Synthetic Tabular Data Generation for Imbalanced Classification: The Surprising Effectiveness of an Overlap Class

Annie D'souza*, Swetha M*, Sunita Sarawagi

Indian Institute of Technology, Bombay
20d070028@iitb.ac.in, 23m0756@iitb.ac.in, sunita@iitb.ac.in

Abstract

Handling imbalance in class distribution when building a classifier over tabular data has been a problem of long-standing interest. One popular approach is augmenting the training dataset with synthetically generated data. While classical augmentation techniques were limited to linear interpolation of existing minority class examples, recently higher capacity deep generative models are providing greater promise. However, handling of imbalance in class distribution when building a deep generative model is also a challenging problem, that has not been studied as extensively as imbalanced classifier model training. We show that state-of-the-art deep generative models yield significantly lower-quality minority examples than majority examples. We propose a novel technique of converting the binary class labels to ternary class labels by introducing a class for the region where minority and majority distributions overlap. We show that just this pre-processing of the training set, significantly improves the quality of data generated spanning several state-of-the-art diffusion and GAN-based models. While training the classifier using synthetic data, we remove the overlap class from the training data and justify the reasons behind the enhanced accuracy. We perform extensive experiments on four real-life datasets, five different classifiers, and five generative models demonstrating that our method enhances not only the synthesizer performance of state-of-the-art models but also the classifier performance.

Code — https://github.com/Annie2603/ORD

1 Introduction

The problem of imbalance in class distributions is encountered in several fields, including fraud detection in the banking sector, disease diagnosis in the medical sector, and anomaly detection in finance, cybersecurity, and manufacturing industries. Imbalanced data refers to the distribution of classes or categories being highly skewed or disproportionate in a dataset, i.e., one class or category is significantly more prevalent than the others. The presence of imbalance introduces several challenges in training classification models as they are unable to learn the distribution of the rare class. This has been a problem of long-standing interest and many different techniques have been proposed ranging from cost-sensitive loss functions (Zhou and Liu 2005), oversampling minority instances and/or under-sampling majority instances (Li et al. 2021), and augmentation with synthetic instances (Johnson and Khoshgoftaar 2019). In this paper, we focus on data augmentation-based techniques since these are particularly effective (Chawla et al. 2002). One popular classical data augmentation method, SMOTE, generates new minority instances using a convex combination of existing minority instances. However, recent deep data generators yield even higher accuracy due to the enhanced quality of generated examples.

Our focus in this paper is tabular data, and recently many high-quality generators have been proposed for generating tabular data ranging from GAN-based models like CTabGAN (Zhao et al. 2021) to diffusion-based models like TabSyn (Zhang et al. 2024) and ForestFlow (Jolicoeur-Martineau, Fatras, and Kachman 2024) Most of these have been evaluated after training on balanced datasets, where their generated data have been shown to be highly effective in training classifiers that match the accuracy of real data. We are not aware of any study that evaluates them in highly imbalanced input data.

In this paper, we show that default training of generative models yields significantly worse quality minority instances than majority instances as measured by the label provided by an Oracle Bayesian model. Simple fixes to the problem by over and under-sampling initial real data distributions are not effective.

We present a method called ORD[1] that employs three key ideas to improve the training of imbalanced classifiers on tabular data using synthetic data generators: (1) We preprocess the available real data by identifying a small subset of the majority instances called the *overlap* set that lies on the boundary of majority and minority examples. (2) We modify the generator to be class conditional and generate three-way labeled instances corresponding to minority, overlapping majority, and clear majority. (3) We train the final classifier with a careful mix of real and synthetically generated instances comprising of balanced sample of synthetic data generated consisting of only the minority and clear majority classes, and real minority.

*These authors contributed equally.

[1]for Overlapped Region Detection

Our method is orthogonal to the type of data generator and classifier, and we evaluate it on four different tabular data generator architectures and five different classification models. We perform a set of insightful ablation studies to tease apart the reason why our method outperforms existing methods. We conclude that our method yields improved accuracy both due to the generator providing higher quality synthetic minority examples, and the classifier improving due to selective under-sampling of overlapping majority examples.

The main contributions of this paper are as follows:

- We evaluate state-of-the-art deep generative models for tabular data on imbalanced training data, and show that the quality of minority examples they generate is much worse than majority examples.
- We propose a new method called **ORD** that introduces a new class comprising of the subset of majority examples that lie in the boundary of the two classes, harnesses these to impose a ternary classification loss when training the generative model, and finally trains the classifier using generated examples that lie outside the overlap region.
- We propose a new SOTA model **CTabSyn**, a variation of TabSyn (Zhang et al. 2024) with the inclusion of ORD and conditional training and generation.
- We provide extensive experimental evaluation and proof that ORD provides significant increase in accuracy.
- We justify the enhanced accuracy by showing that our method both improves the quality of minority examples generated from our method, and the classifier gains from selective under-sampling of points in the overlap region.

2 Related Works

Traditionally, for training classifiers on imbalanced data, either the minority class is oversampled (Chawla et al. 2002) or majority class is undersampled. While efficient to some extent, majority under-sampling methods can discard potentially useful instances. Under sampling methods include cluster-based (Zhang, Zhang, and Wang 2010) under-sampling and Neighbourhood (Vuttipittayamongkol and Elyan 2020) based boundary undersampling. While over sampling methods include Tomek link based oversampling (Leng et al. 2024).Over-sampling can be ineffective when number of minority points is very small. In such cases, a more effective alternative is generating synthetic examples. A pioneering method in this category is SMOTE (for Synthetic Minority Oversampling Technique) (Chawla et al. 2002) that interpolates between minority points to generate new minorities. Several variations of SMOTE like BorderlineSMOTE (Han, Wang, and Mao 2005) and ADASYN (He et al. 2008), SMOTE Tomek (Wang et al. 2019) also exist.

Recently, deep generative models have provided much higher quality synthetic data across a variety of domains but we specifically focus on synthetic generators for tabular data. Conditional Tabular GAN (CTGAN) (Xu et al. 2019) is a Generative Adversarial Network based synthetic data generator. (Zhao et al. 2024) proposed to use density-based clustering to filter out noisy and boundary samples before feeding to CTGAN synthesizer. (Adiputra and Wanchai 2023) proposes to oversample minority with CTGAN and combine with existing majority points. CTABGAN (Zhao et al. 2021) and CTABGAN+ (Zhao et al. 2022) are improvements over CTGAN. They perform training by sampling based on frequency of discrete columns to handle imbalances. Among the transformer-based (Vaswani 2017) synthesizers, Generation of Realistic Tabular data i.e. GReaT (Borisov et al. 2023), TabuLA model (Zhao, Birke, and Chen 2023) and Tabular Masked transformers (Gulati and Roysdon 2023), TabLLM (Hegselmann et al. 2023) are noteworthy synthetic data generators. Normalizing flows are also used to estimate the data distribution (Papamakarios, Pavlakou, and Murray 2017).

Of late, diffusion-based models are providing higher quality generations of synthetic data. These include the Denoising diffusion probabilistic model TabDDPM (Kotelnikov et al. 2022), TabSyn (Zhang et al. 2024) a Latent diffusion model, and Forestflow (Jolicoeur-Martineau, Fatras, and Kachman 2024) based on XGBoost (Chen and Guestrin 2016). We experiment with several categories of existing generators for tabular data, and found ForestFlow and TabSyn to provide the best accuracy.

Most existing data synthesizers have not considered **imbalanced** data. Ours is the first work that harnesses these models for generating synthetic data in the imbalanced class setting. We show that even the training of the generators is affected by the data imbalance. Our method ORD can be applied to all of the existing synthetic generators to enhance its quality of imbalanced data.

3 Problem Statement

Let $X = X_1 \times \ldots \times X_d$ denote the space of d-dimensional record data where each X_j could be continuous or categorical. Let $Y = \{0, 1\}$ denote a binary class label space and $P(X, Y)$ denote their unknown joint distribution. We are provided a labeled dataset $D = \{(x_1, y_1) \ldots (x_N, y_N)\}$ from the distribution where the fraction of instances in D with label 1 (minority class) is significantly smaller than (order of 2%) N. Our goal is to train a classifier $M : X \mapsto Y$ that provides high accuracy separately for both the minority and majority class. Instead of depending on the given imbalanced data D alone, we seek to sample synthetic data D_S from a generative model G trained on D. The classifier may be trained on a mix of synthetic and real data, but it is always evaluated on real data. A baseline method is using any of the SOTA tabular data generators to sample a balanced and large number of majority and minority examples as D_S to train the classifier M. In the following section, we present our proposed method that improves upon this baseline.

4 The Overlap Region Detection (ORD) Method

One of the major sources of difficulty in training a classifier with highly imbalanced class distributions is that the discriminator fails to find any region where the minority examples are more prevalent than the majority examples. Our proposed ORD method addresses this challenge by increasing the concentration of minority examples in select

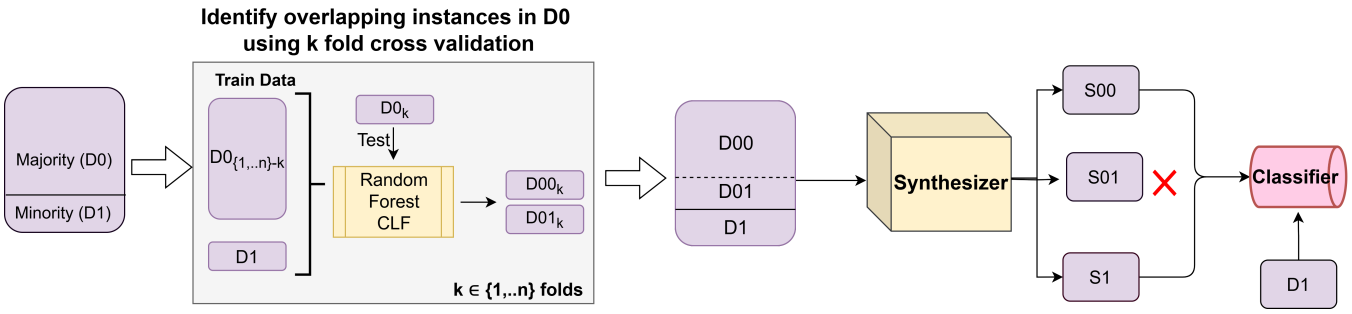

Figure 1: The method of ORD and its application to generate synthetic data. Synthetic data is then used to train a better classifier compared to when no ORD is used. Steps in the overall algorithm: 1. To use k-fold training to identify the overlap in the validation set. This uncertainty is labelled as a third class label D_{01}. 2. Generate synthetic data using the three class dataset instead of binary class. The synthetic data quality is much better as the distribution is better reproduced by Generative models. 3. Train the final classifier with equal proportions of majority and minority synthetic data while discarding the overlapped region D_{01} which makes learning the decision boundary easier.

regions via data generated from a deep generative model. We achieve this over three steps as outlined in Figure 1: (1) First, we identify a subset D_{01} of the majority instances D_0 in D that overlaps significantly with the minority examples. We use a time-tested k-fold cross-validation on a random forest classifier for this step, as we elaborate in Section 4.1. The output of this stage is a split of the majority instances D_0 into the overlapping majority D_{01} and clear majority $D_{00} = D_0 - D_{01}$. (2) Second, we train a conditional generator G on the ternary labeled data as elaborated in Section 4.2. We show that the introduction of a finer-grained class label dataset improves the quality of the generated examples in Table 4 (3) We train the final classifier M on synthetically generated instances S_1, S_{00} from G that are sampled in equal parts from the minority class and clear majority class, respectively. We show that excluding the overlap majority instances declutters the decision boundary and improves overall accuracy. We also found it useful to include only the minority instances D_1 in the training pool of M. The inclusion of real majority instances did not help improve the accuracy, possibly because the generator is well-trained on the majority instances. This is somewhat contrary to the results on balanced dataset where classifiers trained on real data outperform those on synthetic (Kotelnikov et al. 2022).

4.1 Identifying Overlapping Majority Instances

In this section, we describe the method to identify the subset D_{01} of majority instances D_0 that we call the overlapping instances. Since neighborhood-based methods are not meaningful for high dimensional data, we use the disagreement of ensemble classifiers like random forest (Breiman 2001) to identify overlap. We depend on k-fold cross-validation to define splits used to train the random forest and obtain disagreement scores. Since we need the scores only on majority instances, we split only the majority instances D_0 into k disjoint sets $F_1, \ldots, F_k$. For each fold j, we train a random forest RF_j on $D_0 - F_j$ instances as class 0 and D_1 as

Algorithm 1: Synthesis and Classification using ORD

Input: Imbalanced real data: majority D_0, minority D_1.
Parameter: Threshold τ, Folds k
Output: Classifier M.

Split D_0 into k folds $F_1, \ldots F_k$.
$D_{01}, D_{00} = \phi$
for $i = 1$ to k **do**
  Train RF_j on $D_0 \setminus F_j$ and D_1
  for $x \in F_i$ **do**
    confidence_maj = RF_j.predict_prob_0(x)
    if confidence_maj $\leq 1 - \tau$ **then**
      Add x to D_{01} with class 2
    else
      Add x to D_{00} with class 0
    end if
  end for
end for
Train G on ternary labeled data $D_{00} \cup D_{01} \cup D_1$
Sample S_{00}, S_1 from G for label 0 and 1.
Train M using $S_{00} \cup S_1 \cup D_1$ and return M

class 1. We then apply RF_j on the set-aside fold F_j and obtain the fraction $r_j(x)$ of committee members in the random forest that label each instance $x \in F_j$ as class 0. If there is high disagreement, that is $r_j(x) < 1 - \tau$, where τ is a given threshold, we identify x as lying in the overlap region and add it to the overlap set D_{01}. Across the k folds, each instance in D_0 gets evaluated for its potential of being in the overlap set D_{01}. The remaining majority instances are put in D_{00} which form the clear majority set. In Figure 3 the top left figure shows an original binary dataset with overlap points extracted. Note that the overlap points are nicely identified as being on the boundary of the majority and minority regions.

We take τ such that, the number of overlapped majority points = min(number of minority points, r% of majority points). The r above is generator dependent and we find it by checking the performance on the validation set of a dataset

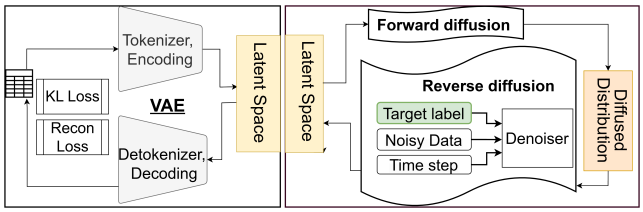

Figure 2: CTabSyn for class conditional tabular data generation. We needed to make only a small change (highlighted in green) over existing TabSyn model to improve the quality of generations for imbalanced class distribution and benefit from our finer-grained class labels. Conditional diffusion is implemented by adding the true target embedding as input to denoiser for efficient sampling.

with different r values (we used 3%, 5%, 7%, 9%) and use the best r value with all datasets for a given synthesizer.

4.2 Conditional Generators With Ternary Labels

Once we assigned finer-grained class labels to the original data D, we show in the experiment section that almost all existing deep generative models provided higher quality generators than training an identical model with binary labels. The only modification we made was to TabSyn (Zhang et al. 2024) one of the SOTA synthetic tabular data generation models which consists of a diffusion process modeled in a learnt latent space. The tabular data is first passed through a Variational auto-encoder which consists of a tokenizer that generates unique representations for each column which are then fed to a transformer to capture the intricate relations between the columns to obtain data in a continuous latent space instead of separately for numerical and categorical columns. This model does not support class conditional generation. We modified the TabSyn to support conditional generation and call this variant **CTabSyn**. The denoiser MLP which takes as input - a noisy version of data (x_{in}) and timestep (t) of the data now also has an embedding of the true target value (y) input to it.

$$\begin{aligned} t\_emb &= \texttt{TimeEmbedding}(t) \\ y\_emb &= \texttt{Embedding}(y) \\ x &= \texttt{Linear}(x_{in}) + t\_emb + y\_emb \end{aligned} \tag{1}$$

The overall architecture of CTabSyn is shown in Figure 2. CTabSyn with ORD considerably improves the performance of TabSyn as shown in Table 2.

5 Experiments

We present a detailed evaluation of the performance of ORD vis-a-vis several existing methods of generating synthetic data for training classifiers with imbalanced data. Our method is agnostic to the type of data generator and classifier, and we perform evaluation on five data generators and five different classifiers. More than the overall results, we dwell on the analysis of the reasons why ORD provides gains. Here we seek to answer questions like: Are ORD's gains due to better quality synthetic data generated due to training with finer-grained ternary labels? Or, is the main reason the selective under-sampling of majority points from the overlapping region?

Dataset	# Maj	# Min	# Unb Min	%Min/Maj
Adult	34514	11208	500	1.5
Heloc	5559	34979	700	2
Fintech	11560	11174	200	1.3
Cardio	35700	5000	100	1.8
Abalone	3076	265	265	8.6
Bank	31970	4198	4198	13.1
Car	1324	58	58	4.4
Yeast	1056	127	127	12

Table 1: Dataset details with count of original majority, original minority and the count of minority in the imbalanced setting with the Minority-Majority ratio for all datasets considered are depicted.

Datasets We evaluate our approach on eight real-world tabular datasets comprising a mix of numerical and categorical features, all with binary target variables. Four datasets—Adult, Heloc, Fintech Users, and Cardio—are originally less imbalanced. To simulate higher imbalance, we undersample the minority class to constitute only 1.5% to 2% of the total data. These datasets use a balanced test set for evaluation, while the training set is highly skewed.

Additionally, we use four inherently imbalanced UCI datasets—Abalone, Bank, Car, and Yeast—without introducing further skew. The test sets for these datasets retain their natural imbalance, reflecting real-world distributions. This combination of balanced and imbalanced test scenarios enables a comprehensive evaluation of the robustness and effectiveness of our method. The overall statistics of these datasets and their detailed descriptions including the number of D_{01} points for each threshold have been made available in the appendix.

Baselines Since ORD is an addendum over synthetic data generation models, we demonstrate the performance improvement it gives over five existing synthetic tabular data generation methods: **CTGAN, CTABGAN+, TabDDPM, ForestFlow and TabSyn**. In addition, from classical methods, we also include **SMOTE** and its popular variants i.e. **borderline SMOTE** (Han, Wang, and Mao 2005) and **AdaSyn**. (He et al. 2008)

Evaluation Metrics We follow standard practice for evaluating classifiers on imbalanced data and measure accuracy as the macro average of accuracy on **real** unseen minority and majority instances. This metric is also known as machine learning efficacy. We train five different classifiers: **XGBoost, Logistic Regression, Decision Tree, Multi-Layer Perceptron, AdaBoost**. However, to reduce clutter, we present accuracy grouped as follows: We separately report accuracy on XGBoost since it is generally the best-performing classification method for tabular data, and then we report the average accuracy over the remaining four classifiers. For the test set, we sampled 2000 of each class for the

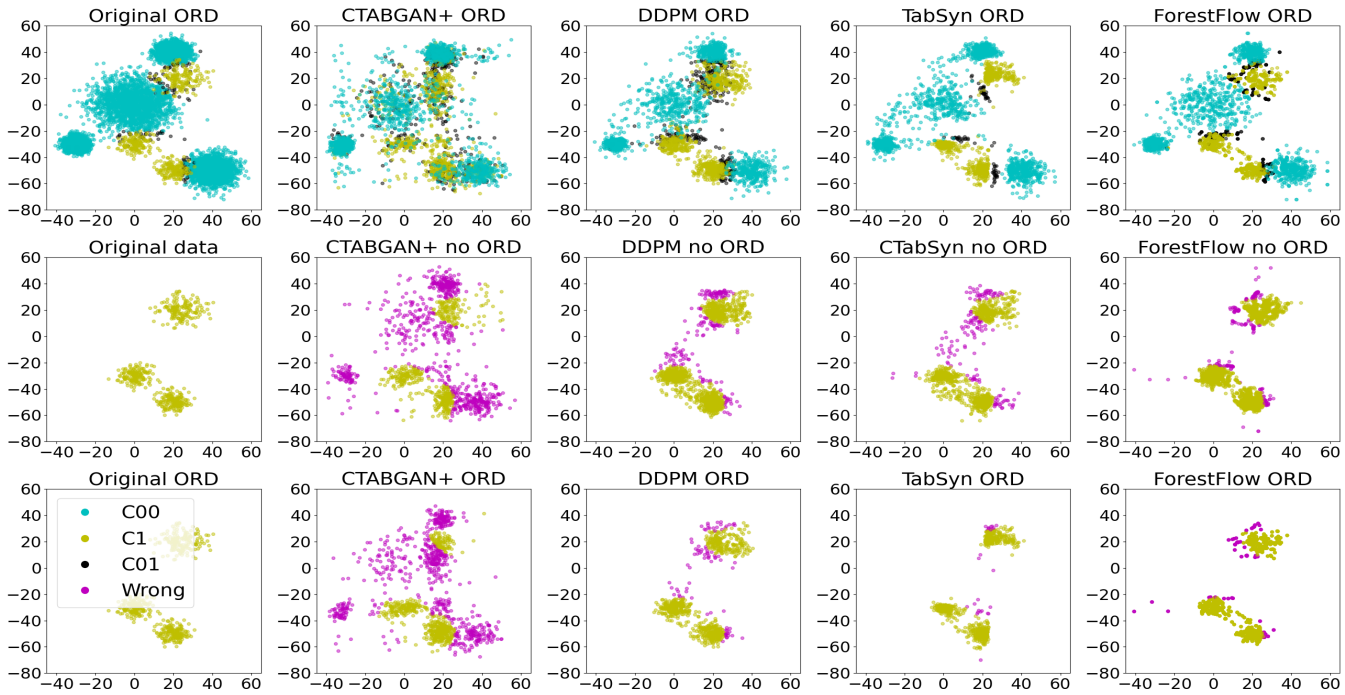

Figure 3: Visualisation of Synthetic data for ORD in 2D datasets. The first row shows data synthesized with ORD for different Synthesizers along with a clear indication of the overlap class D_{01}. The second row zooms only on sampled minority with wrong generations marked in pink from generators without ORD. The third row shows the same data but with ORD enhanced generators. The columns correspond to different generators. Particularly for the diffusion models (last two columns), ORD provides much fewer errors in generated examples than baseline diffusion models, which is already much higher quality than earlier GAN-based generators in columns 2 and 3.

first 4 datasets since they were created by under-sampling from the original datasets. Since the test sets of the UCI datasets are also imbalanced, we additionally report the minority & majority accuracies, F1 scores and AUCROC in Appendix.

Experimental Setup The code for all baseline synthesizers was obtained from their official GitHub repositories. All steps on environment creation, package installation, dataset pre-processing, training, and generation were followed as specified in the repository of the given baseline. Each baseline synthesizer underwent training and generation twice: Once with the original imbalanced dataset to synthesize D_0 and D_1 and once with the ORD processed imbalanced dataset to synthesize D_{00}, D_{01} and D_1. For each dataset, an equal number of D_1 and D_0 synthetic points were used for training the classifier in the case of both with and without ORD. The reported numbers are an average over 3 runs of synthesizer training. We used τ=0.3 for all real datasets and τ=0.2 for the 2D toy datasets. The random forest for overlap detection had 50 trees while the other parameters were used at their default values. k=2 was sufficient for the k-fold training.

Visualization on 2D Toy Datasets We provide a better understanding of the working of our method via visualization of ORD on a 2D synthetic toy datasets. Detailed information on the creation of these datasets and minority-majority counts has been provided in the appendix. Dataset Blobs consists of four blobs that belong to the majority D_0 and three smaller blobs belonging to the minority D_1. This was created using Scikit Learn Gaussian Blob generation.

Figure 3 top left corner shows the original majority instances (cyan) and minority instances (yellow), and the subset of majority that our method marked as Overlap points. In each of the subsequent plots, we show data generated by ORD using four different synthetic generators. First observe that recent diffusion models like TabSyn and ForestFlow (last two columns) provide higher quality generations as measured by their visual similarity to the original data. We zoom in further to highlight the difference between baseline generations and ORD generations. We assign to each generated minority point an error label based on the gold $P(y|x)$ model that is known in this toy setting. In the second row, we show the generated minority with wrong generations shown in pink from the baseline model. In the third row, we show the same for ORD variants of the generators. Observe that there are significantly fewer wrong generations from our ORD generators. We will later quantify the exact error of generation.

Overall Accuracy Comparison on Real Datasets In this section, we first report the main accuracy results for all synthesizers and how overall accuracy increases due to ORD in Table 2. From these tables we can make the following important observations. (1) Our proposed ORD with CTabSyn, a conditional latent diffusion model, as a generator, provides

Training Data	Adult		Cardio		Fintech		Heloc	
	Avg of 4	XGBoost	Avg of 4	XGBoost	Avg of 4	XGBoost	Avg of 4	XGBoost
Unbalanced, original	47.74	60.88	32.46	50.00	33.74	66.30	30.90	58.40
Smote	69.34	71.45	61.62	58.77	37.90	56.00	52.27	55.90
Borderline Smote	64.83	68.10	60.72	58.02	37.71	56.00	45.50	54.32
ADASYN	70.79	71.30	61.09	58.15	38.23	55.80	56.52	53.14
TabSyn	76.87	78.28	67.38	69.56	**63.60**	63.21	65.92	65.79
CTabSyn ORD	**80.67**	**81.18**	**70.17**	**73.10**	63.58	**66.46**	**66.36**	**67.77**
CTGAN	65.33	70.31	61.96	63.85	56.14	56.44	**62.93**	**62.86**
CTGAN ORD	**72.81**	**74.52**	**64.90**	**66.33**	**57.58**	**58.38**	61.47	62.34
CTAB-GAN+	77.06	78.09	68.97	57.33	**57.45**	59.40	56.26	54.77
CTAB-GAN+ ORD	**78.90**	**79.72**	**69.74**	**61.46**	54.83	**57.66**	**63.82**	**65.03**
ForestFlow	67.79	69.64	67.42	65.25	61.19	66.22	**65.17**	**66.80**
ForestFlow ORD	**71.20**	**72.88**	**68.94**	**66.03**	**62.91**	**66.58**	64.83	66.22
TabDDPM	69.55	69.76	64.91	64.02	61.88	**60.50**	52.10	60.70
TabDDPM ORD	**77.92**	**78.33**	**66.87**	**68.77**	**63.07**	57.80	**63.22**	**61.30**
TabSyn	76.87	78.28	67.38	69.56	63.60	63.21	65.92	65.79
CTabSyn	79.80	80.01	69.45	71.80	**65.37**	65.95	**66.69**	65.65
CTabSyn ORD	**80.67**	**81.18**	**70.17**	**73.10**	63.58	**66.46**	66.36	**67.77**

Table 2: Comparison of classification accuracy. First part focuses on existing imbalance handling baselines and our best method CTabSyn ORD. The next part shows an increase in accuracy for each individual synthesizer using ORD. Across 26 out of 32 (dataset, generative model) combinations we find that ORD provides a gain in accuracy.

the highest accuracy across all four datasets and all classifiers. (2) Compared to the baseline classifier trained on original data D (first row of numbers), most synthetic data generation methods boost accuracy. The only exception is Fintech where only our methods provide gains. (3) If we replace the data generator from CTabSyn with any of the alternative generators, our ORD extension of training the generator and classifier, provides a boost in accuracy.

A paired T-test was performed on 5 synthesizers and 4 datasets with the balanced test sets, comparing accuracy with and without ORD. A p-value of 6.84e-05 < 0.05 was obtained, **showing the statistical significance of ORD**.

We next investigate the reasons behind the gains that ORD provides over the respective baseline generative model. ORD impacts both the data synthesis step and the classifier training step. In subsequent experiments, we tease apart the contribution of each step in the overall accuracy gains.

ORD Gains Are Due to Better Synthetic Data Generation. We first demonstrate that generators G trained with ORD's ternary labels provide better quality instances. We wanted to check the agreement in class label assigned by generator and true label of generated instances. However, the true label is not available for real data. So, we found two work arounds. First, for D we used a toy dataset where the true class distribution $P(y|x)$ is known. Table 3 shows the results on a mixture of Gaussian dataset as shown in Figure 3. Observe that in all cases minority accuracy is worse than majority, and recent diffusion based models like TabSyn and ForestFlow provide more accurate generations. Our extension ORD improves the minority accuracy even further in all cases, and also provides modest improvement for majority instances. Second, for experiments on real data, we trained an accurate XG Boost model on real balanced data. Note that this data is not available to train the synthesizer, and is only used here to define an oracle. Table 4 shows overall accuracy of generated instances. Across all generators and datasets, we observe that our ORD variant generates instances with greater accuracy.

Model	Avg of Maj, Min (%)	Minority Acc. (%)	Majority Acc.(%)
CTabGan+	67.98	41.67	94.30
CTabGan+ ORD	70.52	46.08	94.95
DDPM	93.51	88.72	98.31
DDPM ORD	94.21	89.87	98.56
TabSyn	91.93	85.75	98.10
CTabSyn ORD	97.72	96.50	98.93
ForestFlow	95.68	91.90	99.47
ForestFlow ORD	95.74	92.02	99.46

Table 3: Accuracy of synthetic data label with true labels coming from the Bayes classifier. Minority data quality improves by the ORD method in most synthesizers. Since the generated data is mostly majority, Overall accuracy as the weighted avg. of majority and minority accuracies.

ORD Trains a Better Classifier by Downsampling Overlapping Majority Note for training the classifier we only sample instances from clear majority and minority. We show that this is another reason why ORD leads to higher classification accuracy. We disantangle the effect of the quality of synthetic instances by training classifiers on real dataset. We train two classifiers: the first is trained on the entire dataset - D_0 and D_1 while second is trained in ORD mode where we remove the instances from D_0 that falls in the overlap re-

Model	**Adult**	**Cardio**	**Fintech**	**Heloc**
CTGAN	67.94	70.89	55.22	54.84
CTGAN ORD	67.24	64.38	56.36	53.97
CTAB-GAN+	54.54	68.08	53.69	52.84
CTAB-GAN+ ORD	77.06	70.48	50.92	49.80
TabDDPM	75.38	59.35	65.83	49.80
TabDDPM ORD	77.36	73.60	61.53	73.82
TabSyn	67.19	69.22	60.73	63.86
CTabSyn ORD	72.64	72.07	65.97	66.91
ForestFlow	69.78	73.49	77.12	74.01
ForestFlow ORD	70.21	73.54	72.42	73.23

Table 4: Accuracy of testing synthetic data on a classifier trained on real **balanced** datasets. An increase with ORD shows how synthesis is improved by ORD.

gion. We show the results in Table 5 and observe that overall accuracy increases with removing Overlap majority points in most cases.

Metric	Accuracy (%)	Minority Accuracy (%)	Majority Accuracy (%)
Adult	76.23	57.06	95.40
Adult ORD	81.57	75.68	87.45
Fintech	69.25	54.27	85.60
Fintech ORD	72.97	76.77	69.16
Heloc	71.04	65.95	76.14
Heloc ORD	70.55	84.56	56.52

Table 5: Effect of removing Overlap i.e. D_{01} on real data is shown in this table. The improvement shows that removing D_{01} helps classifier to learn better decision boundary.

Ablations We perform another interesting ablation to show that ORD directly affects the synthetic data generator and not just the classifier. Here we compare ORD with an ablation where we directly use D to train the generators, sample D_0 and D_1 from them, and then remove overlap instances from D_0. Table 6 presents a comparison of accuracy of the XGBoost classifier trained using the synthetic data in the two settings. We observe that for two of the datasets — Adult and Heloc, the accuracy drops by a huge amount when the filtering of overlapping instances is done after training the generator on binary labels. This is another evidence to support our claim that the finer-grained ternary labels leads to a better quality generator. In Table 2 we did not include the real minority instances to train the classifier to enable fair comparison across all methods. We have found that, out of the various possible augmentation combinations, augmenting the synthetic data with only the minority class D_1 original data points provides the greatest gains. The results are tabulated in Table 7. We present more ablations in the supplementary.

Dataset	ORD Before Synthesis (%)	ORD After Synthesis (%)
Adult	79.19	65.32
Heloc	67.00	44.72
Fintech	60.19	60.90
Cardio	72.04	72.31

Table 6: Comparison of XGBoost Accuracy of ORD done Before and After Synthesis shows that ORD aids Synthesis. It shows that not just removing overlap for the classifier but also providing that information to the synthesizer is necessary.

Training Data	Adult (%)	Fintech (%)	Heloc (%)	Cardio (%)
Real Unbalanced	46.52	25.87	31.04	32.46
Synthetic ORD	78.30	60.52	65.74	63.81
Synthetic ORD + D_1	79.15	63.92	67.40	65.62

Table 7: Effect of augmenting synthetic data with real minority data $D1$. There is an improvement in classification accuracy. The Classifier used is an Average of 4 classifiers LR, MLP, DT and Adaboost.

6 Conclusion, Limitations, Future Work

In this paper, we proposed ORD, a new technique for training classifiers with highly imbalanced class distribution for augmenting with synthetic data from generative models. A key idea of ORD is identifying overlapping majority instances and converting the original binary labels into a ternary labeled dataset. We show that existing deep generative models are also adversely affected by imbalance in training data, and show that class conditional generators trained with our ternary labels provide higher quality data. Finally, ORD under-samples overlapping majority instances and trains the classifier using balanced synthetic minority and clear majority instances along with real minorities. We present a detailed comparison on four real datasets and obtain improvement in classification accuracy in most settings. We explain the reasons behind the gains via three insightful experiments that show that ORD improves both the data synthesis step and the classifier training step.

Limitations and Future Works (1) The method proposed is for datasets having binary class target columns only. However, with some tweaking, there is a possibility of extending this method to datasets with multi-class categorical target columns. (2) While the method has been shown to work on datasets with highly imbalanced categorical columns, it will be interesting to extend to datasets with continuous-valued target columns with a very skewed distribution. (3) The focus of this work is not privacy preservation but an interesting extension is to train the data generator without compromising privacy of the real data.

Acknowledgments

Work done in this project was funded by the State Bank of India (SBI) Data Analytics Hub at IIT Bombay. We thank

SBI data scientists for introducing us to the problem. We acknowledge discussions with Dr Rajbabu in initial phases of the project.

References

Adiputra, I. N. M.; and Wanchai, P. 2023. CTGAN-ENN: A tabular GAN-based Hybrid Sampling Method for Imbalanced and Overlapped Data in Customer Churn Prediction.

Borisov, V.; Sessler, K.; Leemann, T.; Pawelczyk, M.; and Kasneci, G. 2023. Language Models are Realistic Tabular Data Generators. In *The Eleventh International Conference on Learning Representations*.

Breiman, L. 2001. Random forests. *Machine learning*, 45: 5–32.

Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; and Kegelmeyer, W. P. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16: 321–357.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Gulati, M. S.; and Roysdon, P. F. 2023. TabMT: Generating tabular data with masked transformers. arXiv:2312.06089.

Han, H.; Wang, W.-Y.; and Mao, B.-H. 2005. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, 878–887. Springer.

He, H.; Bai, Y.; Garcia, E. A.; and Li, S. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, 1322–1328. Ieee.

Hegselmann, S.; Buendia, A.; Lang, H.; Agrawal, M.; Jiang, X.; and Sontag, D. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, 5549–5581. PMLR.

Johnson, J. M.; and Khoshgoftaar, T. M. 2019. Survey on deep learning with class imbalance. *Journal of big data*, 6(1): 1–54.

Jolicoeur-Martineau, A.; Fatras, K.; and Kachman, T. 2024. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees. In *International Conference on Artificial Intelligence and Statistics*, 1288–1296. PMLR.

Kotelnikov, A.; Baranchuk, D.; Rubachev, I.; and Babenko, A. 2022. TabDDPM: Modelling Tabular Data with Diffusion Models. arXiv:2209.15421.

Leng, Q.; Guo, J.; Tao, J.; Meng, X.; and Wang, C. 2024. OBMI: oversampling borderline minority instances by a two-stage Tomek link-finding procedure for class imbalance problem. *Complex & Intelligent Systems*, 1–18.

Li, Z.; Huang, M.; Liu, G.; and Jiang, C. 2021. A hybrid method with dynamic weighted entropy for handling the problem of class imbalance with overlap in credit card fraud detection. *Expert Systems with Applications*, 175: 114750.

Papamakarios, G.; Pavlakou, T.; and Murray, I. 2017. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30.

Vaswani, A. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Vuttipittayamongkol, P.; and Elyan, E. 2020. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Information Sciences*, 509: 47–70.

Wang, Z.; Wu, C.; Zheng, K.; Niu, X.; and Wang, X. 2019. SMOTETomek-based resampling for personality recognition. *IEEE access*, 7: 129678–129689.

Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; and Veeramachaneni, K. 2019. Modeling Tabular data using Conditional GAN. In *Neural Information Processing Systems*.

Zhang, H.; Zhang, J.; Srinivasan, B.; Shen, Z.; Qin, X.; Faloutsos, C.; Rangwala, H.; and Karypis, G. 2024. Mixed-Type Tabular Data Synthesis with Score-based Diffusion in Latent Space. arXiv:2310.09656.

Zhang, Y.-P.; Zhang, L.-N.; and Wang, Y.-C. 2010. Cluster-based majority under-sampling approaches for class imbalance learning. In *2010 2nd IEEE International Conference on Information and Financial Engineering*, 400–404. IEEE.

Zhao, X.; Guan, S.; Xue, Y.; and Pan, H. 2024. HS-CGK: A Hybrid Sampling Method for Imbalance Data Based on Conditional Tabular Generative Adversarial Network and K-Nearest Neighbor Algorithm. *Computing and Informatics*, 43(1): 213–239.

Zhao, Z.; Birke, R.; and Chen, L. 2023. TabuLa: Harnessing Language Models for Tabular Data Synthesis. arXiv:2310.12746.

Zhao, Z.; Kunar, A.; Birke, R.; and Chen, L. Y. 2022. CTAB-GAN+: Enhancing Tabular Data Synthesis. arXiv:2204.00401.

Zhao, Z.; Kunar, A.; der Scheer, H. V.; Birke, R.; and Chen, L. Y. 2021. CTAB-GAN: Effective Table Data Synthesizing. *CoRR*, abs/2102.08369.

Zhou, Z.-H.; and Liu, X.-Y. 2005. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on knowledge and data engineering*, 18(1): 63–77.