Student Grade Prediction Using Multiclass Model

Anitha K[1] , Bhoomika C[2], J Andrea Kagoo[3], Kruthika K[4], Aruna MG[5]
[1,2,3,4] *Students, Dept of CSE, M S Engineering College, Bangalore, Karnataka*
[5] *Associate Professor, Dept of CSE, M S Engineering College, Bangalore, Karnataka*

Abstract **- Today, predictive analytics applications became an urgent desire in higher educational institutions. Predictive analytics use advanced analytics that encompasses machine learning implementation to derive high-quality performance and meaningful information for all education levels. We know that student grade is one of the key performance indicators that can help educators monitor academic performance. During the past decade, researchers have proposed many variants of machine learning techniques in education domains. However, there are several challenges in handling imbalanced datasets for enhancing the performance of predicting student grades. This project presents a comprehensive analysis of machine learning techniques to predict the final student grades in the first semester courses by improving the performance of predictive accuracy. This is carried out in a two-step process. First, we compare the accuracy performance of six well-known machine learning techniques namely Decision Tree (J48), Support Vector Machine (SVM), Naïve Bayes (NB), K-Nearest Neighbor (kNN), Logistic Regression (LR) and Random Forest (RF) using 1282 real student's course grade dataset. Second, we proposed a multiclass prediction model to reduce the overfitting and misclassification results caused by imbalanced multi-classification based on oversampling Synthetic Minority Oversampling Technique (SMOTE) with two features selection methods. The obtained results show that the proposed model integrates with RF give significant improvement with the highest f-measure of 99.5%. This proposed model indicates the comparable and promising results that can enhance the prediction performance model for imbalanced multi-classification for student grade prediction based on data given.**

I.INTRODUCTION

In higher education institutions (HEI), every institution has its student academic management system to record all student data containing information about student academic results in final examination marks and grades in different courses and programs. All student marks and grades have been recorded and used to generate a student academic performance report to evaluate the course achievement every semester. The data keep in the repository can sbe used to discover insightful information related to student academic performance. Due to this, many previous researchers have well-defined the influence factors that can highly affect student academic performance. However, most common factors are relying on socioeconomic background, demographics and learning activities compared to final student grades in the final examination. As for this reason, we observe that the trend of predicting student grades can be one of the solutions that are applicable to improve student academic performance.

Predictive analytics has shown the successful benefit in the HEI. It can be a potential approach to benefit the competitive educational domain to find hidden patterns and make predictions trends in a vast database. It has been used to solve several educational areas that include student performance, dropout prediction, academic early warning systems, and course selection. Moreover, the application of predictive analytics in predicting student academic performance has increased over the years. The ability to predict student grade is one of the important areas that can help to improve student academic performance. Many previous research has found variant machine learning techniques performed in predicting student academic performance. The related works on mechanism to improve imbalanced multi-classification problem in predicting students' grade prediction are difficult to find.

Therefore, in this study a comparative analysis has been done to find the best prediction model for student grade prediction by addressing the following questions.

RQ1: Which predictive model among the selected machine learning algorithms performs high accuracy performance to predict student's final course grades?

RQ2: How can imbalanced multi-classification datasets be addressed with selected machine learning algorithms using oversampling Synthetic Minority Oversampling Technique (SMOTE) and feature selection (FS) methods? To address the above-mentioned questions, we collect the student final course grades from two core courses in the first semester of the final examination result. We present a descriptive analysis of student datasets to visualize student grade trends, which can lead to strategic planning in decision making for the lecturers to help students more effectively. Then, we conduct comparative analysis using six well-known machine learning algorithms, including LR, NB, J48, SVM, kNN and RF on the real student data of Diploma in Information Technology (Digital Technology) at one of Malaysia Polytechnic. As for addressing the imbalanced multi-classification, we endeavour to enhance the performance of each predictive model with data-level solutions using oversampling SMOTE and FS.

To overcome challenges in handling imbalanced datasets for enhancing the performance of predicting student grades, to efficiently deal with the overfitting and misclassification results caused by imbalanced multi-classification of datasets and develop a model that indicates the comparable, promising results that can enhance the prediction performance model of the machine learning techniques by improving the imbalanced multi- classification for student grade prediction.

The main objectives of the project describe a brief outline of what is expected. It is as follows:

- To help educators monitor the academic performance of students and give an idea about the final result.
- To generate a multiclassification model based on the six algorithms –Decision Tree, Support Vector Machine, Random Forest, Naïve Bayes, K-Nearest Neighbor (kNN), Logistic Regression.
- To create a predictive model that is – accurate and precise, which gives the lowest error margin for prediction.
- To build an automated detection system for student academic performance with 98% accuracy.

In the world of open education systems, students have flexibility to learn anything with ease as the learning content is easily available. This facility can make a student complacent. Therefore, it becomes difficult to predict the student's performance in advance. In this project, an attempt is made to help the student to know their performance in advance.

II. RELATED WORKS

Several studies have been conducted in HEI for predicting student grades using various machine learning techniques. It involves analytical process of many attributes and samples data from variety of sources for student grade prediction in different outcome. However, the performance of predictive model for imbalanced dataset in education domains are still rarely discussed.

A study used discretization and oversampling SMOTE methods to improve the accuracy of students' final grade prediction. Several classification algorithms have been applied such as Naïve Bayes (NB), Decision Tree and Neural Network (NN) for classifying students' final grade into five categories; A, B, C, D and F. They showed that NN and NB applied with SMOTE and optimal equal width binning outperformed other methods with similar highest accuracy of 75%. A method for predicting future course grades obtained from the Computer Science and Engineering (CSE) and Electrical and Computer Engineering (ECE) programs at the University of Minnesota. Based on the proposed methods, the results indicated that Matrix Factorization (MF) and Linear Regression (LinReg) performed more accurate predictions than the existing traditional methods. It is also found that the use of a course-specific subset of data can improve prediction accuracy for predicting future course grades.

A study has developed a predictive model that can predict student's final grades in introductory courses at an early stage of the semester, where eleven machine learning algorithms were compared in five different categories consist of Bayes, Function, Lazy (IBK), Rules-Based (RB) and Decision Tree (DT) using WEKA. To reduce high dimensionality and unbalanced data, they have performed feature selection correlation- based and information-gain for data pre-processing. Among the 11 algorithms, it is indicated that Decision Tree classifier (J48) have the highest accuracy of 88% compared to other categories of algorithms.

III. SYSTEM DESIGN

The system design is defined as the process of applying various requirements and permits it physical realization. Various design features are followed to develop the system the design specification describes the features of the system.

System Architecture

The Figure 1.1 illustrates the flowchart of the proposed multiclass prediction model used in this study, input given is the records of students, and output is the visual representation in the form of results and graphs between different algorithms.

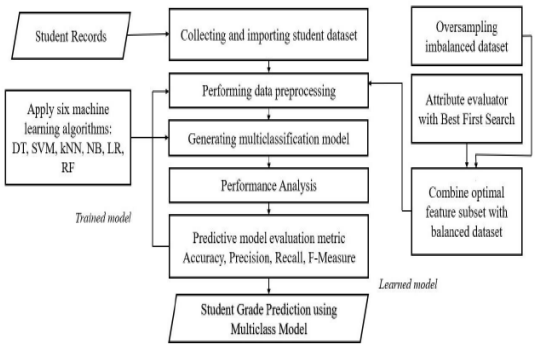

Figure 1.1: System Architecture

The process as indicated starts from the collection of student records. The records are then imported to a file to be read. Data pre-processing is an essential step in removing null data and encoding the attributes. The pre-processed data is oversampled with the SMOTE method. A multiclassification model is generated based on the six algorithms – Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbor (kNN), Naïve Bayes (NB), Logistic Regression (LR) and Random Forest (RF). The algorithms are each analysed on the basis of test and train datasets. Further, the performance is evaluated using accuracy, precision, recall and f-measure factors. Finally, the output given is the student grade predicted.

IV. FRAMEWORK OF STUDENT GRADE PREDICTION USING MULTICLASS MODEL

It describes the various modules of the project. The modular description aims at providing information on each segment of the process of the project. It is important since it provides a summary of the project at a glance and can be used in the future for further development and improvements in the project. The project modules are described in the forthcoming section.

A. DATA PREPARATION

The dataset used was collected by the Department of Information and Communication Technology (JTMK) at one of the Malaysia Polytechnics. The dataset contains 1282 instances which consists of the total course grades of the first semester students taken from the final examination during June 2016 to December 2019 session. Students need to take some compulsory, specialization and core courses modules to qualify them for the next academic semester. However, in this study only two core courses that contained the percentage of final examination and course assessment marks have been selected. All features which are used for prediction are listed and the process is illustrated . The attributes contained in the dataset are Student ID, Year, Class, Session, Credits, Course Code, Total Marks obtained, Grade Point Average (GPA), Grade, Group – which is calculated from the grades obtained.

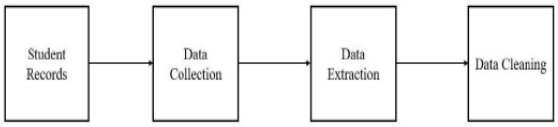

Figure 2.1 : Data Preparation

In the above Figure 2.1 shows Student records are stored in the form of datasets. Data information such as Student ID, year, class, session, credit hour, total marks etc. are collected from database. The data is then extracted from dataset and converted into raw data. This gets stored in a readable file, so as to help in reading values from the dataset. The whitespaces and blank spaces in data, if any are removed.

The algorithm ReadData() is tasked with reading the contents of file to form the dataset. The process followed is given in the following algorithm.

Algorithm ReadData()
{
//Input: Data from csv file
//Output: Dataset
Begin
STEP 1: Read data from csv file
STEP 2: Store values in dataset
STEP 3: Return raw dataset
End
}

The algorithm DataPrepare() helps in forming meaningful data by removing unwanted symbols and whitespaces in the contents of the dataset.

Algorithm DataPrepare()
{
//Input: Dataset
//Output: Raw dataset
Begin
STEP 1: Read dataset
STEP 2: Remove unwanted symbols in data
STEP 3: Return raw dataset
End
}

B. DATA PRE-PROCESSING AND DESIGN MODEL

In this phase, data pre-processing is applied for the collected dataset. For the convenience of data pre-processing, the data is used to rank and group the students into 5 categories of grades: Exceptional (A+), Excellent (A), Distinction (A−, B+, B), Pass (B−, C+, C, C−, D+, D) and Fail (E, E−, F). The group was created to be the output of the prediction class. However, the class distribution of the dataset indicated an imbalanced class instances containing number of (63) exceptional, (377) excellent, (635) distinction, (186) pass and (21) fail with high number of ratios 3:18:30:9:1 that can lead to overfit results.Therefore, data-level solution using oversampling SMOTE and two FS methods. Wrapper and Filter were used as the benchmark methods in this study to overcome the problem of imbalanced multi-classification dataset.the process shown in figure 2.2

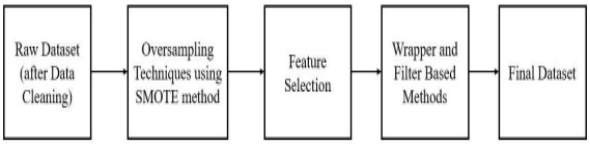

Figure 2.2 :Data Pre-Processing

Wrapper Method: Wrapper method is a subset of features is tried and used to train model. Based upon inferences from the previous model, features are added or removed from the subset as shown in figure 2.3, Problem is essentially reduced to a search problem.

Filter Method: Filter method are generally used as preprocessing step as shown in figure 2.4 The selection of features is independent of any machine learning algorithms. Features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable.

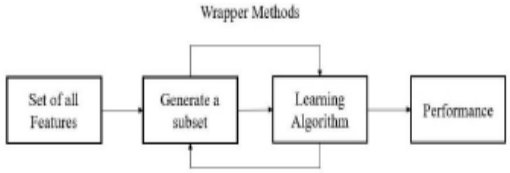

Figure 2.3 : Wrapper Method

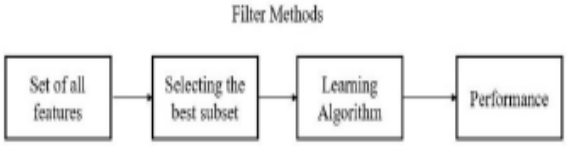

Figure 2.4: Filter Method

Filter methods measure the relevance of features by their correlation with dependent variable while wrapper methods measure the usefulness of a subset of feature by actually training a model on it. Filter methods are much faster compared to wrapper methods as they do not involve training the models. On the other hand, wrapper methods are computationally very expensive as well. Filter methods use statistical methods for evaluation of a subset of features while wrapper methods use cross validation.

The algorithm process for data pre-processing is given by the method DataPreprocess().

Algorithm DataPreprocess()
{
//Input: Raw Dataset
//Output: Cleaned dataset
Begin
STEP 1: Read dataset
STEP 2: Find any missing values
STEP 3: IF any missing values THEN Remove the row by using dropna()
STEP 4: Return raw dataset with no null values
End
}

The algorithm BuildSMOTE() helps in building a model using the SMOTE method to handle oversampling of imbalanced datasets.

Algorithm BuildSMOTE()
{
//Input: Cleaned Dataset
//Output: Oversampled dataset
Begin

STEP 1: Read dataset T
2: Calculate oversampling data size N
STEP 3: Initialize T*=T
STEP 4: for i from 1 to N
Select one sample from b, e, n samples as x
Select nearest neighbor of x in majority
Select x in minority classes
Calculate values with respect to sample importance of x and x_knn
Generate synthetic minority oversample to T
STEP 5: Return T
End
}

The algorithm Select Points finds the features to plot the graph for the dataset.

Algorithm SelectPoints()
{
//Input: X and Y values
//Output: Selected features
Begin
STEP 1: Read X, Y
End
}

The final dataset is obtained, to which the machine learning algorithms are applied to generate a multiclassification model. This process is represented by Figure 2.5 given below.

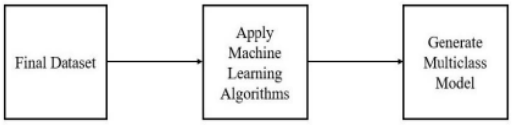

Figure 2.5 : Design Model

In particular, the following are the theoretical algorithms used as basis to construct the multiclass prediction model.

1) Decision Tree
2) Support Vector Machine (SVM)
3) Naïve Bayes (NB)
4) K-Nearest Neighbor (kNN)
5) Logistic Regression (LR)
6) Random Forest (RF)

Machine Learning algorithms are reliable for many applications. The algorithms are made to be fast and efficient. The various regressions are applied for different datasets and efficiently predict according to the system requirements or as specified by the user. Machine learning algorithms are used to automatically understand and realize the day-to-day problems that people are facing.

The DesignModel() details the model design module.
Algorithm DesignModel()
{
//Input: Cleaned dataset
//Output: Prediction model
Begin
STEP 1: Read dataset T

STEP 2: Read n //to determine algorithm

STEP 3: IF(n==1) {
Initialize RF=RandomForestMethod()
Call RF.SelectPoints()
Store result1 in predictions1}

STEP 4: ELSE IF(n==2) {
Initialize DT=DecisionTreeMethod ()
Call DT.SelectPoints()
Store result2 in predictions2 }

STEP 5: ELSE IF(n==3) {
Initialize KNNs=KNearestNeighborMethod ()
Call KNNs.SelectPoints()
Store result3 in predictions3 }

STEP 6: ELSE IF(n==4) {
Initialize NB=NaiveBayesMethod ()
Call NB.SelectPoints()
Store result4 in predictions4 }

STEP 7: ELSE IF(n==5) {
Initialize SVM=SupportVectorMachineMethod()
Call SVM.SelectPoints()
Store result5 in predictions5 }

STEP 8: ELSE IF(n==6) {
Initialize LR=LogisticRegressionMethod()
Call LR.SelectPoints()
Store result6 in predictions6 }
End
}

Logistic Regression (LR) is known as cost function that uses logistic functions. It represents mathematical modeling to solve classification problems. The model performs great contextual analysis for categorical data

to understand the relationship between variables. The method estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. Data is fit into linear regression model, which then be acted upon by a logistic function predicting the target categorical dependent variable. The output from the hypothesis is the estimated probability.

Algorithm LogisticRegressionMethod()
{
//Input: Training Data
//Output: Resultant class
Begin
STEP 1: Loop i from 1 to k
STEP 2: For each training data instance d_i:
Set target value for regression to z
STEP 3: Initialize the weight of instance d_j to P(1|d_j). (1-P(1-d_j))
STEP 4: Finalize to the data with class value (z_j) and weights (w_j)
STEP 5: Assign (class label: 1) if P (1|d_j) > 0.5, otherwise (class label: 2)
End
}

Naïve Bayes (NB) is based on Bayesian theorem that widely used as it is simple and able to make fast predictions. It is suitable for small datasets that combines complexity with a flexible probabilistic model. It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge depends on the conditional probability.

Algorithm NaiveBayesMethod()
{
//Input: Training dataset T, F=(f1,f2,f3….,$f\_(n)$) ← value of the predictor variable
//Output: Resultant class
Begin
STEP 1: Read the training dataset T
STEP 2: Calculate the mean and standard deviation of the predictor variables in each class
STEP 3: Calculate the mean & standard deviation of predictor in each class
Until the probability of all predictor variables ($f\_$ (1,)..f_n) is calculated.
STEP 4: Calculate the likelihood for each class
STEP 5: Find the greatest likelihood class
End
}

Decision Tree (J48) a widely used in several multi class classification that can handle missing values with high dimensional data. It has been implemented effectively for giving an optimum result of accuracy with minimum number of features.
It is a flowchart-like structure in which each internal node represents a "test" on an attribute. A tree can be “learned” by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning.
An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute.
The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. The construction of a decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.
Decision trees can handle high-dimensional data. Generally, decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge based on classification

Algorithm DecisionTreeMethod()
{
//Input: Training dataset
//Output: Resultant class
Begin
STEP 1: Set the value of k
STEP 2: Loop 1 to N //To get predicted class
2.1 Calculate the distance Di Euclidian/ Cosine/Chebyshev between data instance in training data and test data.

STEP 3: Increasingly arrange the computed distances (Di)
STEP 4: Populate the upper k results from the arranged list.
STEP 5: Pick up the most frequent class from the list
End
}

Support Vector Machine (SVM) is based on the notion of decision planes that states decision boundaries which handle classification problem successfully. a sorted dataset and predicts, which of two conceivable classes includes the information, making the SVM a non-probabilistic binary linear classifier.
Support Vector Machine or SVM is one of the most popular supervised learning algorithms, which is used for classification as well as regression problems.
The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
SVM chooses the extreme points or vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.
The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, they are called support vectors.
SVM finds the closest point of the lines from the classes. The distance between the vectors and the hyperplane is called as margin. The SVM maximizes this margin. The hyperplane with maximum margin is called the optimal hyperplane.
Algorithm SupportVectorMachineMethod()
{
//Input: α:=0 or α:=partially trained SVM, X and y
//Output: Support Vector Class (SV) (α_i>0)
Begin
STEP 1: C:=Approximate upper bound constant value
STEP 2: Loop for all: {X_i, y_i} and {$X_(j,) y_j$}
Do
Optimize α_i and α_j
End loop
STEP 3: Until no changes in α (or some resource constraint encounters)
End
}

K-Nearest Neighbor (kNN) is a non-parametric algorithm that classifies and calculate the difference between instances in the dataset based on their nearest vectors, where k refers to the distance in the n-dimensional space. K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
The kNN algorithm assumes the similarity between the new case or data and available cases and put the new case into the category that is most similar to the available categories.
It stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using kNN algorithm.
kNN can be used for regression as well as for classification but it is mostly used for the classification problems. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
kNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
The benefits of kNN are that it is simple to implement, provides robustness to the noisy training data and can be more effective if the training data is large.

Algorithm KNearestNeighborMethod()
{
//Input: Data
//Output: Resultant class
Begin
STEP 1: Set the value of k
STEP 2: Loop 1 to N //to get predicted class
Calculate the Euclidean distance Di between training data and test data
STEP 3: Increasingly arrange the computed distances Di
STEP 4: Populate the upper k results from the arranged list
STEP 5: Select the most frequent class from list
End
}

Random Forest (RF) is a classifier based on ensemble learning that used number of decision trees on various subset to find the best features for high accuracy. It prevents the problem of overfitting. It is relatively robust to outliers and noise that operates effectively in classification.

The fundamental concept behind random forest is a simple but powerful one. It utilizes the wisdom of crowds. The reason that the random forest model works so well is that a large number of relatively uncorrelated models or trees operating as a committee will outperform any of the individual constituent models.

The low correlation between models is the key. Correlations come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors, as long as they don't constantly all err in the same direction. While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The RF method is advantageous as it takes less training time as compared to other algorithms, predicts output with high accuracy, even for a large dataset it runs efficiently and also maintains accuracy when a large proportion of data is missing.

Algorithm RandomForestMethod()
{
//Input: Training data
//Output: Final predicted class
Begin
STEP 1: Predict and store the outcome of each randomly created decision trees
on given test data
STEP 2: Compute the occurrence for individual class
STEP 3: Declare majority class as the final outcome class
End
}

C. PERFORMANCE ANALYSIS

This project aims to predict students' final grades based on their previous course performance records in the first semester's final examination. The proposed model applied different machine learning algorithms to evaluate which of the algorithms performed the highest performance for predicting student's final grades. There are three experiments conducted in four distinct phases based on the five different classes. The accuracy is evaluated using ten-fold cross-validation for which the dataset is partitioned into 90% for training set and 10% for testing set, as shown in Figure 2.6

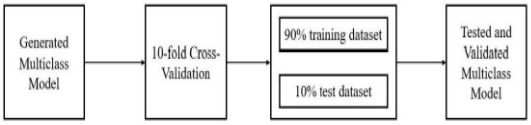

Figure 2.6 : Performance Analysis

Cross validation is method applied to a model and a dataset to estimate the out of sample error. In k-fold cross validation, the data set is split randomly into k partitions, model is fit to a data set consisting of k-1 of the original k parts, and the remaining portion of validation. The out of sample error is estimated using portion of data left out of the fitting procedure. This is repeated k times and estimate for the out of sample error is average or mean over the k-validation run.10-fold cross validation performs the fitting procedure a total of ten times given by Figure with each fit being performed on a training set consisting of 90% of the total training set selected at random, with the remaining 10%used as a hold-out set for validation.The choice of k is usually 5 or 10, but there is no formal rule. As k gets larger, the difference in size between the training set and the resampling subsets gets smaller. As this difference, the bias of the technique becomes smaller(the bias is smaller for k=10 then k=5). Here, the bias is the difference between the estimated and true values of performance. This process is shown using TrainModel() for generating the multiclassification model.

Algorithm TrainModel()
{
//Input: Cleaned dataset
//Output: Trained model
Begin
STEP 1: Read dataset

STEP 2: Split dataset into train and test
STEP 3: Build model using sklearn
STEP 4: Train dataset using the model
STEP 5: Return Trained Model
End
}

D.PERFORMANCE EVALUTION

The validated multiclass model with help of a confusion matrix is checked for accuracy, precision, recall and f-measure, the process is shown in Figure 2.7

Confusion Matrix is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values: (1) True Positive: Predicted value is positive and actual value is true (2) True Negative: Predicted value is negative and actual value is true (3) False Positive (Type 1 Error): Predicted value is positive actual value is false (4) False Negative (Type 2 Error): Predicted value and actual value is false.

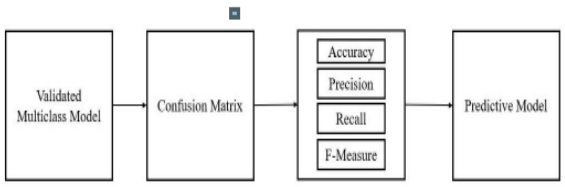

Figure 2.7 : Performance Evolution

Accuracy (A) is the total number of correct predictions divided by the total number of predictions made for a dataset. It is useful when all classes are of equal importance. Precision (P) is a metric that quantifies the number of correct positive predictions made. Precision, calculates the accuracy for the minority class. In imbalanced classification problems, the majority class is typically referred to as the negative outcome (such as “no change” or “negative test result“), and the minority class is typically referred to as the positive outcome (“change” or “positive test result”). The simplest confusion matrix is for a two-class classification problem, with negative (class 0) and positive (class 1) classes.In a binary classification, it is calculated as the ratio of correctly predicted positive examples divided by the total number of positive examples that were predicted. The goal of the precision is to classify all the Positive samples as Positive, and not misclassify a negative sample as Positive.

Precision = TruePositives / (TruePositives + FalsePositives)

In an imbalanced classification problem with more than two classes, precision is calculated as the sum of true positives across all classes divided by the sum of true positives and false positives across all classes.

Precision = Sum c in C TruePositives_c / Sum c in C (TruePositives_c + FalsePositives_c)

Recall (R) is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made. The result is a value between 0.0 for no recall and 1.0 for full or perfect recall.

Recall = TruePositives/(TruePositives + FalseNegatives) In an imbalanced classification problem with more than two classes, recall is calculated as the sum of true positives across all classes divided by the sum of true positives and false negatives across all classes. Recall = Sum c in C TruePositives_c / Sum c in C (TruePositives_c + FalseNegatives_c) Precision: Appropriate when minimizing false positives is the focus. Recall is appropriate when minimizing false negatives is the focus. The scenario in which there is a need to choose a model with high recall is when the predicted outcome leads to a critical decision.

Algorithm EvaluateMethod()
{
//Input: Prediction for each algorithm
//Output: Accuracy of algorithm
Begin
STEP 1: Read input values from predictions
STEP 2: Generate the confusion matrix for each algorithm
STEP 3: Compare the accuracy
STEP 4: Select the algorithm with best accuracy for predicting grade
STEP 5: Return comparison graph
End
}

E.DATA VISUALIZATION

The basic use of the Data Visualization technique is that it is a powerful technique to explore the data with presentable and interpretable results. In the data mining process, it acts as a primary step in the pre-

processing portion. The entire process is represented by Figure 2.8

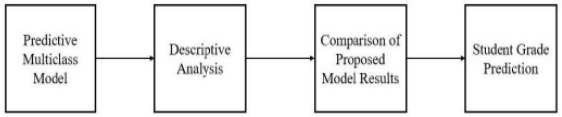

Figure 2.8 : Data Visualization

The final output is the predicted student grade, which is given in the form of a single letter grade such as 'A', 'B', 'C', 'D', 'E' and 'F'. The best-case scenario is the grade 'A', which means that the student has the possibility of obtaining results within the scope of it being an outstanding result. The worst-case scenario is obtaining the 'F' grade, which means the student could be at risk of failure. When the criteria for prediction changes, the grade changes as well.

V EXPERIMENTAL RESULTS

This phase involves the evaluation of algorithmic performance and which algorithm gives the best possible results. This is measured on the basis of accuracy.

Algorithmic Comparison

The performance of the algorithms is calculated on the basis of accuracy given by each, figure 3.1 shows the graphs obtained on the result given by each algorithm.

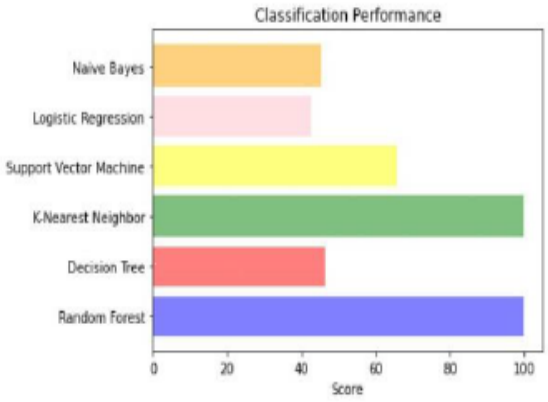

Figure 3.1: Accuracy Comparison of Algorithms

On observing the graph, it can be seen that the algorithms giving the most accuracy are the K-Nearest Neighbor and the Random Forest algorithm. However, since the project deals with handling imbalanced datasets, this graph alone isn't enough in finding the best algorithm. To solve this issue, we employ the SMOTE method. On creating a graph with SMOTE for the same algorithms.

The following histogram provides accuracy for algorithms with and without the SMOTE method applied.

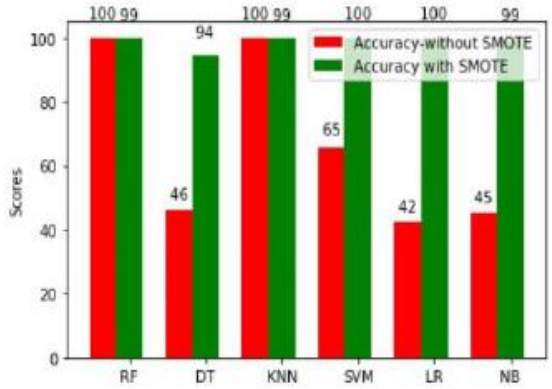

Figure 3.2: Accuracy Comparison with and without SMOTE

In figure 3.2, The red bars of the graph show values without the SMOTE method applied, and the green bars on the graph represent accuracy values with the application of SMOTE. It can be concluded that the Random Forest and K-Nearest Neighbor algorithms are capable of giving accurate results for both balanced and imbalanced datasets. However, kNN algorithm is incapable of handling large datasets, unlike Random Forest algorithm. These values are bound to change for different datasets used. The algorithms have the following results on accuracy which are given as:

Random Forest (RF) shows 100% without SMOTE and 99% with SMOTE method applied.

Decision Tree (DT) gives 46% without SMOTE and 94% with SMOTE method applied.

K-Nearest Neighbor (kNN) shows 100% without SMOTE and 99% with SMOTE method.

Support Vector Machine (SVM) has 65% without SMOTE and 100% with SMOTE.

Logistic Regression (LR) has only 42% without SMOTE and 100% with SMOTE method.

Naïve Bayes (NB) gives 45% without SMOTE and 99% with SMOTE method applied.

VI. CONCLUSION AND FUTURE DIRECTIONS

It is important to have a predictive model that can reduce the level of uncertainty in the outcome for an imbalanced dataset. This project proposes a multiclass prediction model with six predictive models to predict

final student's grades based on the previous student final examination result of the first-semester course.

The comparative analysis of combining oversampling SMOTE with different FS methods evaluates the performance accuracy of student grade prediction. We also infer that the explored oversampling SMOTE is overall improved consistently than using FS alone with all predictive models. However, the proposed multiclass prediction model performed more effectively than using oversampling SMOTE and FS alone with some parameter settings that can influence the performance accuracy of all predictive models.

The project contributes to be a practical approach for addressing the imbalanced multi-classification based on the data-level solution for student grade prediction. It is also essential to select several multi- class imbalanced datasets to be analysed with appropriate sampling techniques and different evaluation metrics which suitable for the imbalanced multi-class domain such as Kappa, Weighted Accuracy and other measures. Thus, using machine learning in higher learning institutions for student grade prediction will ultimately enhance the decision support system to improve their student academic performance in the future.

The application can be further enhanced to suit smartphones and tablets, instead of being exclusive to web application on PCs alone. Also, a database to save a record of the system users can be implemented to track user activity.

SCREENSHOTS

This Paper contains the snapshots of the results obtained for Student Grade Prediction using Multiclass Model.

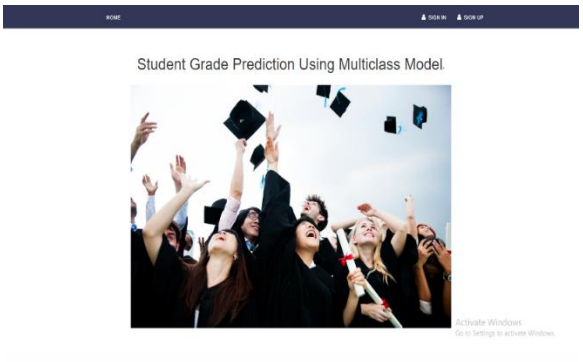

Figure 4.1: Home Page

Figure 4.1 shows the home screen that the user sees on launching the application.

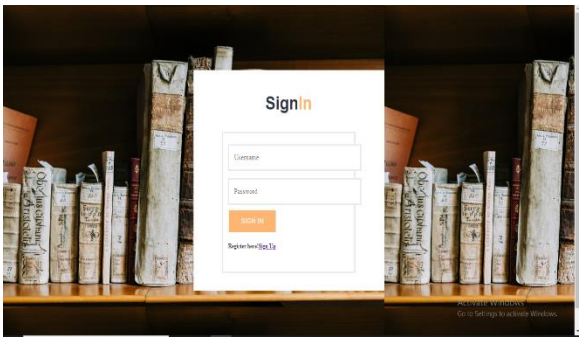

Figure 4.2: Sign-in page

Figure 4.2 shows the Sign In page of the application where registered users can log in. New users can register through the Sign Up option shown.

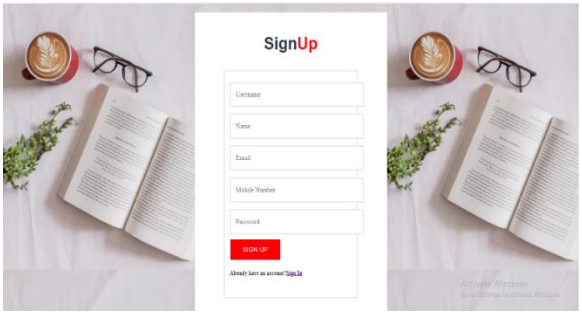

Figure 4.3: Sign-up Page

Figure 4.3 shows the Sign Up page where new users can register themselves, with the username, name, email address, phone number and a password. This is essential in allowing the user to create an account, so as to use the application again, whenever necessary.

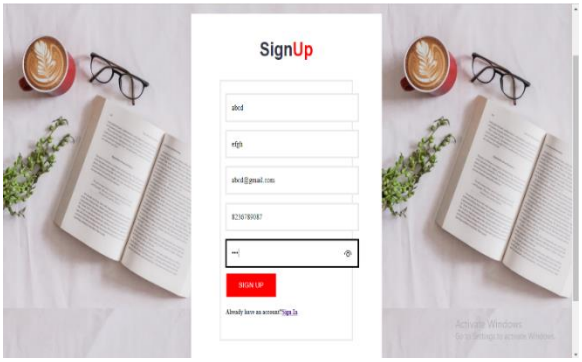

Figure 4.4: Sign-up Page with User Registration

Figure 4.4 shows a sample filled in sign up page, where the user is in the process of completing their profile. The entered details are username, name, email address, phone number, password which is encrypted. Once the user registers, the data is saved for later use as well. This ensures the application is secure on some level and can be used to restrict unauthorized access to sensitive information.

Figure 4.5: Sign-in Page after Registration

Figure 4.5 redirects the user to the sign in page. Here the user can enter their username and password to get started with the application. The login credentials are already known to the user and were created at the time of user registration.

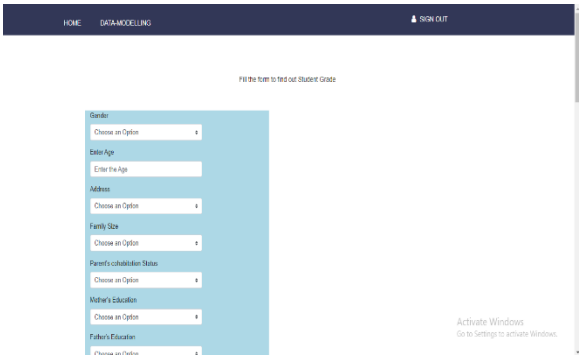

Figure 4.6: Student Grade Analysis Form

Figure 4.6 shows the form to be completed. The form takes the details in the form of options for mostly all entries and numeric entries where numbers play a role. Here, it is used to determine the age of the student and the number of days the student was absent. The options are presented as a dropdown to make it more convenient for users. This aspect makes the application user friendly.

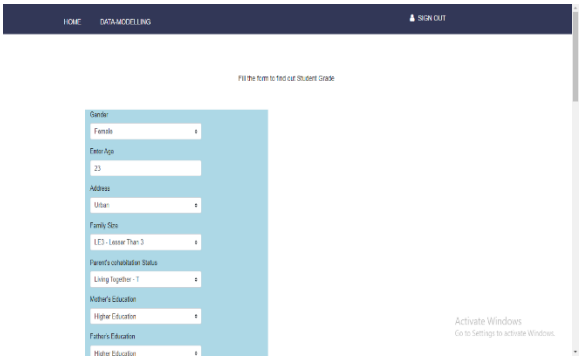

Figure 4.7: Student Grade Form with Entries

Figure 4.7 shows the sample filled form. The form is completed in the figure above. When an incomplete form is given or the details aren't filled entirely, the form cannot be submitted. Form can only be saved on submission when it is completely filled.

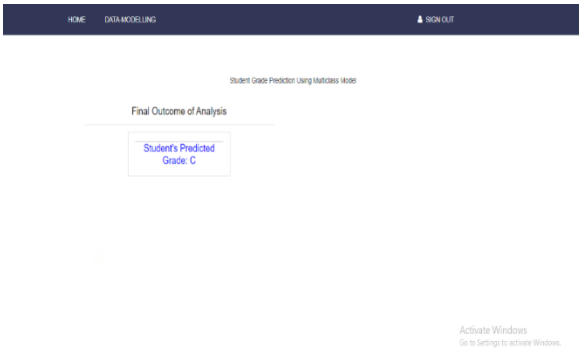

Figure 4.8: Predicted Grade Page

Figure 4.8 gives the final expected output. The expected grade for students are 'S', 'A', 'B', 'C', 'D', 'E' and 'F'. The best case scenario is when the grade is 'S' or outstanding. A grade of 'F' indicates that the likelihood to fail is high.

ACKNOWLEDGMENT

The authors are grateful for student support in consultations regarding application aspects

REFERENCE

[1] "Predicting performance and potential difficulties of university student using classification: Survey Paper" by D Solomon, S Patil and P Agarwal – 2018

[2] "Early segmentation of students according to their academic performance: A predictive modelling approach" by V L Migueis, A Freitas, P J V Garcia, A Silva – 2018

[3] "Tracking student performance in introductory programming by means of machine learning" by I Khan, A Al Sadiri, A R Ahmad, N Jabeur – 2019

[4] "Prediction of student's performance by modelling small dataset size" by L M Abu Zohair – 2019

[5] "Educational data mining and learning analytics for 21st century higher education: A review and synthesis" by H Aldowah, H Al-Samarraie, W M Fauzy – 2019

[6] "Big educational data & analytics: Survey, architecture and challenges" by KLM Ang, FL Ge, KP Seng
[7] "Get more from less: A hybrid machine learning framework for improving early predictions in STEM education" by Mohammad Rashedul Hasan, Mohamed Aly –2019
[8] "Student performance predictor using multiclass support vector classification algorithm" by Suhas S Athani, Mayur N Banavasi, Sharath A Kodli, P G Sunitha Hiremath – 2017
[9] "An empirical analysis of classification techniques for predicting academic performance" by S Taruna, Mrinal Pandey – 2014