

Enhancing Imbalanced Data Classification with Class-Aware Synthetic Minority Over-sampling Technique

RUTURAJ MAHAJAN¹, SACHIN PATIL², VILABHA PATIL³

¹Department of Computer Science and Engineering,
Rajarambapu Institute of Technology, Affiliated to Shivaji University,
Kolhapur, Rajaramnagar,
INDIA

²Department of Computer Science and Engineering,
Rajarambapu Institute of Technology, Affiliated to Shivaji University,
Kolhapur, Rajaramnagar,
INDIA

³Department of Electronics and Telecommunication Engineering,
Rajarambapu Institute of Technology, Affiliated to Shivaji University,
Kolhapur, Rajaramnagar,
INDIA

Abstract: - Imbalanced data classification is one of the difficult machine learning tasks in which the majority class outnumbers the minority classes. This problem is prevalent in various domains such as medical disease detection, spam/fraud detection, digital marketing, agriculture and telecommunications. Synthetic Minority Over-sampling Technique (SMOTE) is a popular oversampling strategy that has been used in addressing the imbalanced dataset classification. However, SMOTE has limitations such as generating noisy samples, ignoring the underlying distribution of the data and over-sampling specific minority regions to the point of over-fitting. To address these limitations, several variants of SMOTE have been proposed, including ADASYN, Borderline-SMOTE, and Safe-Level-SMOTE. However, while these variants aim to address the limitations of SMOTE, they too have their own set of limitations, and their effectiveness may vary across different datasets and problem domains. Therefore, there is still room for improvement in over-sampling techniques to address the limitations of existing methods and improve their performance in diverse scenarios. As a solution, a novel extension of the SMOTE algorithm named "Class-Aware Synthetic Minority over-sampling Technique" (CA-SMOTE) has been proposed. By identifying and utilizing both minority and majority classes to create a synthetic sample. CA-SMOTE generates synthetic samples that are more diverse and closer to the underlying distribution of the data. With this approach, this technique aims to improve the accuracy and reliability of machine learning models trained on imbalanced data. The performance of CA-SMOTE is evaluated on several imbalanced datasets and compared it with SMOTE as well as other modern over-sampling algorithms. The experimental results prove that CA-SMOTE outperforms existing methods in terms of F1-score and AUC-ROC. Overall, CA-SMOTE can be a promising solution for enhanced imbalanced data classification in various real-world applications, improving the accuracy and reliability of machine learning models.

Key-Words: - Data Classification, Data preprocessing, Class imbalance, Over-sampling, SMOTE, Performance evaluation.

Received: April 21, 2026. Revised: June 6, 2026. Accepted: July 11, 2026. Published: September 29, 2026.

1 Introduction

Class imbalance is a common issue in classification problems, where the number of examples in one class significantly outnumbers those in the other(s). Traditional classification algorithms often perform poorly in such scenarios, because they frequently

favor the majority class while providing incorrect predictions for the minority class. As a result, the demand for practical methods to solve the problem of class imbalance has grown during the past few years.

In various real-world classification problems, the Synthetic Minority Over-sampling Technique (SMOTE) and its variants have been widely used to address the issue of class imbalance. However, these techniques have limitations that can affect their effectiveness. One of the major flaws of SMOTE and its variants is that they can produce noisy samples. Additionally, over-sampling the minority class can lead to overfitting, which can negatively impact classification performance, [1]. Furthermore, when the minority class and the majority class are not sufficiently separated, SMOTE and its variants may not be effective. In such cases, synthetic samples generated by SMOTE and its variants may be located in regions of the majority class, resulting in suboptimal classification performance. Finally, SMOTE and its variants often do not consider the unique characteristics of the minority class, such as the distribution of feature values or the high proportion of outliers in the data, which can limit their applicability in certain scenarios.

To address these limitations, a new variant of SMOTE has been proposed, called the "Class-Aware Synthetic Minority over-sampling Technique" (CA-SMOTE), which incorporates the usage of both minority and majority classes in SMOTE to create new samples. The technique is aware of both minority and majority classes and it creates synthetic samples differently based on classes of both the selected sample and its nearest neighbor. This creates synthetic samples that are nearer to the decision boundary but not in the region of another class. This helps to get over SMOTE's drawbacks and can enhance the effectiveness of classifiers developed on unbalanced datasets.

The performance of the CA-SMOTE technique was evaluated on several benchmark datasets and compared with existing techniques, including SMOTE and its variants. The experimental findings show that the suggested technique performs better than the currently used techniques in terms of classification's F1-score and AUC-ROC. It also demonstrates the interpretation of the synthetic samples generated by the method, which can help in understanding of the characteristics of the minority class and identifying important features for classification.

Overall, the proposed technique offers a viable solution to the class imbalance issue and may enhance the effectiveness of classifiers developed using imbalanced datasets.

2 Related Work

Class imbalance is a prevalent issue in various fields, including business, medical diagnosis, fraud detection, digital marketing, and telecommunications. The synthetic minority over-sampling technique (SMOTE) is a widely used method for addressing this problem. However, limitations of SMOTE have been discovered over time, which have led to the development of several extensions of the original technique.

The Borderline-SMOTE technique [2] improves the performance of SMOTE in imbalanced datasets with a small number of borderline cases by only generating synthetic samples closer to borderline instances. Safe-Level SMOTE [3] addresses the problem of generating noisy synthetic samples by using a safe-level to measure the degree of noise in each generated sample. The MWMOTE technique [4] has been shown to outperform SMOTE and other extensions on a variety of datasets because it takes into account both the density and distribution of minority class instances when generating synthetic samples. The Adaptive Synthetic Sampling (ADASYN) technique [5] adjusts the weights of minority class instances based on their level of difficulty in learning, and has been shown to outperform SMOTE and other extensions on imbalanced datasets with varying degrees of imbalance. The G-SMOTE technique [6] generates synthetic samples using a Gaussian distribution instead of the Euclidean distance used in SMOTE and has been shown to outperform SMOTE and other extensions on imbalanced datasets with different levels of complexity.

Recently, two new extensions of SMOTE have been proposed. SMOTE-GN [7] uses a geometric approach to generate synthetic examples while taking into account the variance of the data. SMOTE with Topological Features [8] combines topological features with the synthetic example generation process to improve the performance of the algorithm. [5] provide a comprehensive survey of resampling methods for imbalanced data classification, including SMOTE and its variants, and discuss their strengths and weaknesses.

DataBoost-IM is a method that integrates data resampling with boosting algorithms to generate new minority class samples while preserving the original data distribution. It has been shown to outperform traditional data resampling techniques like SMOTE on several imbalanced datasets [9]. CURE-SMOTE is an improved version of the Borderline-SMOTE algorithm that clusters minority class samples before generating synthetic samples. It has been shown to deal efficiently with both the

imbalanced classes and clustering issues in imbalanced datasets [10].

Another oversampling technique is ROSE, which stands for Random over-sampling Examples, and involves randomly duplicating minority class samples to increase their representation in the dataset. It can be used as an alternative to SMOTE and has been proven to be successful in addressing class imbalance [11]. MDO-SMOTE means Multi-Domain over-sampling with SMOTE. It makes new training points in more than one domain. It does this using SMOTE. Studies have shown that MDO-SMOTE can improve how well SMOTE works. This is especially true when the data are imbalanced and there are multiple domains [12, 18].

Recent updates to SMOTE have helped with some of the older method's limits. In many cases, they also give better results when the data is imbalanced. Still, there is no single best version for every case. Which extension you pick should fit the dataset details and the specific goal you have.

3 Motivation

In related work, different over-sampling methods have been proposed to resample the data distribution. However, SMOTE and its variants have sometimes been found to suffer from problems. In this section, these situations are described and the reason why the CA-SMOTE algorithm is proposed is explained.

Issues with SMOTE-based algorithms are as follows:

1. Generated instances penetrating majority domain (SMOTE)

SMOTE and some its variants use k-nearest neighbors of minority class to interpolate and create synthetic samples. They create this samples by selecting a minority instance, finding its k-nearest minority neighbors and creating synthetic samples on line connecting them. This can be formulated as:

$$x_{\text{new}} = x_{\text{Selected}} + \alpha \cdot (x_{\text{NN}} - x_{\text{Selected}}) \quad (1)$$

Here, α is a random number in the range [0, 1].

But this can encounter several problems. For example, If the x_{Selected} is in the majority domain, then the probability that x_{new} is in majority domain increases. The same is true for the majority domain's nearest neighbor or both of them. These situations provide a noisy synthetic sample that may have an impact on the classification model's decision boundary. Figure 1 shows how sample is being created in majority domain while Figure 2 shows in minority domain.

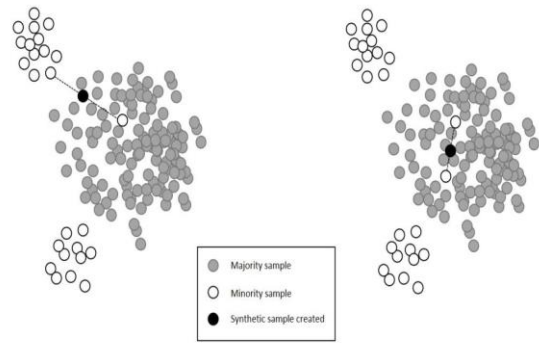


Fig. 1: Creation of Synthetic sample in Majority domain

Source: Created by Ruturaj Mahajan

2. Over-fitting due to duplicated samples (SMOTE and variants)

Most of the datasets tend to have minority clusters. These clusters are dense with minority samples at the center. When SMOTE and some of its variants are applied to such datasets, the majority of synthetic minority samples that are created tend to be at the center of such clusters. These samples are nearly useless to classification model and tend to over-fit the model. In such scenarios minority samples which are harder (near the decision boundary) to learn get neglected.

And value of k is increased to create synthetic points between two distinct clusters, they probably end up in majority domain. This has resulted in creation of better algorithms like ADASYN and Borderline-SMOTE but this issues still occur frequently.

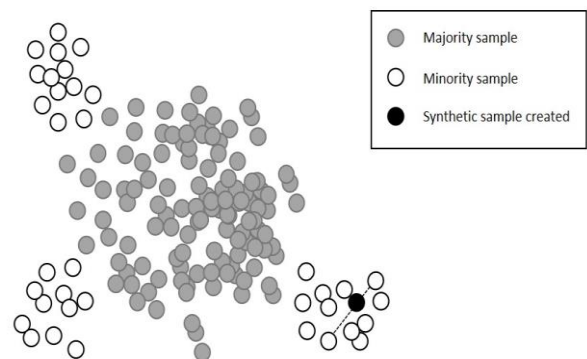


Fig. 2: Creation of duplicated samples

Source: Created by Ruturaj Mahajan

3. Bias towards noisy samples (ADASYN):

ADASYN (Adaptive Synthetic Sampling) is a variant of SMOTE that generates synthetic samples based on the density of minority class instances. The key idea is to generate more synthetic samples for

minority class instances that are harder to learn by the classifier. ADASYN's focus on hard-to-learn minority class cases, which may include noisy data, is one of its disadvantages. The classification performance may suffer as a result of this bias toward creating artificial samples that are not typical of the minority class distribution. Moreover, ADASYN may still suffer from the same limitations as SMOTE when the minority class and majority class are not sufficiently separated from each other.

4. Ambiguous Decision boundary (Borderline-SMOTE, Safe-level SMOTE):

Some variants of SMOTE that generate synthetic samples only for minority class instances that are close to the majority class like Borderline-SMOTE. However, they have some limitations that can affect its performance in certain cases. One of the main limitations of this approach is that it assumes that the boundary between the minority and majority classes is well-defined and can be easily identified. In cases where the boundary is not well-defined, Borderline-SMOTE may generate synthetic samples that are not located in the optimal region for improving the classification performance.

5. Cost of Computing (MWMOTE, G-SMOTE):

MWMOTE, which combines SMOTE with a modified version of distance-weighted minority oversampling, can generate high-quality synthetic samples that are closer to the true distribution of the minority class. However, it may be computationally expensive for large datasets, and its performance may degrade if the distance metric used is not appropriate for the data.

Same can be said about G-SMOTE. G-SMOTE generates synthetic samples by interpolating between pairs of minority class instances that form a convex hull in the feature space. This results in more diverse and representative synthetic samples that better reflect the underlying structure of the minority class. Due to the need to identify the appropriate density distribution for each minority class instance, the process of generating synthetic samples can be time-consuming and resource-intensive, particularly for large datasets. This can make G-SMOTE less practical for real-time or high-throughput applications where efficiency is a key concern.

4 The Proposed Method

To address these issues with existing methods, the CA-SMOTE resampling technique is proposed. The CA-SMOTE technique, which is a further

improvement of SMOTE and its variants, has the primary objectives:

- 1) To avoid the effects of class mixing due to outlier samples.
- 2) To focus on minority samples near supposed decision boundary of model for sample creation to help classification model.
- 3) To increase the variety of generated minority samples.

4.1 The CA-SMOTE Algorithm

SMOTE is similar to random over-sampling in the way it eliminates class imbalance by creating new minority points. Additionally, the synthetic samples created are not exactly the same as the original samples, but samples created on a line between two minority class instances, it improves minority diversity. However, SMOTE sometimes mixes classes. Some of the new synthetic points created affect the decision boundary severely.

Other point to note is that, SMOTE only deals with minority class and hence lack the information about majority class. Hence in some scenarios SMOTE creates either useless or wrong synthetic samples.

The proposed CA-SMOTE technique overcomes these problems using various means. The steps to this CA-SMOTE technique can be explained as follows:

1. Creation of borderline instance Set $S_{border1}$ and $S_{border2}$:

To make sure that CA-SMOTE technique only creates useful samples, it only takes in consideration the samples near decision boundary. This samples are selected with a very easy criteria: If they have one of their k nearest neighbor of opposite class

- The borderline samples of minority class are placed in set $S_{border1}$ and
- That of majority class are placed in $S_{border2}$.

This make sure that only samples useful to model's learning are created.

2. Removal of outliers:

Outliers can affect the decision boundary and hence should be taken care of. In the CA-SMOTE technique identifies outliers on criterion of k -nearest neighbors.

If all of sample's k -nearest neighbors are from the opposite class, the specimen is regarded as an outlier and is excluded from the process. This reduces CA-SMOTE chances of wrong learning of model and class mixing as well. The Figure 3 shows the outliers in data distribution.

3. Minority point Synthesis:

Contrary to original SMOTE and its variants, CA-SMOTE technique takes class of nearest neighbors in consideration. Based on type of nearest neighbor sample CA-SMOTE technique interpolates synthetic point differently. The CA-SMOTE technique selects random minority sample from $S_{border1}$ and performs one of following actions:

- If the nearest neighbor is a fellow sample from $S_{border1}$, it interpolates the synthetic sample like normal smote, on the line connecting selected sample and the nearest neighbor.
- If the nearest neighbor is a fellow sample from S_{maj} which is a majority point, it interpolates the synthetic sample on the line connecting selected sample and that nearest neighbor. But the main difference here is that synthetic sample does not lies on line between two points but is on exact opposite side of nearest neighbor. This makes sure that classes are not mixed up and strengthen the decision boundary. But to make sure that sample does not enter majority domain, the point is plotted near to selected minority point.

Figure 4 shows how synthetic sample is created with selected minority samples and its nearest neighbors

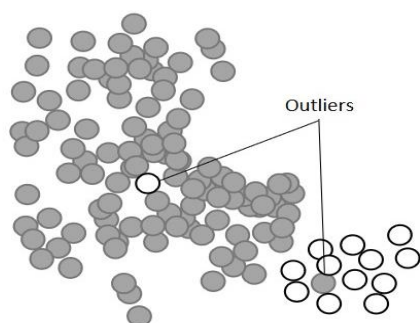


Fig. 3: The Outlier samples

Source: Created by Ruturaj Mahajan

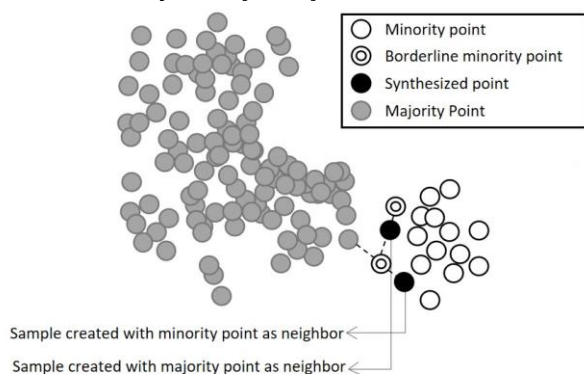


Fig. 4: Creation of synthetic samples using minority-class selected sample

Source: Created by Ruturaj Mahajan

4. Majority point Synthesis:

Very different to the original SMOTE and its variants, the CA-SMOTE technique also synthesizes minority points with respect to majority point. The majority samples from $S_{border2}$ are used to interpolate synthetic samples with another samples. This step also takes class of nearest neighbors in consideration. Based on type of nearest neighbor sample CA-SMOTE technique interpolates synthetic point differently. The CA-SMOTE technique selects random majority sample from $S_{border2}$ and performs one of following actions:

- If the nearest neighbor is a sample from $S_{border1}$, it interpolates the synthetic sample like normal smote, on the line connecting selected sample and its one of its k-nearest neighbor. But nearer to this neighbor minority sample.
- If the nearest neighbor is a fellow sample from S_{maj} which is a majority point, it interpolates the synthetic sample on the line connecting selected sample and that nearest neighbor. But the main difference here is that synthetic sample does not lies on line between two points but is on exact opposite side of nearest neighbor. This specific step ensures that classes are separated but very closely.

Figure 5 shows how synthetic sample is created with selected majority samples and its nearest neighbours

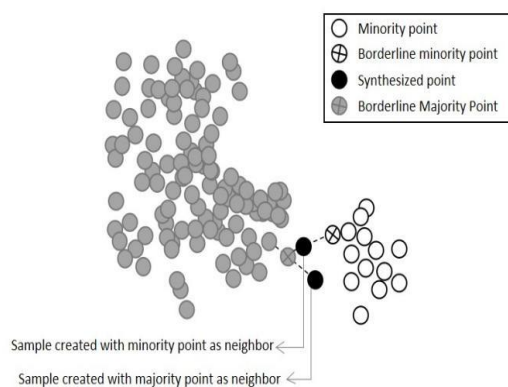


Fig. 5: Creation of synthetic samples using majority-class selected sample

Source: Created by Ruturaj Mahajan

The focus on borderline majority points make this technique more aware of majority class when considering of creating synthetic minority samples.

4.2 The CA-SMOTE Algorithm Pseudo-code

Algorithm 1 CA-SMOTE algorithm

```

1: function CA-SMOTE(Dtr, N, k)
   Input: Dtr; N; k. #Dataset, Percentage of oversampling, #nearest neighbors
   Output: synthetic minority class samples
   Variables:
   Smin(i): array for original minority class samples;
   Smaj(i): array for original majority class samples;
   Sborder1(i): Borderline minority instance set;
   Sborder2(i): Borderline majority instance set;
   Sborder(i): Borderline instance set;
   Newindex: keeps a count of synthetic samples generated, initialized to 0;
   Synthetic(i): array for synthetic samples
2: for i = 1 to len(Smin) do
   n = 0
   Compute k nearest neighbors for Smin(i), and save the indices in the nnarray
   for nn = 1 to k do
     if nn ∈ Smaj(i)
       n = n + 1
     end if
   end for
   if n = k then
     Smin(i) = Smin(i) - Smin(i)
   else if 0 < n < k then
     Sborder1(newindex) = Smin(i)
     newindex = newindex + 1
   end if
   Newindex = 0
   end for
3: for i = 1 to len(Sborder1) do
   nn_maj is user defined function to find nearest majority neighbor of given instance
   Sborder2(newindex) = nn_maj(Sborder1(i))
   newindex = newindex + 1
   end if
   end for
4:
   Sborder(i) = concatenate(Sborder1(i), Sborder2(i))
   X is number of synthetic samples to be created per Sborder sample
    $X = \lceil N/100 * (\text{len}(\text{Smin}) - \text{len}(\text{Smaj})) \rceil$ 
   len(Sborder)
5: for i = 1 to len(Sborder) do
   Compute k nearest neighbors for Sborder(i), and save the indices in the NNarray
   POPULATE(X, i, NNarray)
   end for

```

```

1. function Populate(X, i, NNarray)
2. Input: N; i; NNarray. #instances to create, original sample index, array of nearest neighbors
3. Output: N new synthetic samples in Synthetic array
4. while X ≠ 0 do
5.   NN = random(1, k)
6.   numattrs = Number of attributes
7.   for attr = 1 to numattrs do
8.     Compute: dif = Sample(nnarray[nn])(attr) - Sample(i)(attr)
9.     if Sborder(i)(attr) ∈ Sborder1(i) then
10.      if NN ∈ Sborder1(i) then
11.        Synthetic(newindex)(attr) = Sample(i)(attr) + random(0,1) · dif
12.      else if NN ∈ Smaj(i) then
13.        Synthetic(newindex)(attr) = Sample(i)(attr) - random(0,0.5) · dif
14.      end if
15.    else if Sborder(i)(attr) ∈ Sborder2(i) then
16.      if NN ∈ Sborder1(i) then
17.        Synthetic(newindex)(attr) = Sample(i)(attr) + random(0.2,1) · dif
18.      else if NN ∈ Smaj(i) then
19.        Synthetic(newindex)(attr) = Sample(i)(attr) - random(0.5,1) · dif
20.      end if
21.    end for
22.    newindex + +
23.    X - -
24.  end while
25. end function

```

The algorithmic steps of synthetic sample creation are majorly based on Community-Score (CS) factor (Number of instances in KNN from minority class) which justifies the displacement factor as follows:

- M_n : Sample under consideration (Minority class)
 - T_{cs} : Total CS of all KNN instances and M_n
 - $CS[a]$: Individual CS values in an array of all KNN instances and M_n
 - N_v : Normalized value (0 to 1 and $\sum N_v[a] = 1$)
 - Dis_{fac} : Displacement factor
1. Calculating N_v of M_n and its KNN instances

$$N_v[a] \leftarrow CS[a] / T_{cs}$$
 2. Fixing the Dis_{fac} for over sampling process:
 Situation 1:

$$N_V [M_n] = N_V [KNN(M_n)]$$

$$Dis_{fac} \leftarrow 0.5$$

Situation 2:

$$N_V [M_n] > N_V [M_n]$$

$$Dis_{fac} \leftarrow N_V [KNN (M_n)]$$

Situation 3:

$$N_V [M_n] < N_V [KNN (M_n)]$$

$$Dis_{fac} \leftarrow 1 - N_V [KNN (M_n)] // \text{ if } N_V [M_n] < 0.5, \text{ else } N_V [KNN (M_n)]$$

The Community-Score (CS) factor-based over sampling justify an invulnerable displacement factor implicitly addressing the issues of decision boundary samples.

Research Methodology

The methodology section outlines the method used to assess CA-SMOTE's performance and compare it with other widely used techniques. The evaluation was performed on different datasets, each of which was over-sampled using the new technique and the other techniques.

Once the datasets were over-sampled, classification models were fitted to the resulting data. The performance of each model was then evaluated using standard metrics.

To establish a fair comparison of the various methods, the same hyperparameters were used for each classifier in each experiment. Additionally, the performance of each oversampling technique was evaluated on both SVM and RF models to assess its effectiveness across different classifiers. Figure 6 shows exact flow of procedure.

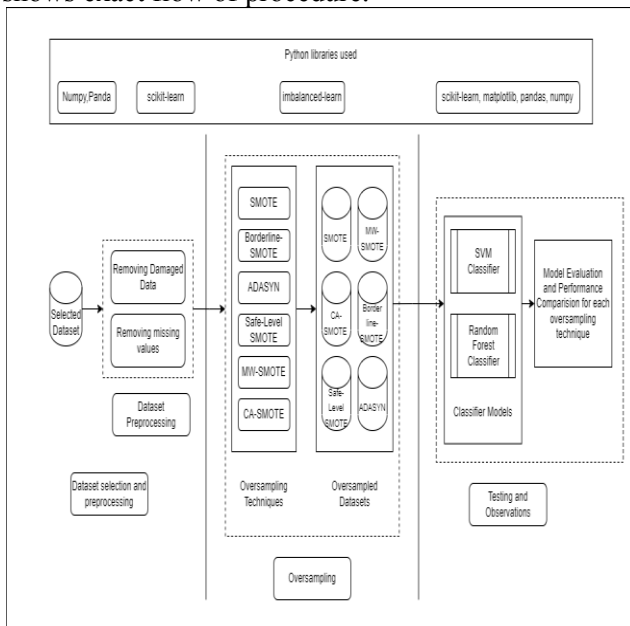


Fig. 6: Experimental workflow

Source: Created by Sachin Patil, Vilabha Patil

Dataset Selection and Preprocessing:

The CA-SMOTE over-sampling technique is evaluated on four different datasets, including: the Pima diabetes dataset, the Haberman dataset, the HTRU2 dataset, and the UAH driving dataset. The Pima diabetes dataset [14] contains data on medical conditions of females of Pima Indian heritage, while the Haberman dataset contains data on breast cancer patients [15]. The HTRU2 dataset includes pulsar star detection data [16], and the UAH driving dataset contains driving data collected from a driver's simulator [17]. Some modifications had been made to the UAH driving dataset to use it in this work. Table 1 shows the properties of datasets selected.

Table 1. Imbalanced datasets

Dataset Name	Number of Instances	Number of Features	Number of Classes	Class Distribution
Pima diabetes dataset	768	8	2	Class 0: 500 (65.1%) Class 1: 268 (34.9%)
Haberman dataset	306	3	2	Class 0: 225 (73.5%) Class 1: 81 (26.5%)
HTRU2 dataset	17,898	8	2	Class 0: 16,259 (91%) Class 1: 1,639 (9%)
UAH driving dataset	6168	4	3	Class 0: 3667 (59.4%), Class 1: 2303 (37.33), Class 2: 198 (0.032)

Source: Created by Sachin Patil

The Numpy and Panda libraries are used here as they are the most common libraries for data cleaning and preprocessing.

Over-sampling Techniques:

A number of over-sampling methods are chosen for comparison including Borderline-SMOTE, ADASYN, Safe-Level SMOTE, MWMOTE with the proposed CA-SMOTE method. These methods were chosen because they have been compared in earlier studies and have been demonstrated to be

successful in resolving class imbalance in a variety of sectors.

One of the most popular over-sampling methods, SMOTE has been shown to be effective at addressing class imbalance. SMOTE has some shortcomings, such as producing noisy samples and over-sampling the minority class to the point of over-fitting. Borderline-SMOTE and Safe-Level SMOTE are extensions of SMOTE that seek to solve these limitations. In order to create synthetic samples, the MWMOTE over-sampling technique takes into account both the density and distribution of cases belonging to minority classes.

All the over-sampling techniques other than CA-SMOTE mentioned above can be found on imbalanced-learn library.

Model Selection:

Model selection is a crucial step in every machine learning study since it affects how well the model performs on unseen data. In this study, two classification algorithms are used namely Support Vector Machine (SVM) and Random Forest.

SVM is known for handling non-linearly separable data well and is robust to outliers. It finds a hyperplane that maximally divides the data points into their corresponding classes by maximizing the margin between the classes. SVM works well in high-dimensional domains and has a low processing cost for handling big datasets.

On the other hand, Random Forest [18] is an ensemble learning technique that creates multiple decision trees during training and outputs the class that represents the mean of the classes. Each tree in the forest is trained on a random subset of the data, and the final result is determined by the combined votes of all trees. Random Forest is useful for handling complex decision boundaries and is also able to handle missing values and outliers.

Both SVM and Random Forest are widely used and well-established algorithms for classification problems, and have been shown to perform well on a variety of datasets. Additionally, both algorithms can handle both linearly and non-linearly separable data, which is important in cases where the decision boundary between classes is complex and hence the selection of this models.

Scikit-learn library is the most used library to fit these well-known classifiers on selected dataset.

Performance Metrics:

The performance metrics chosen for evaluation are F-score, AUC-ROC, accuracy, recall, and precision. These metrics are widely used for evaluating

classification models and are particularly suitable for imbalanced datasets.

Accuracy is a deceptive measure in imbalanced datasets since it may be high because the majority class was accurately predicted while the minority class was incorrectly forecasted. In these situations, recall and precision are preferable metrics because they provide insight into how well the model performs on samples from minority classes.

Recall measures how many predicted positive samples were truly positive samples in the dataset by taking their ratio. It measures the model's ability to identify positive samples correctly. The ratio of correctly identified positive samples to all of the model's positive predictions, on the other hand, is known as precision. It measures how well the model is able to avoid false positives.

Nonetheless, both recall and precision depend on the threshold adopted to classify the sample as either positive or negative. The entire performance of the model across all thresholds is represented by AUC-ROC since it is a threshold-independent statistic. In imbalanced datasets, where the negative class samples outnumber the positive ones, the AUC-ROC checks the model's ability to differentiate between the two classes.

Another such measure that combines recall and precision into a single performance indicator is the F-score. To find the balance between recall and precision, the F-score is helpful.

Overall, by using a combination of these metrics, an evaluation of the performance of CA-SMOTE model on imbalanced datasets can be done to make more informed decisions about the effectiveness of different techniques.

Model Evaluation and Performance Comparison:

Each classifier uses two sets: training and testing. The training set is over-sampled, and the SVM/RF model is fitted using the over-sampled training set. The classifier is then fitted on the testing set (original dataset) using the performance indicators already discussed above. For each of the four datasets, this procedure is repeated.

To find the performance of this techniques on models, the Scikit-learn library is used as it helps to easily visualize the performance classifier model.

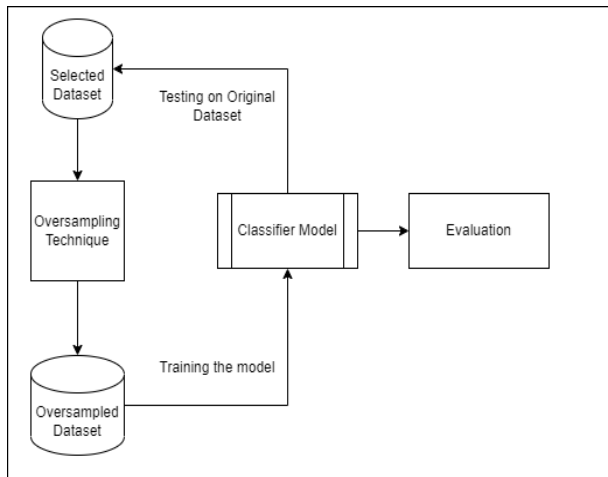


Fig. 7: Model Evaluation Process

Source: Created by Sachin Patil, Vilabha Patil

Experimental Results:

This section presents the performance of the suggested over-sampling methodology and compares it to a number of cutting-edge over-sampling methods, including SMOTE, Borderline-SMOTE, ADASYN, Safe-Level SMOTE, and Minority Weighted SMOTE. The effectiveness of these strategies will be assessed using the four imbalanced datasets viz. Pima Diabetes, Haberman, HTRU2, and UAH Driving.

Table 2 shows the notations used to denote the techniques used for comparison.

Table 2. Over-sampling techniques and Short Notations

Notation	Technique
a	SMOTE
b	Borderline SMOTE
c	ADASYN
d	Safe-level SMOTE
e	MW-SMOTE
f	CA-SMOTE

Source: Created by Sachin Patil

Pima Diabetes Dataset

The Pima Indians Diabetes dataset is a dataset often used for binary classification tasks. It contains data about female patients of Pima Indian heritage who have age of minimum 21, including features such as age, BMI, blood pressure, and the number of times a patient was pregnant, among others. The target variable is whether or not the patient has diabetes, with a value of 1 indicating that the patient has the condition and a value of 0 indicating that they do not. The Pima Indians Diabetes dataset can be found in UCI Machine Learning Repository. Table 3 shows performance metrics of two classification

models namely SVM and Random Forest on resampled Pima Diabetes dataset.

Table 3. Performance comparison of different techniques on Pima Diabetes dataset

Technique	Model	Accuracy	Precision	Recall	F-Score	AUC-ROC
a	SVM	0.750	0.672	0.732	0.698	0.824
	RF	0.752	0.675	0.728	0.702	0.827
b	SVM	0.760	0.686	0.724	0.710	0.832
	RF	0.762	0.690	0.730	0.712	0.835
c	SVM	0.730	0.641	0.719	0.677	0.794
	RF	0.734	0.645	0.722	0.680	0.798
d	SVM	0.760	0.680	0.764	0.706	0.820
	RF	0.762	0.683	0.768	0.710	0.822
e	SVM	0.760	0.672	0.763	0.711	0.831
	RF	0.762	0.674	0.761	0.712	0.832
f	SVM	0.780	0.720	0.774	0.739	0.854
	RF	0.782	0.722	0.776	0.742	0.856

Source: Created by Ruturaj Mahajan, Sachin Patil

Haberman Dataset

The Haberman dataset is a medical dataset that contains information about the survival of patients who had undergone surgery for breast cancer. It has four attributes including the age of the patient at the time of the operation, the year of the operation, the number of positive axillary nodes detected, and the survival status of the patient. Table 4 shows performance metrics of two classification models namely SVM and Random Forest on resampled Haberman dataset.

HTRU2 Dataset

The HTRU2 dataset is a binary classification dataset used for pulsar detection in radio astronomy. It describes the various characteristics of the pulsar candidates, such as their mean and standard deviation of profile and DM-SNR curve, and 1 binary class label indicating whether or not the candidate is a pulsar. The dataset is publicly available and has been used for research in machine learning, data mining, and astrophysics. Table 5 shows performance metrics of two classification models namely SVM and Random Forest on resampled HTRU2 dataset.

Table 4. Performance comparison of different techniques on Haberman Diabetes dataset

Technique	Model	Accuracy	Precision	Recall	F-Score	AUC-ROC
a	SVM	0.710	0.649	0.682	0.659	0.700
	RF	0.712	0.652	0.680	0.662	0.702
b	SVM	0.730	0.683	0.629	0.654	0.722
	RF	0.732	0.686	0.635	0.658	0.724
c	SVM	0.680	0.599	0.684	0.639	0.690
	RF	0.682	0.603	0.688	0.643	0.692
d	SVM	0.720	0.643	0.723	0.681	0.710
	RF	0.722	0.646	0.726	0.683	0.712
e	SVM	0.740	0.672	0.711	0.678	0.730
	RF	0.742	0.675	0.714	0.680	0.732
f	SVM	0.760	0.711	0.704	0.702	0.755
	RF	0.762	0.714	0.708	0.706	0.758

Source: Created by Ruturaj Mahajan, Sachin Patil

Table 5. Performance comparison of different techniques on HTRU2 Diabetes dataset

Technique	Model	Accuracy	Precision	Recall	F-Score	AUC-ROC
a	SVM	0.930	0.923	0.882	0.902	0.954
	RF	0.935	0.925	0.890	0.908	0.958
b	SVM	0.920	0.908	0.872	0.890	0.942
	RF	0.925	0.910	0.880	0.897	0.948
c	SVM	0.940	0.927	0.882	0.901	0.960
	RF	0.945	0.930	0.900	0.917	0.970
d	SVM	0.930	0.918	0.880	0.899	0.950
	RF	0.935	0.920	0.870	0.889	0.960
e	SVM	0.920	0.898	0.864	0.882	0.940
	RF	0.925	0.905	0.850	0.869	0.950
f	SVM	0.950	0.938	0.919	0.928	0.970
	RF	0.955	0.942	0.925	0.934	0.980

Source: Created by Ruturaj Mahajan, Vilabha Patil

UAH Drivers Dataset

The UAH driving dataset is a collection of data on driver behavior and vehicle sensor measurements during a series of simulated driving scenarios. The dataset includes data including measurements of vehicle speed, acceleration, and steering angle, as well as driver behavior metrics such as reaction time and gaze behavior. The goal of the dataset is to enable the development of intelligent transportation systems and to improve understanding of driver behavior. Table 6 shows performance metrics of two classification models namely SVM and Random Forest on resampled UAH driving dataset.

Table 6. Performance comparison of different techniques on UAH Diabetes dataset

Technique	Model	Accuracy	Precision	Recall	F-Score	AUC-ROC
a	SVM	0.800	0.751	0.782	0.760	0.727
	RF	0.805	0.756	0.776	0.744	0.730
b	SVM	0.811	0.759	0.773	0.756	0.719
	RF	0.815	0.764	0.784	0.758	0.726
c	SVM	0.790	0.742	0.764	0.734	0.716
	RF	0.795	0.747	0.766	0.739	0.711
d	SVM	0.810	0.762	0.782	0.760	0.727
	RF	0.813	0.768	0.780	0.760	0.733
e	SVM	0.800	0.752	0.772	0.740	0.716
	RF	0.805	0.758	0.766	0.748	0.720
f	SVM	0.820	0.772	0.792	0.770	0.741
	RF	0.825	0.778	0.800	0.780	0.751

Source: Created by Ruturaj Mahajan, Vilabha Patil

Hypothesis Testing for Means (z Test)

Low classification performance is projected by an unbalanced data collection. Different results may result from variations in the ratio of majority to minority cases per sample indicating imbalance within the data set. The structured z Test accurately predicts the changes in mean vs sample size that implicitly affect categorization accuracy.

A total of 33 random samples are used to count 315 out of 768 cases for CA-SMOTE in an oversampled Pima Diabetes dataset (Table 3). The test's specifics are provided as:

- Assumptions
 - $\alpha = 0.05$ // Level of significance

- $\mu = 0.70$ // mean of F-score values overall techniques across SVM {Table 3}
- 1. Claim Statement:
 H_0 : Average GM Values ≤ 0.70
 H_a (claim): Average GM Values > 0.70
- 2. The experimental outcomes of CA-SMOTE for managing bootstrapped samples of the Pima data set for F-score values are shown in Table 7.

Table 7. F-score Values (z test)

Sample no.	F-score	Sample no.	F-score	Sample no.	F-score
1	0.71	13	0.71	25	0.72
2	0.72	14	0.73	26	0.71
3	0.74	15	0.70	27	0.70
4	0.70	16	0.71	28	0.73
5	0.72	17	0.72	29	0.70
6	0.72	18	0.72	30	0.72
7	0.71	19	0.70	31	0.71
8	0.73	20	0.73	32	0.72
9	0.74	21	0.71	33	0.70
10	0.70	22	0.70		
11	0.71	23	0.73		
12	0.72	24	0.72		
Sample Mean (μ_s)					0.73
Samples Standard Deviation (σ_s)					0.011

Source: Created by Ruturaj Mahajan, Sachin Patil

- 3. Calculate the test statistics (z)

$$z = (\mu_s - \mu) / (\sigma_s / \sqrt{n})$$

$$= (0.73 - 0.70) / (0.011 / \sqrt{33})$$

$$= -0.03 / 0.019$$

$$= -1.57$$
- 4. P-value = Find area that relates to z
 $P\text{-value} < 0.04$
- 5. As P-value is less than α : $0.04 < 0.05$, the decision is to reject H_0
 - Interpretation: The claim that CA-SMOTE performs better than other methods is supported by sufficient evidence at $\alpha = 0.05$, which also claims that differences in sample means are the reason for performance variance and that imbalanced data samples result in poorer classification performance.

Overall, the proposed over-sampling technique outperformed all selected modern over-sampling techniques on all the examined datasets in terms of AUC-ROC score, F-score, accuracy, precision, and recall. These findings imply that the suggested technique may be a useful method for dealing with unbalanced datasets in classification problems.

Following graphs (Figure 7, Figure 8, Figure 9, Figure 10, Figure 11, Figure 12, Figure 13, Figure 14 and Figure 15) show the experimental results of Over-sampling techniques fitting two models on each dataset diagrammatically.

Performance metrics of SVM model on Pima Indian Database

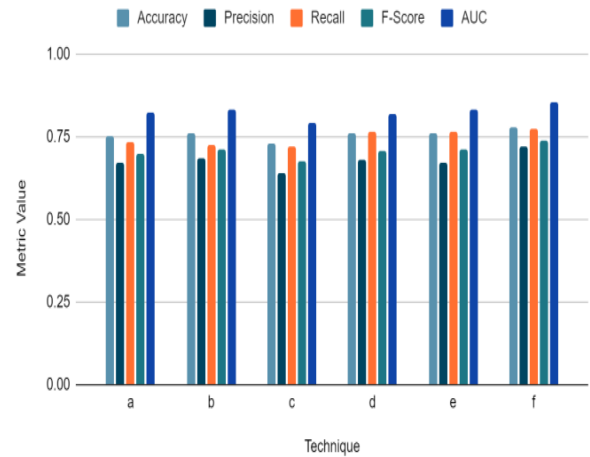


Fig. 8: Performance of SVM classifier on Pima Indian dataset

Source: Created by Vilabha Patil

The Pima Diabetes dataset table shows the performance of two techniques, SVM and RF, on six different techniques (a-f). RF classifier gave slightly better performance in terms of accuracy, precision, recall, F-score, and AUC-ROC than SVM. The highest metrics were achieved by the Random Forest classifier with CA-SMOTE(technique f).

Performance metrics of Random Forest model on Pima Indian Database

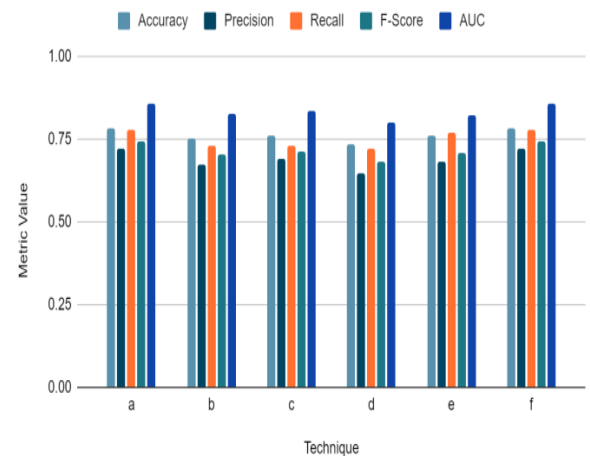


Fig. 9: Performance of RF classifier on Pima Indian dataset

Source: Created by Vilabha Patil

Performance metrics of SVM model on Haberman Database

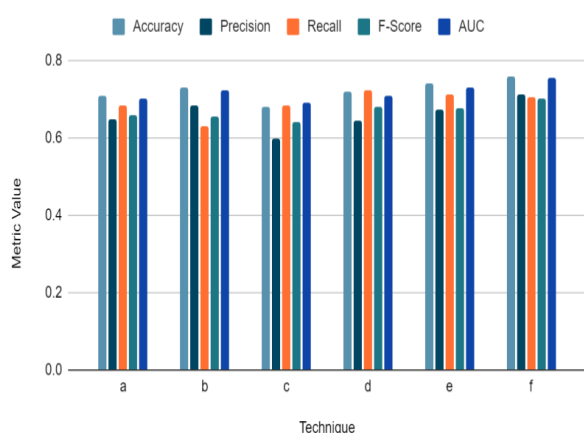


Fig. 10: Performance of SVM classifier on Haberman dataset

Source: Created by Vilabha Patil

Performance metrics of Random Forest model on HTRU Dataset

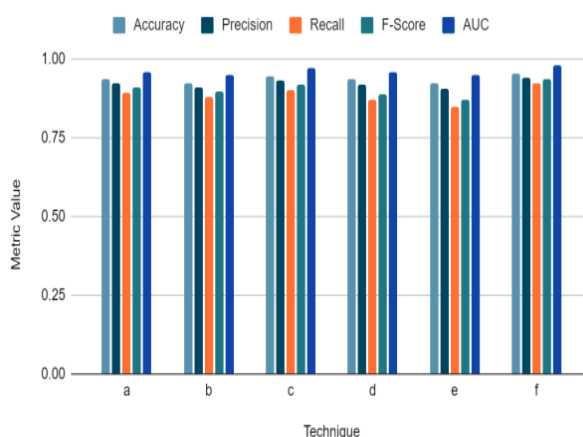


Fig. 13: Performance of RF classifier on HTRU2 dataset

Source: Created by Ruturaj Mahajan

Performance metrics of Random Forest model on Haberman Database

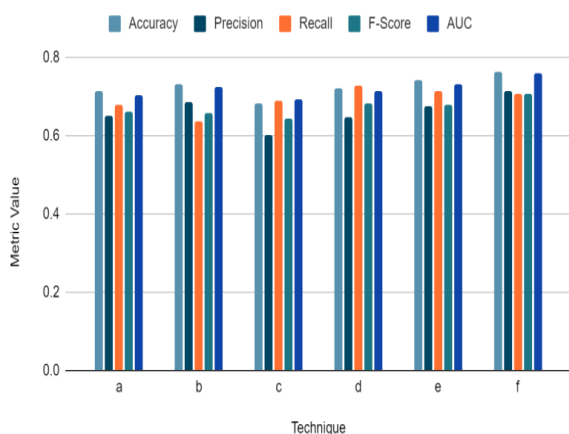


Fig. 11: Performance of RF classifier on Haberman dataset

Source: Created by Vilabha Patil

Performance metrics of SVM model on HTRU Dataset

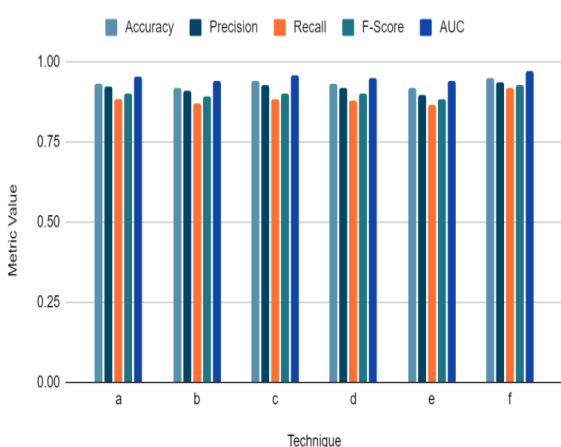


Fig. 12: Performance of SVM classifier on HTRU2 dataset

Source: Created by Ruturaj Mahajan

Performance metrics of SVM model on UAH driving Dataset

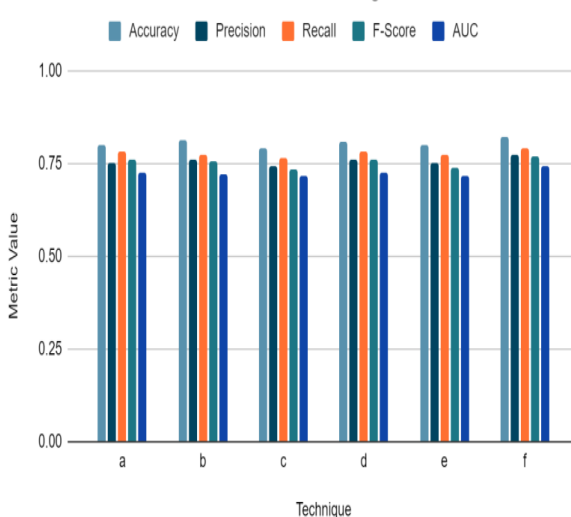


Fig. 14: Performance of SVM classifier on UAH driving dataset

Source: Created by Ruturaj Mahajan

The Haberman Breast Cancer dataset table also shows the slightly better performance of Random Forest classifier than SVM. The highest metrics achieved are again in by CA-SMOTE with Random Forest model with exception being the Recall.

The HTRU2 dataset table shows the highest classification performance than other datasets, where both classifiers achieve high accuracy, precision, recall, F-score, and AUC-ROC. CA-SMOTE achieved slightly better classification than other techniques.

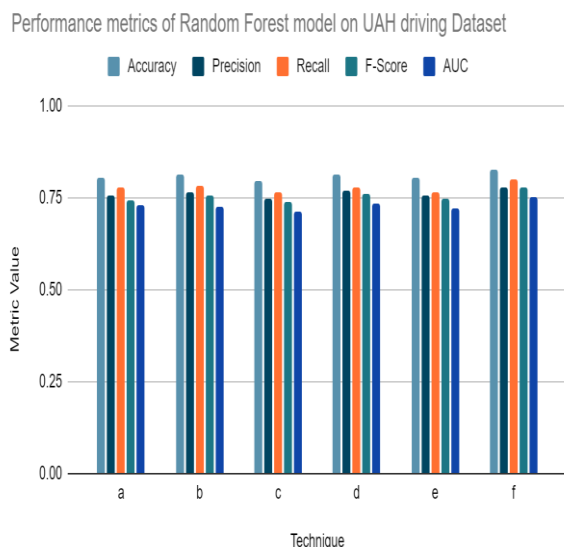


Fig. 15: Performance of RF classifier on UAH driving dataset

Source: Created by Ruturaj Mahajan

The UAH Drivers dataset follows same trend as above datasets. Overall, the tables suggest that both SVM and RF models perform similarly on the given datasets, with Random Forest giving slightly better performance.

Time Complexity Analysis (RF Classifier and HTRU2 dataset):

The execution times of various oversampling techniques utilizing RF classifier over HTRU2 dataset is shown in Table 8. SMOTE has the quickest execution times. Regarding execution time, MW-SMOTE and ADASYN takes longer time. The results highlight how crucial it is to take into account both execution time and predictive performance when choosing a classifier and oversampling technique. Faster execution rates are advantageous, but it's important to consider how they affect handling class imbalance and accuracy in general. By maximizing model efficiency without sacrificing accuracy, researchers and practitioners can use this knowledge to make well-informed decisions for their particular situation.

The time complexity of almost all techniques is quadratic except ADASYN and MW-SMOTE. The implementation and the particulars of the dataset, however, may affect the actual computing complexity.

Table 8. Time Complexity Analysis

Sr. No.	Technique	Time Complexity	Average Runtime (Sec)
1	SMOTE	$O(n)$	0.03273
2	Borderline SMOTE	$O(n^2)$	0.03421
3	ADASYN	$O(n^3)$	0.03784
4	Safe-level SMOTE	$O(n^2)$	0.03379
5	MW-SMOTE	$O(n^3)$	0.03812
6	CA-SMOTE	$O(n^2)$	0.03381

Source: Created by Ruturaj Mahajan

Discussion:

The tables (Table 3, Table 4, Table 5 and Table 6) demonstrate the performance of various oversampling techniques on various imbalanced datasets using two classification models viz. SVM and Random Forest. The evaluation metrics used are accuracy, precision, recall, F-score, and AUC-ROC. Experimental analysis depicts that CA-SMOTE had higher performance than others. CA-SMOTE focused on creating synthetic samples only near decision boundary that are important for model, rather than creating synthetic samples for all minority class instances. It includes those that may not contribute much to the decision boundary. CA-SMOTE also created samples using majority samples, which resulted in more diverse samples that are near to decision boundary but still avoided the class mixing problem. It also reduces limitations of k-hyperparameter to some extent. Due to its ability to remove outliers, CA-SMOTE minimizes the generation of noisy samples. This helped CA-SMOTE to achieve better performance than other oversampling techniques. In all the datasets, the HTRU2 dataset gave very high performance after oversampling indicating a lack of randomness and high correlation between features. Overall, the study shows the effectiveness of the proposed oversampling method, which outperforms several state-of-the-art techniques. It can be noted that the oversampling technique used can have a significant impact on a classification model's performance on imbalanced datasets.

5 Conclusion

After conducting a comprehensive evaluation of various modern oversampling techniques on imbalanced datasets, it was observed that CA-SMOTE exhibited superior performance compared to all other methods in terms of AUC-ROC and F-score. The evaluation results consistently showed

that CA-SMOTE outperformed other techniques across both Random Forest and SVM models.

The superior performance of CA-SMOTE could be attributed to its ability to identify and utilize both minority and majority classes, removal of noisy samples and different methods for interpolating synthetic samples. The given datasets had different class imbalances. CA-SMOTE uses a combination of over-sampling and under-sampling based on class of samples to balance the classes, which resulted in better performance, justifying the name Class-Aware SMOTE. Other techniques that were used in this study did not utilize the majority class correctly, which could have impacted their performance.

It is worth noting that while CA-SMOTE may require more computational resources than some of the other techniques used in this study. Moreover, it is still more efficient than the modern oversampling techniques like MWMOTE and G-SMOTE. It is also important to mention that recent advancements in distributed parallel computing and processing have significantly reduced the computational costs. This shows that the computational costs associated with CA-SMOTE should not be a major concern for most applications, especially when considering the potential gains in performance.

However, there are still several potential areas for future research that could further improve the performance of CA-SMOTE. One such area is feature selection, as high-dimensional datasets may not be optimally handled by the technique. Additionally, ensemble learning methods, optimization of hyperparameters like k-nearest neighbors, integration with deep learning models are all potential research directions for improving CA-SMOTE. Addressing these areas could lead to even better performance and make CA-SMOTE an even more compelling option for handling class imbalance in real-world applications.

CA-SMOTE is a highly recommended approach for improving the classification performance of imbalanced datasets. Its ability to handle class imbalance using class awareness makes it a compelling option for classification tasks, and the significant performance gains achieved using CA-SMOTE make it a worthwhile investment.

References:

- [1] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P.. SMOTE: Synthetic minority over-sampling technique, *Journal of Artificial Intelligence Research*, Vol. 16, 2002, pp. 321–357.
- [2] H. Han, W. Wang, and B.-H. Mao, Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning, *International Conference on Intelligent Computing*, 2005, pp. 878-887.
- [3] K. Li, K. Liu, Y. Chen, and S. Zhang, Safe-Level-SMOTE: Safe-Level Synthetic Minority Over-sampling Technique for Handling the Class Imbalance Problem, *IEEE Access*, Vol. 7, 2019, pp. 108260-108272.
- [4] Chawla, N. V., S. M. Japkowicz, A. Kotcz, and C. D. A. Wojciechowski, MWMOTE—Majority Weighted Minority over-sampling Technique for Imbalanced Data Set Learning, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 26, No. 2, 2014, pp. 405-425.
- [5] He, H., Bai, Y., Garcia, E. A., and Li, S. (2008), ADASYN: Adaptive synthetic sampling approach for imbalanced learning, *In Proceedings of the IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 1322-1328.
- [6] Y. Zhang, X. Yang, and Y. Chen, G-SMOTE: A New over-sampling Method for Imbalanced Data Set Based on GMM, *International Conference on Natural Computation*, 2018, pp. 1-6.
- [7] D. Li, G. Gao, and F. Xu, SMOTE-GN: A Geometric Approach to SMOTE for Imbalanced Data Classification, *Information Sciences*, Vol. 540, 2020, pp. 109-125.
- [8] B. Torgo, P. Ribeiro, and I. Gama, SMOTE with Topological Features for Class-Imbalanced Domains, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 27, No. 8, 2015, pp. 2289-2301.
- [9] H. Guo and H. L. Viktor, Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach, *SIGKDD Explorations*, Vol. 6, No. 1, 2004, pp. 30-39.
- [10] G. B. Huang, Q. Y. Zhu, and C. K. Siew, Extreme learning machine: theory and applications, *Neurocomputing*, Vol. 70, No. 1-3, 2006, pp. 489-501.
- [11] Dong, Y., Zhang, X., and Liang, Y., Imbalanced data classification based on random over-sampling and under-sampling (ROSE) with clustering, *Knowledge-Based Systems*, Vol. 108, 2016, 64-82.
- [12] Khan, M. R., Shah, M., and Bennamoun, M., MDO-SMOTE: Multi-objective dynamic over-sampling for class imbalance problem in supervised learning, *Information Sciences*, Vol. 467, 2018, pp. 311-331.

- [13] Nursel S. R. and Clare C., Data Level Approach for Multiclass Imbalance Financial Data, *WSEAS Transactions on Computers*, Vol. 19, 2020, pp. 165-182.
- [14] Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S., Using the ADAP learning algorithm to forecast the onset of diabetes mellitus, *In Proceedings of the Symposium on Computer Applications and Medical Care*, IEEE Computer Society Press, 1988, pp. 261-265.
- [15] Haberman, S. J., Generalized Residuals for Log-Linear Models, *In Proceedings of the 9th International Biometrics Conference*, Boston: Boston University, 1976, pp. 104-122.
- [16] Keith, M. J., Data analysis using the HTRU2 pulsar dataset, *PhD thesis*, University of Manchester, 2010.
- [17] Chandrashekar, G., & Sahin, F., A survey on driving style analysis for intelligent transportation systems, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 19, No. 3, 2018, pp. 927-947.
- [18] Suvaporn H. and Krung S., A Random Forest with Minority Condensation and Decision Trees for Class Imbalanced Problems, *WSEAS Transactions on Systems and Control*, Vol. 16, 2021, pp. 502-507.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

- Dr. Sachin Patil: Conceptualization of technique & its algorithm
- Ruturaj Mahajan: Literature survey and Gap analysis, Overall scripting, Experimentation.
- Vilabha Patil: Statistical analysis, Manuscript orientation/flow

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0

https://creativecommons.org/licenses/by/4.0/deed.en_US