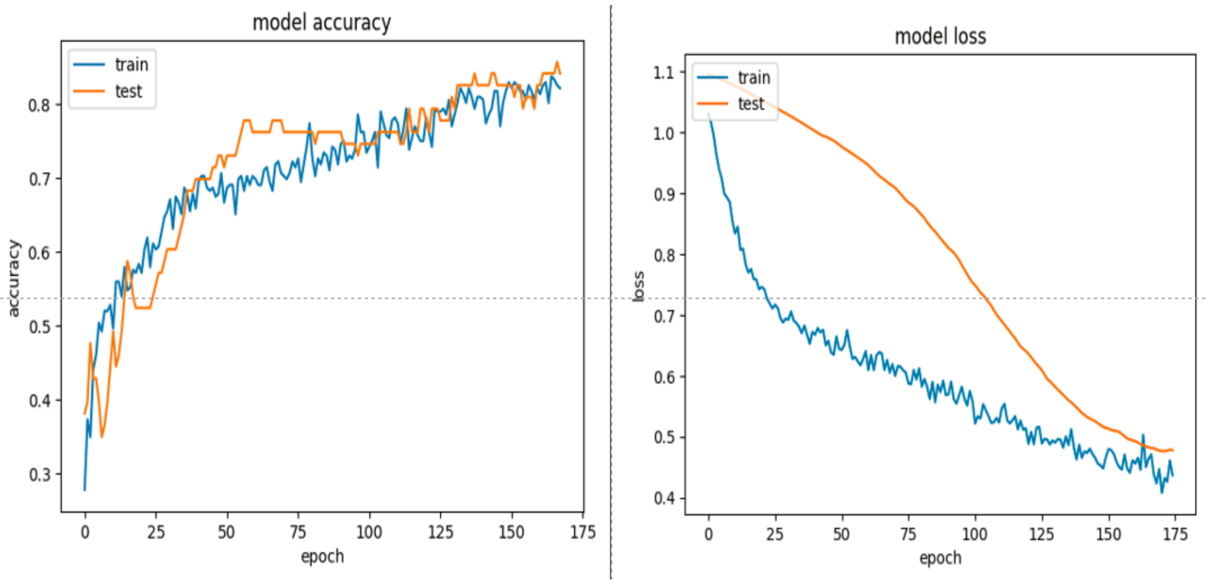


FIGURE 4.2.6. *The Loss and Accuracy of each epoch during the training process.*

For the approach of model explanation, we will apply a deep model measurement technique, called confusion matrix. It is easier for us to have a distinct visualization about how our model interacts with each class on the validation set. In this case, there are 63 students in the validation set in which 23 , 16 and 24 are in Fail, Average and Good, respectively. In the class of Fail, 21 students are predicted correctly and 2 of them are in the average class. The 'average' class has 12 accurate predictions and 4 students are in false classification. Moreover, the accurate labels of Good are 20 and fail labels are 4 students. Logically, we can observe that fail predictions are almost in the adjacent labels of actual. The middle label, the grade range from 50 - 80, the predictions can be misunderstood between the Fail and Good labels and the difference between actual and prediction of edge labels such as Fail and Good is quite small in this case. No student that has the result in the Good label is predicted in the Fail label and in contrast (Fig 4.2.4).

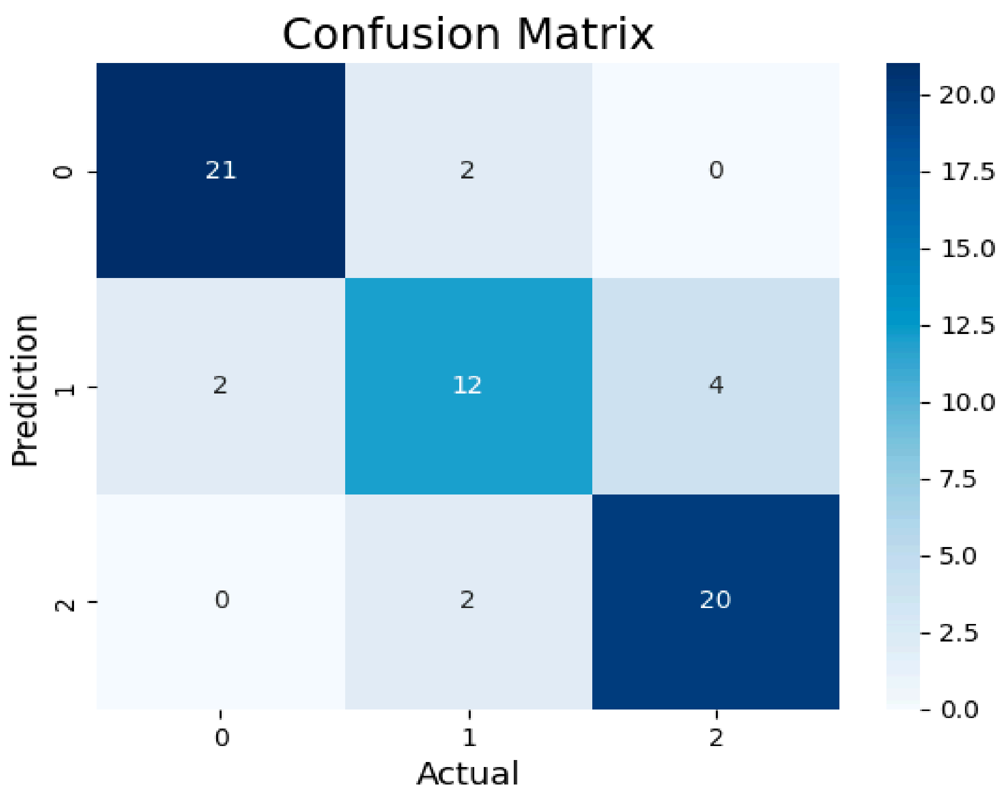


FIGURE 4.2.7. *The confusion matrix of this model on 63 students .*

As mentioned above, pointing out the influence of courses is another way to interpret the result of this model. This experiment highlights the conclusions drawn from the degree of trust model and the ways in which predictive models are used in academic settings. General assumption of this task is that the higher the achievement in the early course, the better the grade in the future course. The aim behind this experiment is to raise each course's grade by a fixed amount of 20, as previously mentioned in the Methodology chapter, and then use the trained model to predict the new outcome that corresponds with these increased scores. The grade difference between prediction of students in actual result and these in the new grade of each course represents how successful students were in. As a result, we can consider some of the definitions of what prerequisite knowledge should be accumulated to achieve a better outcome in the target course, as well as what constitutes a dropout or student quitting the course. From the outcome of **(Fig 4.2.5)**, we can explain that core programming and courses related to the mathematical field have a sizable impact on the student achievement in the target course, specific in Data Structures and Algorithms of this study. In the reality insight of the curriculum of International University for this target course, almost outlined or prerequisite courses are in the top 5 influential courses which are generated by this course-specific model. These are C/C++ Programming, Calculus 1, Calculus 2, Object- Oriented Programming, Digital Logic Design.

In summary, this course- specific model is not only evaluated based on accuracy or various types of measurements, but also interprets how this predictive technique is applied to the educational system. First, the understandable explanation is provided with the idea that the better performance in the past will translate into better outcomes in the future. Moreover, Not only does it satisfy ease of interpretation, the experiment helps us generate a smaller set of influential courses that have a sizable impact on the target course. This desired outcome motivates us in further analysis of a simple early warning system about dropout or leaving the course of students in the future(Fig 4.2.7).

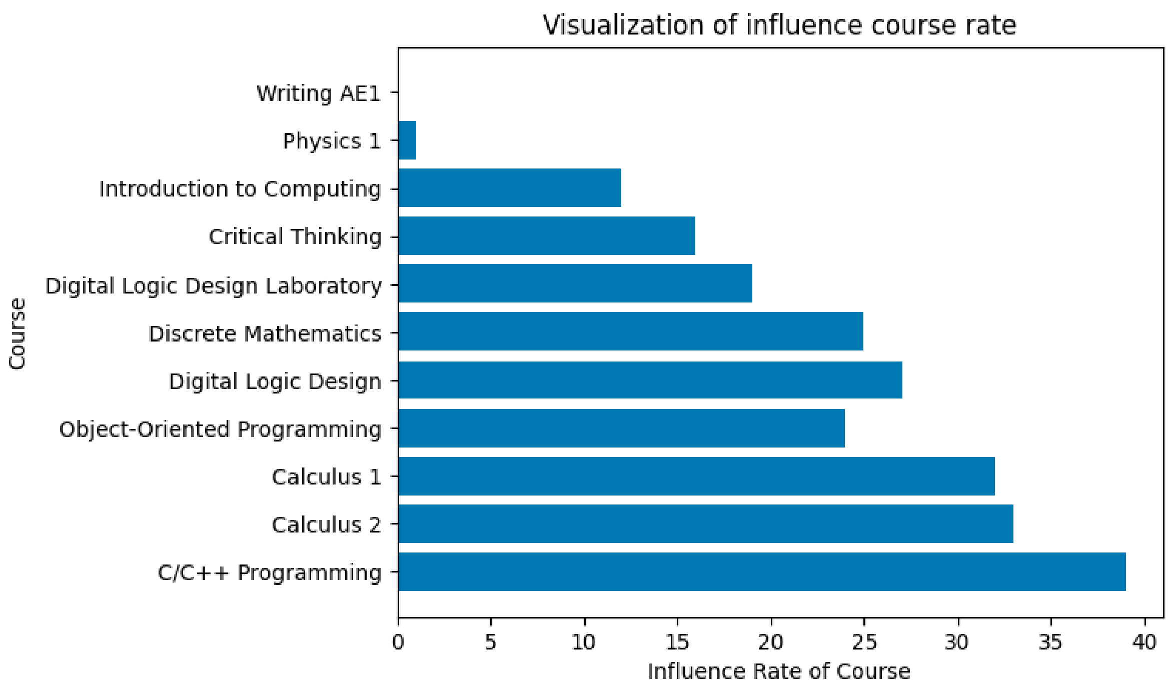


FIGURE 4.2.8. *The visualization of course's influence rate in the dataset .*

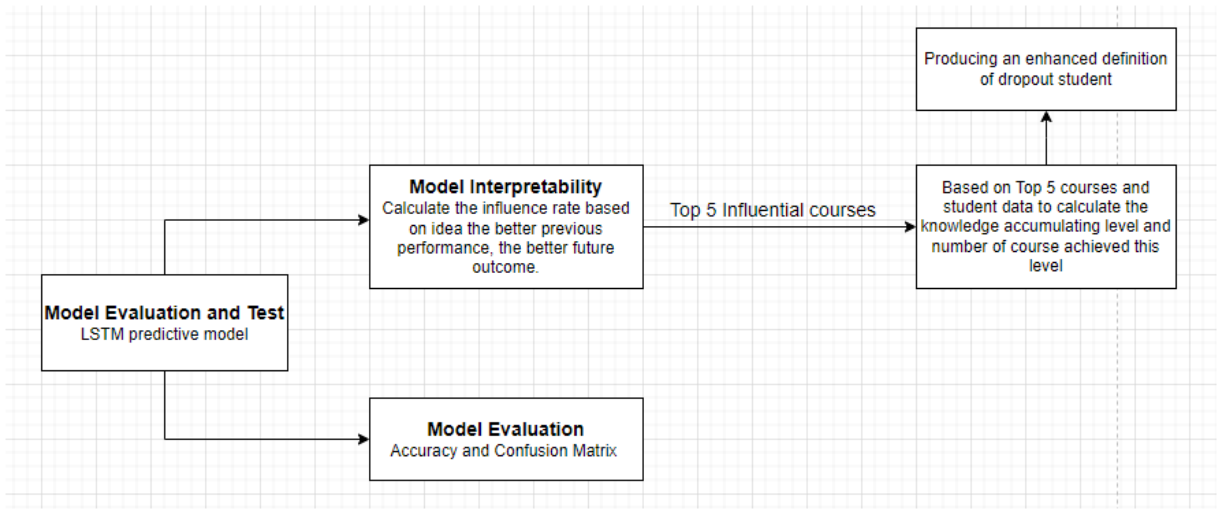
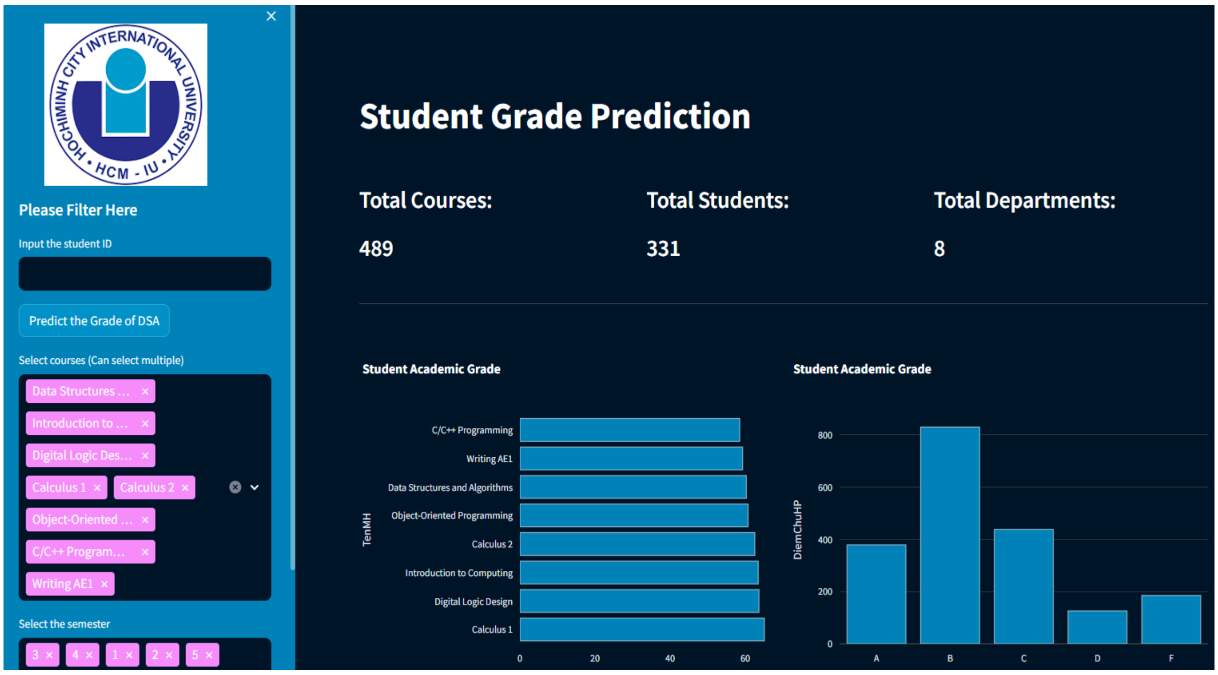

Producing an enhanced definition of dropout student
Model Interpretability
Calculate the influence rate based on idea the better previous performance, the better future outcome.
Top 5 Influential courses
Based on Top 5 courses and student data to calculate the knowledge accumulating level and number of course achieved this level
Model Evaluation and Test
LSTM predictive model
Model Evaluation
Accuracy and Confusion Matrix

FIGURE 4.2.9. *The summary of model Evaluation and Interpretation.*

The next step is to create the user interface so that we can release or make this public once we have a trustworthy prediction model. It is an essential part for both technical and non-technical users to interact with our model. This system can provide users with general insights of each student through charts of grade, GPAs…, as well as courses about the mean grade, correlations of multiple courses, … For interacting with our predictive model, users can predict the academic grade and dropout of a student based on stored data of university or upload the csv file. To implement this system, we used the free and open- source Streamlit framework built on the Python programming language. Streamlit helps us to transform the interacting from command line to user- friendly interface with a little front- end experiences. With a few lines of Python codes, we can display the data and collect required parameters or variables to serve multiple tasks of data science.



Student Grade Prediction

Total Courses: 489

Total Students: 331

Total Departments: 8

FIGURE 4.2.10. *The Data driven system to predict the student's academic grade .*

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1. Conclusion

Educational Data Mining (EDM) is a powerful tool for Knowledge analysis in the education field. One of the most important aspects of EDM is to predict the student's performance based on the huge amount of data stored in the university. From this mission, the University council or lecturer can improve the syllabus, lessons or educational programs to provide in-school students with relevant knowledge. In student's aspects, predicting risk of failure at an early stage can help to have an orientation for revision or remediation. In this study, we use the dataset from International University to build a course-specific model to predict the academic grade and dropout rate of upcoming courses depending on the previous outline, re-prerequisite or related courses. In the inheritance aspects of this study's older version, the traditional machine learning techniques such as decision tree(DT), random forest(RF), support vector machine(SVM) and modern deep learning like deep neural network(DNN) give promising results in the range from 80% to 90%. In addition, the effect of handling with the imbalanced datasets is emphasized through the result of experiments through using evaluation measurements like F1-score, accuracy or confusion matrix. From the combination of this result and many academic studies, we continue to expand the scope of this study from focusing on the static data to exploring the timestamp characteristic hidden in this data. To implement this expanded scope, we used another type of artificial neural network, called Long - short Term Memory (LSTM) based on the Recurrent Neural Network (RNN). The key tasks of pre-process data are cleaning data, transforming data from SQL- record data to the multiple hot-encoding matrix and applying the SMOTE approach to handle unbalanced dataset. The hyperas library is used to determine the optimal set of variables and hyperparameters for the predictive model. The model's accuracy evaluation can reach 84%, which is within the previously indicated accuracy range of baseline methods. And, confusion matrix is another model`s measurement to help researchers have the visual perspective about how this course-specific model acts on the validation dataset. To emphasize reliability of the model's outcome, an experiment which assumes that students with high achievement in previous courses can have better result in target is explained to find the influential courses of the Data Structures and algorithms and pre-produced an clearer definition of student at risk of quitting the course. This study goes through three steps of building a knowledge discovery project from data preparation, building and training model and deploy model. In the end, the friendly- user interface is created using open- source Streamlit library to serve two demands of general analysis and collecting the data for model prediction.

5.2. Future work

In the future, we focus on implementing the graphic user interface to deploy the model and serve some student management tasks in University. Except for predicting pass/fail rate, we will extend proposal purposes to predict advanced aspects such as graduation time, course's score or overall student performance… If possible, we hope that we will improve the dataset vertically with adding more semester's data and horizontally with more personal or family features such as demography, time study, or part-time job… About aspects of data mining techniques, the upcoming work is the solutions which can deal with existing problems of educational data mining, specially predicting the academic grade or reducing the dropout rate. In the recent researches, we can observe that applied techniques for this study expands from both the technical aspects and the features selected. From the chaos of the pandemic, the interaction of students in learning management platforms has become a new interest of researchers. Along with developments, this field has transformed from the predictive method to classification that the relationship with similar characteristics of participants through demographic or study activities in school and learning systems has a sizable impact on the performance model. In detail of this, two existing problems are the lack of data and the scope of predictive features. For the lack of availability data or the imbalance, Generative artificial intelligence (AI), a buzzword that is widely used this year across many platforms, may be a potent tool for producing more data from the original or outlier in a variety of formats, from static to dynamic. And, the Generative Adversarial Network (GAN) mentioned is also a powerful technique for data-level issues that can handle imbalanced or missing datasets. The study [15] idea, which divides the input into small partials before sending it to GAN, can be used to produce the minority. The next partial data will be created by the combination of the original data and the previous data made by GAN.

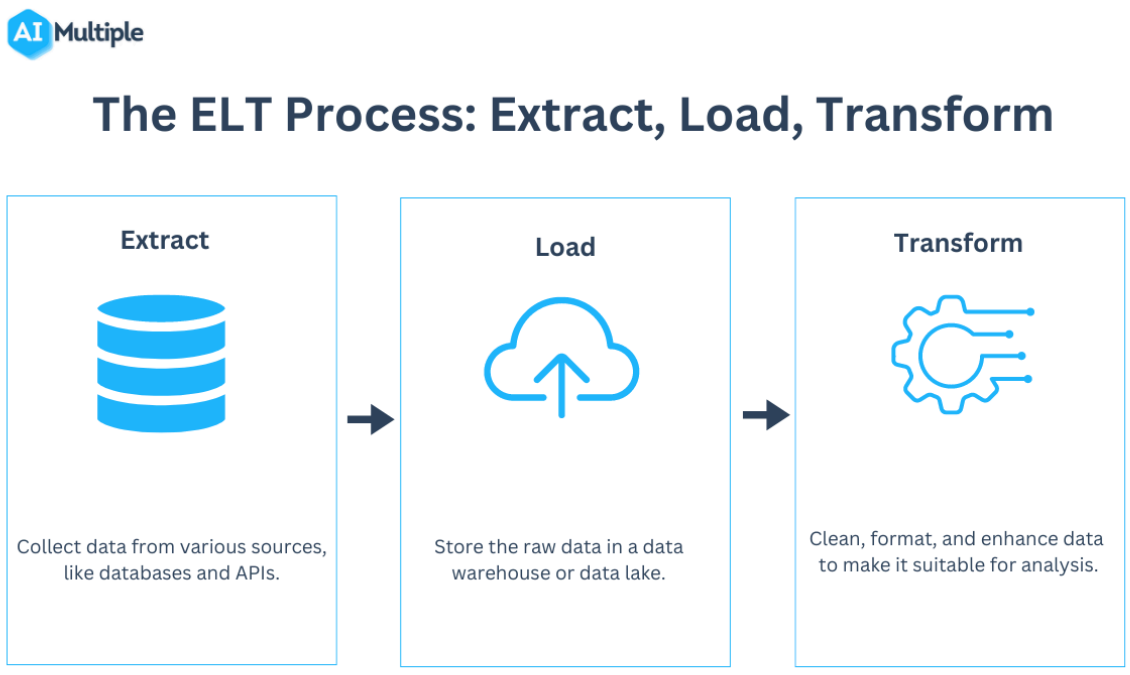

AI Multiple
The ELT Process: Extract, Load, Transform
Extract
Collect data from various sources, like databases and APIs.
Load
Store the raw data in a data warehouse or data lake.
Transform
Clean, format, and enhance data to make it suitable for analysis.

FIGURE 5.2.1. *The ELT Data Process.*

This concept helps the GAN not only learn about the distribution of original data, but also in the new generation data. For the expand the features, nowadays, survey forms can be easily created and delivered to the students through the internet. Going back to the past, the survey analysis is a difficult task because of wasting time and human resources. Today, along with the development of data architecture such as ETL(Extract - Transform - Load) or ELT(Extract- Load - Transform)…. This work becomes an easy task which can be supported end- to -end from a third-party service like Azuze, Amazon or Google. Therefore, updating or integrating a data service part to the student management system which can serve as an automation tool for all types of university's participants to work on is an upcoming work of this research. A system extracting the data from surveys can reach to improving the data horizontally approach mentioned above. From this, promising knowledge analysis will give the university council the omnidirectional evaluation of the student or even educational system.

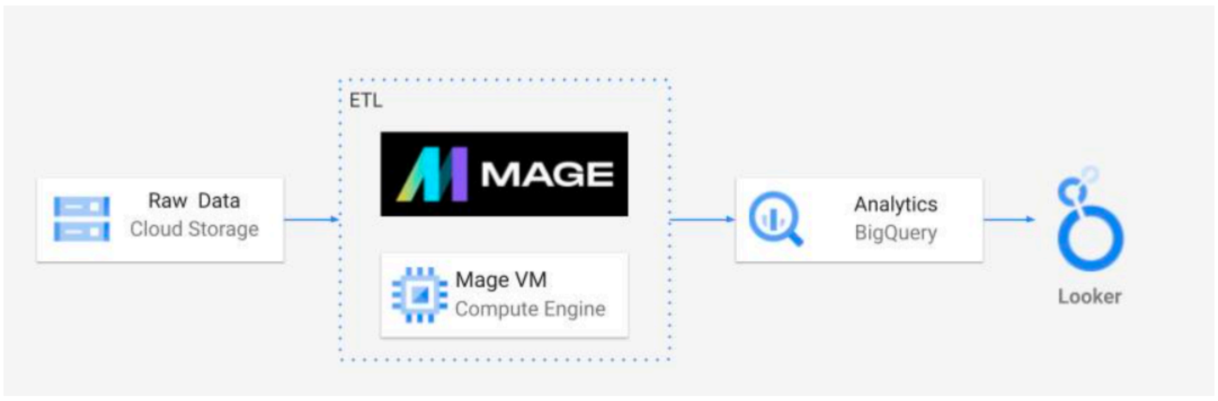


FIGURE 5.2.2. *The simple data architecture based on Google Cloud Platform (From: Google Cloud).*

REFERENCES

1. Qian Hu and Huzefa Rangwala, “Reliable Deep Grade Prediction with Uncertainty Estimation.”, In The 9th International Learning Analytics & Knowledge Conference (LAK19), March 4–8, 2019, available: https://doi.org/10.1145/3303772.3303802.
2. Aya Nabil, Mohammed Seyam, and Ahmed Abou-Elfetouh, “Prediction of Student’s Academic Performance Based on Course’s Grade Using Deep Neural Networks”, Oct. 2021.
3. Siddharth Sharma, Simone Sharma, and Anidhya Athaiya, “Activation Functions in Neural Networks”, vol. 10, pp. 310-316, Apr. 2020.
4. Zachary C. Lipton, “A Critical Review of Recurrent Neural Networks for The Sequence Learning ”, University of California, San Diego, June 2015.
5. Yoshua Bengio, Patrice Simard, and Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult. Neural Networks”, IEEE Transactions on, 5(2):157–166, 1994.
6. Razvan Pascanu, Tomas Mikolov, Yoshua Bengio. “On the difficulty of training Recurrent Neural Network”, International Conference on Machine Learning , Feb 2013, https://arxiv.org/pdf/1211.5063.pdf
7. Van Houdt, G., Mosquera, C. & Nápoles, “A review on the long short-term memory model” , Artif Intell Rev , Dec 2020, https://doi.org/10.1007/s10462-020-09838-1
8. Qing- yun Dai, Chun-ping Zhang and Hao Wu, “Research of Decision Tree Classification Algorithm in Data Mining”, Int. Jour. Database Theory and Application, vol.9, pp.1-8, 2016.

9. L. Bo, A. Abbass and B. Mckay, "Classification rule discovery with ant colony optimization", IEEE Computational Intelligence Bulletin, vol. 3, no. 1, pp. 31- 35, 2004.
10. R Grandhi, "Support vector machine - introduction to machine learning algorithms", Towards Data Science, June.2018, available: https://towardsdatascience.com.
11. Ramentol E, Caballero Y, Bello B and Herrera F, "SMOTE-RSB: A Hybrid Preprocessing Approach based on Oversampling and Undersampling for High Imbalanced Data-Sets using SMOTE and Rough Sets Theory", Knowledge and Information Systems, 2011.
12. B Santoso, H Wijayanto, K A Notodiputro and B Sartono, "Synthetic Over Sampling Methods for Handling Class Imbalanced Problems : A Review", 2017.
13. XuanSheng Lu, Yanmin Zhu, Yanan Xu and Jiadi Yu, "Learning from Multiple Dynamic Graphs of Student and Course Interaction For Student Grade Prediction ", Neurocomputing of Elsevier, 2021.
14. Xiaofeng Ma, Zhurong Zhou, "Student Pass Rates Prediction Using Optimized Support Vector Machine and Decision Tree", 2018.
15. A. Almasri, E.Celebi, and R.S. Alkhawaldeh. "EMT: Ensemble Meta-based Tree Model for Predicting Student Performance", Scientific Programming, 2019. available: https://doi.org/10.1155/2019/3610248.
16. H. Waheed, M. Anas, S.U. Hassan, N.R. Aljohani, S. Alelyani, E.E.Edifor, R. Nawaz, "Balancing Sequential Data to Predict Student At-risk Using Adversarial Network", Computers and Electrical Engineering, 2021.
17. Bulut, O, Gorgun, G, Yildirim-Erbasli, S.N, Wongvorachan, T. Daniels, L.M. Gao, Y. Lai, K.W.Shin, "Standing on the shoulders of giants: Online formative assessments as the foundation for predictive learning analytics models." Br. J. Educ. Technol, 2022.

18. Wongvorachan. T, He.S, Bulut.O, “A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining. Information 2023, 14, 54, available: https://doi.org/10.3390/info14010054.
19. Ma. Y, He.H, “Imbalanced Learning: Foundations, Algorithms, and Applications”, John Wiley & Sons: Hoboken, NJ, USA, 2013.
20. P. Kumari, P. K. Jain, and R. Pamula, “An efficient use of ensemble methods to predict students academic performance,” in Proc. 4th Int. Conf.Recent Adv. Inf. Technol. (RAIT), Mar. 2018, pp. 1–6.
21. N.Srivastava, G.Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov. “Dropout: The Simple Way to Prevent the Neural Network from Overfitting.”, Journal of Machine Learning Research, June. 2016.
22. M.T. Ribeiro, S. Singh, and C. Guestrin. “Why should I trust you?: Explaining the predictions of any classifier.” , International Conference on Knowledge Discovery and Data Mining, 2016. available: http://dx.doi.org/10.1145/2939672.2939778.