AE-SMOTE: A MULTI-MODAL MINORITY OVERSAMPLING FRAMEWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world binary classification tasks are in many cases unbalanced i.e. the minority class is much smaller than the majority class. This skewness is challenging for machine learning algorithms as they tend to focus on the majority and greatly misclassify the minority. Oversampling the minority using *SMOTE* before training the model is a popular method to address this challenge. Inspired by *SMOTE*, we propose *AE-SMOTE*, which by using an autoencoder, (1) maps the features to a dense continuous latent space, (2) applies oversampling by interpolation in the latent space, and (3) maps the synthetic samples back to the original feature space. While *SMOTE* supports discrete (categorical) features, almost all variants and extensions of *SMOTE* do not. Wrapping any one of these *SMOTE* variants with an autoencoder will enable it to support multi-modal datasets that include discrete features. We have empirically shown the effectiveness of the proposed approach on 35 publicly available datasets.

1 INTRODUCTION

Imbalanced classification tasks arise naturally, for example, consider the problem of credit card fraud detection where the vast majority of transactions are legitimate and only a few are fraudulent. This skewness is challenging for machine learning (ML) algorithms since the algorithms tend to focus on the majority and greatly misclassify the minority. The challenge stems from the ML algorithms optimizing a different metric than the user is interested in, resulting in an undesirable bias in the final trained model. Oversampling the minority class and under-sampling the majority class before training the ML algorithm are popular methods to address this challenge. They are effective because they yield an augmented dataset for which the algorithm's loss function and the user's loss function are more similar. For a formal description, see Section 3.1.

As opposed to random oversampling, *SMOTE* was the first framework to propose balancing the dataset by adding synthetic minority samples Chawla (2002). In *SMOTE*, the synthetic minority samples are created by interpolating pairs of the original minority points, hence instead of working in the original sample space i.e. replicating samples, it generates new samples in the feature space. When the feature space is sparse, the linear interpolation of samples might create unrealistic low probability samples. *SMOTE* addresses this challenge by interpolating pairs of points that are relatively close in the feature space. However, this strategy is inefficient when the feature space is high-dimensional, see Blagus & Lusa (2012).

Considering high-dimension multi-modal data, it is commonly assumed that the data reside on an unknown lower dimension manifold. For such sparse high-dimension data, simple linear interpolation of samples can result in low probability synthetic samples. Motivated by this, we propose Auto Encoder *SMOTE* (*AE-SMOTE*) which is a latent space interpolation scheme based on auto-encoders for oversampling. In summary, our method consists of an unsupervised dimension reduction step where samples are mapped to a dense continuous latent space. Subsequently, samples of interest are interpolated in the learned latent space (using *SMOTE* or any other interpolation technique) before being mapped back to the original feature space. Since *AE-SMOTE* interpolates points in the latent space, it creates more genuine synthetic samples thus improving prediction performance. See Figure 1 for an example of *AE-SMOTE* creating a more realistic synthetic data than *SMOTE* due to the manifold embedding.

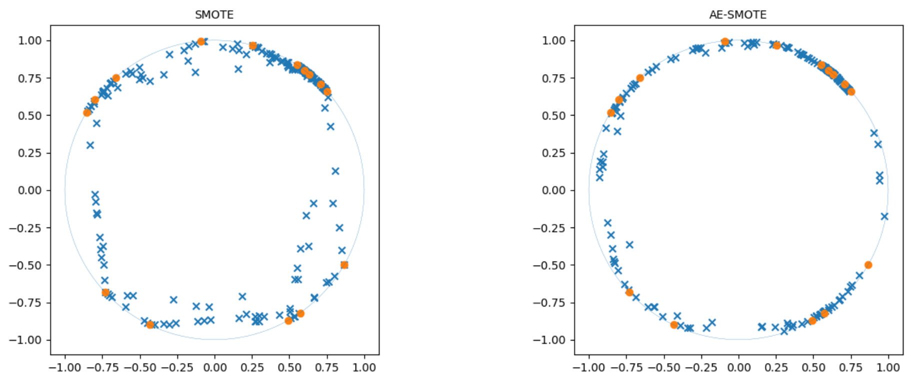

Figure 1: An example of the synthetic minority class samples created using *SMOTE* and *AE-SMOTE*. A $2D$ slice of the $3D$ feature space where the original minority class samples are marked by orange dots and the synthetic by blue $\times$s. The feature space is three dimensional but it is sparse, the samples lie on a cylinder-shaped $2D$ manifold. As a result of this sparsity, *SMOTE* creates some low probability sample toward the center of the cylinder. *AE-SMOTE* on the other hand interpolates samples in the dense latent space resulting in more realistic samples, i.e. closer to the underlying data manifold.

Many real-life tabular datasets are multi-modal and include not only continuous numeric features but also categorical features e.g. gender, color, marital status, etc. We will denote the former continuous and the latter discrete. It is common to assume that the classification of each feature (continuous or discrete) is known. Modern tabular datasets might include additional data types such as free text fields, images, and even audio. These are out of the scope of this work. When trying to apply *SMOTE* to a multi-modal dataset with discrete features we face two challenges: (1) how to calculate distances between samples? and (2) how to set the discrete feature values for the synthetic samples? e.g. how to interpolate ”dog” and ”cat”? The original *SMOTE* paper introduced *SMOTE-NC* which is a *SMOTE* variant that supports discrete features by: (1) using a heuristic to estimate the distance implied by the discrete features and (2) the discrete feature values are set to be the majority of the k nearest neighbors.

To the best of our knowledge, *SMOTE-NC* is the only *SMOTE* variant that supports discrete features explicitly. A common method to apply any interpolation method to discrete features is to encode them ordinally and consider them to be continuous. This method results in synthetic samples that are continuous rather than discrete, thus not realistic. Moreover, many algorithms are optimized for handling discrete features (e.g. Catboost and MLPs with embedding) and such augmentation will render these optimizations useless. Our approach shifts the challenge of the discrete features from the interpolation method to the encoder-decoder, where we can leverage previous research. Additionally, more than 100 extensions and variants of *SMOTE* were proposed. However, to the best of our knowledge, none of them support discrete features. By mapping the discrete and continuous features to a unified continuous latent space, we enable all these algorithms to produce multi-modal data. An overview of our method is shown in Figure 2.

To solve the problem presented here, it is required to create synthetic examples of structured data. We explored several methods proposed in the literature, and conclude that our autoencoder based approach is by far superior to the available alternatives. Unlike naive interpolation-based methods, it is sophisticated enough to avoid unrealistic examples. On the other hand, unlike GAN based approaches, it is simple enough to run at scale and avoid system failures related to an overly complex system such as overfitting and mode-collapse. We studied several autoencoder variants and empirically concluded that the simple vanilla autoencoder provides the best results.

To summarize, the benefits of our approach and our contributions are:[1]

[1]Our code can be found at https://github.com/<anon-user>/<anon-repo-name>

1. Due to the dense embedding and the ability to generate discrete features, *AE-SMOTE* generates more realistic synthetic samples compared to other approaches resulting in better prediction quality

2. More than 100 extensions and variants of *SMOTE* were proposed. However, almost all of them support only continuous features. By mapping the discrete and continuous features to a unified continuous latent space, we enable these algorithms to produce multi-modal data. We demonstrate this by wrapping *polynom-fit*, a top performing *SMOTE* variant, with our encoder decoder, enabling it to produce multi-modal synthetic samples. In our experiments *AE-Poly* yielded better prediction quality than the original *polynom-fit*.

2 Related Work

Since *SMOTE*'s inception, more than 100 extensions and variations have been published. However, to the best of our knowledge only *SMOTE-NC* proposed in the original paper supports discrete features. In fact, the two recent survey papers do not even mention discrete features or multi-modal data, see Fernández et al. (2018) and Kovács (2019). Kovács (2019) empirically compared 85 variants of *SMOTE* on 104 imbalanced datasets including both continuous and discrete features. The paper does not describe how the discrete features were processed. We believe that they were simply treated as continuous after using an ordinal encoder. The best method was found to be *polynom-fit* which, similarly to *SMOTE*, interpolates points in the feature space, see Gazzah & Amara (2008). However, differently from *SMOTE*, *polynom-fit* allows interpolating of minority points that are not very close to each other. The second best performing algorithm, *ProWSyn* by Barua et al. (2013), also allowed interpolating of far apart minority samples. We will not survey all *SMOTE* variants but only the methods that resemble *SMOTE*.

Several variants of *SMOTE* share the idea of mapping the samples to another space which has some desired features, using *SMOTE* to create synthetic samples in the new space and map the synthetic samples back to the original feature space. Wang et al. (2006) proposed to map the samples using local linear embedding aiming to create a lower-dimensional space where the data is more separable. Gu et al. (2009) similarly proposed to use isometric feature mapping (Isomap). Kernel functions were also used to map the features, see Pérez-Ortiz et al. (2016) and Tang & He (2015). When the classifier is SVM, oversampling could be done directly in the kernel space, see Mathew et al. (2015).

As they are very natural, auto-encoders were previously proposed as a means to provide the bi-directional mapping. Bellinger et al. (2015) proposed to create synthetic samples by adding Gaussian noise to the original samples in the latent space and then decode them back to the feature space. Later, the same authors proposed to apply *SMOTE* in the latent space, see Bellinger et al. (2016). However, there are two key differences between their approach and ours: (1) they train the auto-encoder on the minority class samples only which, due to the low number of samples, force them to train shallow models and (2) they did not consider discrete features. Babaei et al. (2019) proposed to encode the samples and train the classifier in the latent space. To improve unsupervised anomaly detection, Lim et al. (2018) proposed to use adversarial auto encoders to encode the features into a Gaussian mixture latent space and augment the dataset in that latent space.

Some methods incorporate mapping the feature to other spaces but use the new space differently. MOT2LD (Xie et al. (2015)) first maps each training sample into a low-dimensional space and then applies clustering and weighting heuristics in the low-dimensional space. In ADOMS (Tang & Chen (2008)) each sample neighbor are derived in the original feature space, however, the synthetic sample is then created along with the first principal component of the k neighbors.

Generation of tabular multi-modal data was studied in the context of GANs (generative adversarial networks). The challenge of synthesizing the discrete features was addressed using three methods: (1) noising the discrete data (Xu & Veeramachaneni (2018)), (2) using a Gumbel softmax (Xu et al. (2019), Park et al. (2018)) or (3) using an autoencoder to map the data to a continuous latent space and train the GAN in that space (Choi et al. (2017)). FAST-DAD (Fakoor et al. (2020)) generates multi-modal synthetic samples by augmenting existing samples using Gibbs sampling. The Gibbs based augmentation method requires a pre-trained conditional expectation model for all features and another model is used to label the new samples.

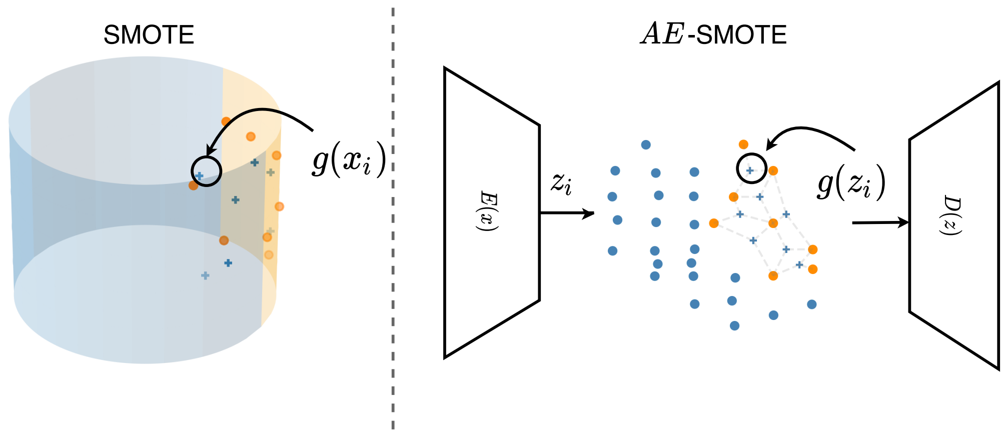

Figure 2: Overview of our proposed method. Left diagram depicts SMOTE in the sample space, where blue o are majority samples, orange o are minority samples present in the dataset and blue + are synthetic minority examples generated. Sample points x are fed to an encoding blocked h_ϕ to produce latent points z. Subsequently, samples are manipulated by the function $g(z_i)$ prior to decoding to the original sample space.

3 Method

3.1 Oversampling

Consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where x_i are observations sampled from a data-generating distribution $p(x)$ and $y_i \in \{0, 1\}$. We consider a multi-modal setting where x_i is a concatenation of discrete $\mathcal{D} = [D_1, \cdots, D_{|\mathcal{D}|}]$ and continuous features $\mathcal{C} = [C_1, \cdots, C_{|\mathcal{C}|}]$. The classifier $f : \mathcal{X} \to \mathcal{Y} \in \mathcal{F}$ is a function learned by an ML algorithm which aims at optimizing f for a given loss function $l_A(\cdot)$

$$f = \min_{f \in \mathcal{F}} \sum_{i=1}^{N} l_A(f(x_i), y_i) \tag{1}$$

Where we assumed that $l_A(\cdot)$ is additive, i.e. the loss of the dataset is the sum of sample losses. To the best of our knowledge, all mainstream ML algorithms assume additivity of the loss.

On the other hand, in imbalanced classification we are usually interested in optimizing ROC-AUC, F_1-score or G-score which are not additive, hence the ML algorithm cannot directly optimize them. We denote this second loss function by $l(\cdot)$. Oversampling schemes address this challenge by adding minority class synthetic samples $D_{aug} = \{(x_i, y_i)\}_{i=1}^M$ to the dataset. The idea is that l_A applied to $D \cup D_{aug}$ will be more similar to l applied to D, i.e.

$$\left| l(D) - \sum_{D \cup D_{aug}} l_A \right| < \left| l(D) - \sum_{D} l_A \right| \tag{2}$$

3.2 Mapping to the Latent Space

To map samples to latent space, we adopt the standard autoencoder scheme proposed in Hinton & Salakhutdinov (2006). As we are concerned with multi-modal setting, for every categorical column D, we introduce an embedding layer $\mathcal{W} \in \Re^{|D| \times d_{|D|}}$. The input which is a concatenation of discrete and continuous features is passed through the embedding layers resulting in a feature vector $x_i \in \Re^{|\mathcal{C}| + \sum_i^{|D|} d_{|D_i|}}$ that is used as the input to the autoencoder

$$z_i = h_\phi(x_i) \tag{3}$$

Where $h_\theta(\cdot)$ is a high capacity deep neural network, such as a set of fully connected layers.

As in traditional autoencoders, a decoder module $h_\theta(z)$ is used to map the latent points back to the feature space x. The objective minimized while training the AE is the reconstruction loss:

$$\arg\min E_{p(x)}[d(h_\theta(z), x)] \tag{4}$$

In a multi-modal setting, the reconstruction contains both discrete and continuous variables therefore the function $d(\cdot)$ is a sum of the softmax loss and mean squared error (MSE):

$$J_{recon}(D;\theta,\phi) = \sum_i^N \sum_c^{|C|} ||h_\theta(z_i)^c - x_i^c||_2^2 + \sum_i^N \sum_d^{|D|} \mathbf{1}[x_i^D = o] \log(h_\theta(z_i)^o) \tag{5}$$

Once we have a fully trained autoencoder model we can leverage prior work (e.g. *SMOTE*, *polynom-fit*) on interpolating minority samples in the latent space and using the deocder to generate synthetic data in the original sample space.

3.3 *AE-SMOTE* AND *E-Poly*

As previously mentioned, in order to generate new synthetic samples the set of minority samples $D_{\text{minority train}} = \{(x_i, y_i)|y_i = 1\}$ are first mapped into the latent space $\mathcal{Z}_{minority}$. Subsequently, based on our scheme, we propose to use two minority oversampling techniques: *AE-SMOTE* that interpolates in the latent space using *SMOTE* and *AE-Poly* that interpolates using *polynom-fit*, Gazzah & Amara (2008). Briefly, *SMOTE* and its variants use a distance metric to find nearest neighbors z_{NN}^i to a minority sample $z_i \in \mathcal{Z}_{minority}$ which are then averaged using a uniform noise vector $u \sim U(0, 1)$

$$z_{synth} = z_i + (1 - u) \odot z_{NN}^i \tag{6}$$

On the other hand, *polynom-fit* was selected based on it's performance in the experiments of Kovács (2019). Additionally, *polynom-fit* allows interpolation of samples that are far apart in the feature space which might result in low probability synthetic samples when the feature space is sparse. This concern is alleviated when the interpolation is carried out in the dense latent space. Recall that *polynom-fit* does not support multi-modal data, thus, we enable it to support discrete features by wrapping it with our autoencoder. We refer the reader to the respective papers for more details on these methods.

4 EXPERIMENTAL SETUP

In this section we carry out experiments to demonstrate the effectiveness of the proposed method in addressing imbalanced binary classification challenges.

Data We evaluated the method on 35 public datasets with a varying number of samples, dimensions, and the ratio of continuous/discrete columns. Out of the 22 datasets were chosen by starting with all the 104 datasets of Kovács (2019) and filtering out all datasets with less than 1000 rows. This is done as the datasets we are concerned should have an adequate number of samples to train an autoencoder and as the training is split into two steps, i.e. an unsupervised learning step and subsequently downstream performance task the train/val/test splits should contain enough samples to represent the original dataset. Additionally, we added 13 challenging datasets from the imbalanced dataset benchmark of Lemaitre et al. (2016). For more details on the datasets used refer to Appendix B. For each dataset, stratified k-fold train/test splits[2] with k set to 7, where 80% is sampled as train and 20% as a test. For datasets where categorical columns exist we encoded them ad ordinal integers. We preprocess continuous features by normalizing using uniform quantile transformation [3]. For the baselines, we try feeding values using our preprocessing steps or the raw values as certain methods internally preprocess values, and the better approach is chosen for each baseline.

Methods As previously noted, there are very few methods that allow the generation of multi-modal tabular data including both continuous and discrete features. We empirically compared *AE-SMOTE* and *AE-Poly* to all available methods:

[2]Using the sklearn.model_selection.StratifiedKFold

[3]Using the sklearn.preprocessing.QuantileTransformer

- *SMOTE* and *polynom-fit*: interpolation methods that support only continuous features
- *SMOTE-NC*: a multi-modal *SMOTE* variant.
- CatSW: CatBoost model trained on data using sample weights as supported by catboost.
- CTGAN Park et al. (2018): a recent generative GAN model specifically designed to handle tabular datasets by conditioning on discrete columns. The open source implementation was used [4].
- DOPING Lim et al. (2018): an adversarial auto-encoder with a gaussian prior and with the same capacity as our autoecnoder is trained on both minority and majority samples. Subsequently new samples are generated by mapping to the learned latent space, filtering minority samples based on magnitude of latent vectors and applying *SMOTE*.
- TGAN Xu & Veeramachaneni (2018): a precursor to CTGAN where a LSTM network is used with different heads for each column, where each step corresponds to a column feature value. This model is trained using the traditional GAN framework.
- TGAN-SkipBauke Brenninkmeijer (2019): Slightly modified TGAN architecture with a skip connection between the input and the generator output to help with gradient flow.
- TGAN-WGAN Bauke Brenninkmeijer (2019): the TGAN network trained using wassertein gan loss.

Metrics We use popular metrics for imbalanced data as done in Kovács (2019): F_1-score, G-score, ROC-AUC (area under the receiver operating characteristic curve) as well as PR-AUC (precision recall area under the curve)

Evaluation To evaluate the synthetic generation methods we augment the training set using the aforementioned methods and use Catboost (Prokhorenkova et al., 2018) as the classifier due to its popularity and strong performance on multi-modal tabular data. The classifier is trained on the same subsets of training/validation sets as the unsupervised training step. For each oversampler, we considered at most 35 hyperparameters combinations. Note that the number (or ratio) of synthetic samples is a hyperparameter of all oversamplers, for which we run experiments for balance ratio $R = \frac{\#\text{minority}}{\#\text{majority}} \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$. 5 sets of hyper-parameters were considered for Catboost. Classification performance was evaluated by stratified 7-fold cross-validation. For each oversampler and each dataset, the best results for the overall combinations of oversampler parameters and classifier parameters are considered as the results of the oversampler for the dataset. For oversamplers that generate valid synthetic discrete features, the indices of the discrete features are passed to CatBoost.

TrainingWe summarize the training protocol. We compose both our encoder h_ϕ and h_θ as fully connected layers FC-BN-ReLU. We try training a 1024-1024 FC for both encoder and decoder, or a 512-512 FC network. When training these networks all samples in the training set are used (minority & and majority samples). We set the latent space dimension d to one of three different values $\{d^{\frac{4}{5}}, d^{\frac{2}{3}}, d^{\frac{4}{7}}\}$. For all experiments for training to our proposed method we use Adam (Kingma & Ba, 2014) optimizer with an initial learning rate of $1e-3$ decayed every 10 epochs by a factor of 0.1 and trained for a maximum of 40 epochs with early stopping on the reconstruction loss of the validation set. Note that the validation set is obtained by splitting the training set indices into 85% train and 15% validation. For most datasets, it suffices to train for about 20 epochs. Note that for datasets that contain categorical columns, the embedding size for these columns is set to $min(600, round(1.6 * |D|^{0.56})$ where $|D|$ is the number of unique values for the categorical column. For the model based synthetic generation methods, we make sure there is no data leakage across the unsupervised training step and classifier training step, this is done by using the pre-generated stratified training and test splits as explained previously. The random seed was fixed across all runs. We train all models on a machine with 8 Intel-Xeon Skylake CPUs and 30GB memory.

5 Discussion: The Bare Minimum

Our proposed solution thus far has been amenable to a simple approach. We will further discuss attempts at improving the proposed method through feature selection and latent space regularization.

[4]https://github.com/sdv-dev/CTGAN

As tabular datasets may be high dimensional and sparse, instead of simply composing the encoder block as a set of fully connected layers, we pass the features through a set of S feature selector blocks $S_j(x_i)$ where each block computes sparse attention on the subset of features $x \in \Re^d$. The learnable feature selector block computes outputs using a matrix $M \in \Re^{k \times d}$

$$s_k(x) = \sum_{i}^{d} x_j \cdot F(M_{kj}) \tag{7}$$

where we desire to select k features. In this network, the input layer is a set of such feature selector blocks $S = (s^1, \cdots s^{|S|})$ which are subsequently fed to an encoder h_ϕ where points are mapped to the latent space representation z. Intuitively, by composing the input with these blocks it helps to reduce overfitting issues as prevalent in simple fully connected networks. We propose to use the Gumbel-softmax (Jang et al., 2016; Maddison et al., 2016) as a differentiable selector function at the input layer. During training each row computes a linear combination of features and gradually through temperature annealing, the network converges to a subset of k features for each block.

Additionally, as our objective is to interpolate in the latent space, it is desirable to obtain a latent space that would allow safe interpolation of samples. Recent developments have shown that adding a regularization loss in the latent space results in a deterministic auto-encoder with similar properties as in VAEs (Ghosh et al., 2019), all be it much faster and easier to train. Motivated by this, we add a co-linearity loss: given two samples that should interpolate in the sample space, we define a transition loss in the latent space as

$$T(x_i, x_j) = \text{Norm}(1 - z_i - z_j) \tag{8}$$

where x_i, x_j are two examples in the sample space and z_i and z_j are their corresponding latent mappings via the encoding block. This regularization is added to the objective function and we train the auto-encoder with this new term:

$$J_{AE}(D; \theta, \phi) = J_{recon} + \eta T \tag{9}$$

where $\eta \in \Re$ is the regularization weight.

We have done an ablation study of these proposed changes in Table 1[5]. Note, from the results the vanilla autoencoder marginally benefits from the regularization term whereas the more complicated autoencoder with selector blocks shows more benefits from using this interpolation loss term. As such, the simpler model with fewer hyperparameters is chosen for our final comparison in the results section.

Table 1: Ablation study on additional changes for improving the simple AE. The values represent the average ranking across all datasets for each metric. SAE denotes the encoder with selector blocks. Rows with added regularization have $+T$ appended to the names.

	Average Rank				
Method	**F1-score**	**G-score**	**PR-AUC**	**ROC-AUC**	**all**
AE-*Poly* $+T$	4.8	5.3	5.4	5.8	5.3
AE-*Poly*	5.3	5.2	5.3	5.2	5.3
AE-*SMOTE*	5.2	5.6	5.1	5.6	5.4
AE-*SMOTE* $+T$	5.6	5.6	5.6	5.6	5.6
SAE-poly $+T$	5.4	5.8	5.4	5.9	5.6
SAE-smote $+T$	7.0	3.8	5.8	6.4	5.7
SAE-smote	7.0	4.4	5.9	6.6	6.0
SAE-poly	6.7	5.6	6.1	5.6	6.0
polynom-fit-SMOTE	4.6	8.4	7.0	6.0	6.5
SMOTE	5.6	7.9	7.5	6.6	6.9

[5]For more details on the experiments and hyperparameter settings refer to section A

6 Results

As previously mentioned, to evaluate our method we have chosen a wide range of datasets with varying feature sizes and discrete/continuous ratios. Following this extensive evaluation, the results of all oversampling methods on all datasets are presented in Appendix B, we leave the results blank for certain datasets where oversamplers failed to produce synthetic samples. The aggregated results are displayed in Table 2. The results were aggregated over all datasets by averaging the methods' rank, i.e. for each dataset and metric the methods were sorted by performance where the rank of the best method is 1. Then, the rank of each method was averaged over all datasets. The final column, "all" column, was produced by averaging the oversamplers ranks over all metrics and datasets.

From Table 2 it is clear that both *AE-Poly* and *AE-SMOTE* outperform all other methods. Specifically, both *SMOTE* and *polynom-fit* yield better prediction quality when they are wrapped with an autoencoder, compare *SMOTE* and *polynom-fit* to *AE-SMOTE* and *AE-Poly* respectively.

Although *polynom-fit* has demonstrated better empirical performance in the experiments of Kovács (2019), in our setting, the traditional *SMOTE* was superior. This discrepancy might result from the filtering of very small datasets (less than 1000 rows). Interestingly, even though *SMOTE-NC* handles categorical columns explicitly by applying heuristics, *SMOTE*s simple approach of treating the categorical features as numeric provides better results.

The advanced GAN based methods recently proposed (including CTGAN which is a recent state of the art method for tabular data generation) performed poorly compared to the simple interpolation methods. This could largely be due to mode collapse, and the difficulty of learning a generative model in a limited data setting. Moreover, due to the complexity of these methods, training them requires considerably more hardware and is more expensive compared to our simpler autoencoder method. In particular TGAN is the slowest and most resource hungry amongst all tested methods, due to the large set of parameters contained within each head representing different columns.

Table 2: Comparison of minority oversampling methods

	Average Rank				
	F1-score	**G-score**	**PR-AUC**	**ROC-AUC**	**all**
Method					
AE-Poly	4.0	2.3	3.7	3.9	3.5
AE-SMOTE	3.8	2.7	3.5	4.1	3.5
polynom-fit	4.2	5.8	5.6	5.0	5.2
SMOTE	4.0	6.5	5.4	4.9	5.2
CTGAN	6.3	4.9	5.8	6.5	5.9
DOPING	7.7	5.3	5.7	5.8	6.1
SMOTE-NC	5.1	7.4	6.1	5.9	6.1
CatSW	7.2	9.3	7.9	6.9	7.8
TGAN	8.6	7.5	8.5	8.8	8.4
TGAN-WGAN	8.4	7.7	8.9	8.8	8.4
TGAN-Skip	8.8	8.9	8.9	9.1	8.9

7 Conclusion

There are very few methods that allow creation of synthetic multi-modal tabular data including both continuous and discrete features. We addressed the multi-modal data challenge by encoding the data in a dense continuous latent space, interpolate there and map the samples back to the original feature space. Thus, we shift the multi-modal data challenge from the interpolation method to the autoencoder. As an example, we introduced *AE-SMOTE* and *AE-Poly* which generated better synthetic data thus provided improved prediction performance on a variety of multi-modal datasets. Our framework, of wrapping an interpolation oversampler with an autoencoder, can be applied to any interpolation oversampler thus enabling it to produce high-quality multi-modal synthetic data.

References

J. Alcala-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. Garcia, L. Sanchez, and F. Herrera. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2-3):255–287, 2011.

Kasra Babaei, Zhi-Yuan Chen, and Tomás Maul. Data augmentation by autoencoders for unsupervised anomaly detection. *ArXiv*, abs/1912.13384, 2019.

Sukarna Barua, Md. Monirul Islam, and Kazuyuki Murase. Prowsyn: Proximity weighted synthetic oversampling technique for imbalanced data set learning. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 317–328, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37456-2.

Arjen P. de Vries Bauke Brenninkmeijer, Youri Hille. On the generation and evaluation of synthetic tabular data using gans. 2019.

C. Bellinger, N. Japkowicz, and C. Drummond. Synthetic oversampling for advanced radioactive threat detection. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 948–953, 2015.

Colin Bellinger, Chris Drummond, and Nathalie Japkowicz. Beyond the boundaries of smote - a framework for manifold-based synthetically oversampling. In *ECML/PKDD*, 2016.

R. Blagus and L. Lusa. Evaluation of smote for high-dimensional class-imbalanced microarray data. In *2012 11th International Conference on Machine Learning and Applications*, volume 2, pp. 89–94, 2012.

N. V. Chawla. Smote : Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002. URL https://ci.nii.ac.jp/naid/20001554764/en/.

Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. Generating multi-label discrete patient records using generative adversarial networks. volume 68 of *Proceedings of Machine Learning Research*, pp. 286–305, Boston, Massachusetts, 18–19 Aug 2017. PMLR. URL http://proceedings.mlr.press/v68/choi17a.html.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Rasool Fakoor, Jonas Mueller, Nick Erickson, Pratik Chaudhari, and Alexander J. Smola. Fast, accurate, and simple models for tabular data via augmented distillation, 2020.

Alberto Fernández, Salvador García, Francisco Herrera, and Nitesh V. Chawla. Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *J. Artif. Int. Res.*, 61(1):863–905, January 2018. ISSN 1076-9757.

S. Gazzah and N. E. B. Amara. New oversampling approaches based on polynomial fitting for imbalanced data sets. In *2008 The Eighth IAPR International Workshop on Document Analysis Systems*, pp. 677–684, 2008.

Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael J. Black, and Bernhard Schölkopf. From variational to deterministic autoencoders. *CoRR*, abs/1903.12436, 2019. URL http://arxiv.org/abs/1903.12436.

Qiong Gu, Zhihua Cai, and Li Zhu. Classification of Imbalanced Data Sets by Using the Hybrid Re-sampling Algorithm Based on Isomap. In Zhihua Cai, Zhenhua Li, Zhuo Kang, and Yong Liu (eds.), *Advances in Computation and Intelligence*, pp. 287–296, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04843-2.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

György Kovács. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, 83:105662, 2019. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2019.105662. URL http://www.sciencedirect.com/science/article/pii/S1568494619304429.

Guillaume Lemaitre, Fernando Nogueira, Christos K. Aridas, and Dayvid V. R. Oliveira. Imbalanced dataset for benchmarking. September 2016. doi: 10.5281/zenodo.61452. URL https://doi.org/10.5281/zenodo.61452. These data are originally from other repository. See the references to know which licensing applied to them.

S. K. Lim, Y. Loo, N. Tran, N. Cheung, G. Roig, and Y. Elovici. Doping: Generative data augmentation for unsupervised anomaly detection with gan. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 1122–1127, 2018.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables, 2016.

J. Mathew, M. Luo, C. K. Pang, and H. L. Chan. Kernel-based smote for svm classification of imbalanced datasets. In *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, pp. 001127–001132, 2015.

Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10): 1071–1083, June 2018. ISSN 2150-8097. doi: 10.14778/3231751.3231757. URL https://doi.org/10.14778/3231751.3231757.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in neural information processing systems*, pp. 6638–6648, 2018.

M. Pérez-Ortiz, P. A. Gutiérrez, P. Tino, and C. Hervás-Martínez. Oversampling the minority class in the feature space. *IEEE Transactions on Neural Networks and Learning Systems*, 27(9):1947–1961, 2016.

B. Tang and H. He. Kerneladasyn: Kernel based adaptive synthetic data generation for imbalanced learning. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pp. 664–671, 2015.

S. Tang and S. Chen. The generation mechanism of synthetic minority class examples. In *2008 International Conference on Information Technology and Applications in Biomedicine*, pp. 444–447, 2008.

J. Wang, M. Xu, H. Wang, and J. Zhang. Classification of imbalanced data by using the smote algorithm and locally linear embedding. In *2006 8th international Conference on Signal Processing*, volume 3, 2006.

Zhipeng Xie, Liyang Jiang, Tengju Ye, and Xiaoli Li. A synthetic minority oversampling method based on local densities in low-dimensional space for imbalanced learning. In Matthias Renz, Cyrus Shahabi, Xiaofang Zhou, and Muhammad Aamir Cheema (eds.), *Database Systems for Advanced Applications*, pp. 3–18, Cham, 2015. Springer International Publishing. ISBN 978-3-319-18123-3.

Lei Xu and Kalyan Veeramachaneni. Synthesizing tabular data using generative adversarial networks, 2018.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 7335–7345. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8953-modeling-tabular-data-using-conditional-gan.pdf.

A Ablation

Our ablation study is conducted over all datasets, using 7-fold stratified splits. The summary of results are shown in Table A. Although all proposed changes lead to improved results over the widely used *SMOTE* and *polynom-fit-SMOTE* the simpler approach is used in our final results table in appendix B.

For this experiment, the autoencoder with selector blocks contains two additional hyperparameters in comparison to the vanilla autoencoder, $|S|$ the number of selector blocks, and the number of features k each block selects. In our ablation study we set these parameters to $2 * \frac{d_{\text{input}}}{d_{\text{latent}}}$ and $\frac{d_{\text{latent}}}{2}$ respectively, where d_{latent} is the latent dimension, and d_{input} is the dimension size of the tabular input post embedding and as detailed n the experiment section 3.3 the same set of latent dimensions are experimented with as with the autoencoder. For rows with regularization, we simply set proposed regularization weight to 10^{-3} as this generally showed to be a reasonable value for all datasets. The same training procedure as detailed in section 3.3 is used for all methods.

B Results

Table 3: Comparison of minority oversampling methods.

		Metric			
Dataset	**Method**	**F1 - score**	**G - score**	**PR - AUC**	**ROC - AUC**
	DOPING	71.53 (±0.58)	**84.03** (±0.63)	83.16 (±0.48)	92.96 (±0.25)
	SMOTE	72.08 (±0.66)	83.35 (±0.43)	82.70 (±0.33)	92.74 (±0.16)
	CatSW	71.32 (±0.64)	75.93 (±0.68)	82.64 (±0.55)	92.80 (±0.24)
	polynom-fit-SMOTE	71.41 (±0.66)	83.64 (±0.55)	82.91 (±0.44)	92.83 (±0.23)
	SMOTE-NC	**72.39** (±0.69)	83.02 (±0.90)	82.82 (±0.64)	92.81 (±0.31)
adult	CTGAN	71.54 (±0.51)	84.01 (±0.67)	83.08 (±0.43)	92.91 (±0.22)
	AE-*SMOTE*	71.53 (±0.51)	83.99 (±0.64)	83.12 (±0.52)	92.95 (±0.26)
	AE-*Poly*	71.64 (±0.49)	84.01 (±0.67)	**83.17** (±0.46)	**92.97** (±0.23)
	TGAN-Skip	71.02 (±0.64)	82.65 (±0.93)	81.98 (±0.73)	92.34 (±0.31)
	TGAN-WGAN	71.37 (±0.53)	83.85 (±0.63)	83.00 (±0.43)	92.88 (±0.24)
	TGAN	71.27 (±0.61)	83.84 (±0.64)	82.98 (±0.48)	92.86 (±0.27)
	DOPING	**86.48** (±2.62)	97.31 (±0.91)	84.93 (±3.15)	97.68 (±0.90)
	SMOTE	84.97 (±2.11)	92.85 (±0.93)	85.00 (±3.75)	97.36 (±1.65)
	CatSW	27.35 (±11.86)	39.92 (±10.17)	75.83 (±5.88)	95.81 (±2.95)
	polynom-fit-SMOTE	85.14 (±2.07)	92.97 (±1.35)	84.89 (±2.92)	**98.04** (±0.93)
	SMOTE-NC	84.97 (±2.11)	92.85 (±0.93)	85.00 (±3.75)	97.36 (±1.65)
fraud	CTGAN	84.48 (±2.05)	94.49 (±1.77)	81.31 (±3.47)	96.22 (±2.01)
	AE-*SMOTE*	85.25 (±2.25)	96.23 (±1.79)	83.34 (±3.52)	97.31 (±1.22)
	AE-*Poly*	86.00 (±2.49)	**97.48** (±0.80)	84.76 (±2.55)	97.87 (±0.98)
	TGAN-Skip	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN-WGAN	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN	- (±−)	- (±−)	- (±−)	- (±−)
	DOPING	84.55 (±0.69)	95.37 (±0.33)	92.95 (±0.38)	99.75 (±0.03)
	SMOTE	85.67 (±0.60)	94.45 (±0.34)	93.12 (±0.22)	99.78 (±0.02)
	CatSW	74.49 (±3.90)	77.57 (±3.29)	91.79 (±1.35)	**99.79** (±0.03)
	polynom-fit-SMOTE	83.70 (±0.49)	94.95 (±0.30)	92.23 (±0.27)	99.73 (±0.03)
	SMOTE-NC	**85.92** (±0.31)	94.73 (±0.31)	**93.23** (±0.28)	99.77 (±0.02)
covtype	CTGAN	83.44 (±0.48)	**95.48** (±0.30)	92.17 (±0.37)	99.69 (±0.05)
	AE-*SMOTE*	84.86 (±0.69)	95.37 (±0.33)	93.04 (±0.37)	99.76 (±0.03)
	AE-*Poly*	83.95 (±0.53)	95.46 (±0.30)	92.52 (±0.35)	99.71 (±0.03)
	TGAN-Skip	83.38 (±0.50)	95.37 (±0.32)	92.14 (±0.20)	99.69 (±0.04)
	TGAN-WGAN	84.28 (±1.31)	95.41 (±0.18)	92.74 (±0.75)	99.72 (±0.06)
	TGAN	84.27 (±1.04)	95.35 (±0.35)	92.70 (±0.67)	99.73 (±0.05)
	DOPING	95.79 (±1.44)	99.35 (±0.48)	99.18 (±0.57)	99.95 (±0.04)
	SMOTE	96.11 (±1.48)	99.16 (±0.53)	99.20 (±0.58)	99.95 (±0.05)
	CatSW	92.57 (±7.93)	93.82 (±6.99)	98.66 (±2.33)	99.91 (±0.16)
	polynom-fit-SMOTE	**96.72** (±1.11)	99.28 (±0.29)	99.36 (±0.52)	99.97 (±0.03)
	SMOTE-NC	96.11 (±1.48)	99.16 (±0.53)	99.20 (±0.58)	99.95 (±0.05)
letters	CTGAN	96.14 (±1.46)	99.54 (±0.33)	99.21 (±0.76)	99.96 (±0.05)
	AE-*SMOTE*	96.25 (±1.29)	99.50 (±0.37)	99.33 (±0.37)	99.96 (±0.02)
	AE-*Poly*	96.30 (±1.03)	**99.61** (±0.26)	**99.42** (±0.58)	**99.97** (±0.03)
	TGAN-Skip	96.24 (±1.44)	99.37 (±0.30)	99.28 (±0.65)	99.96 (±0.03)
	TGAN-WGAN	95.96 (±2.22)	99.45 (±0.45)	98.84 (±1.26)	99.90 (±0.13)
	TGAN	95.76 (±1.30)	99.49 (±0.29)	98.98 (±0.63)	99.92 (±0.08)
	DOPING	86.72 (±1.61)	98.27 (±0.73)	90.80 (±1.37)	99.36 (±0.22)
	SMOTE	87.43 (±1.85)	95.49 (±1.86)	90.74 (±0.92)	99.29 (±0.19)
	CatSW	63.16 (±11.25)	69.85 (±9.84)	88.48 (±1.95)	99.20 (±0.27)
	polynom-fit-SMOTE	86.74 (±1.32)	97.27 (±0.62)	90.50 (±1.96)	99.32 (±0.21)
	SMOTE-NC	87.43 (±1.85)	95.49 (±1.86)	90.74 (±0.92)	99.29 (±0.19)
protein homo	CTGAN	86.80 (±1.51)	98.07 (±0.50)	89.78 (±1.88)	99.20 (±0.30)
	AE-*SMOTE*	86.51 (±1.36)	98.24 (±0.54)	88.89 (±1.70)	99.14 (±0.22)
	AE-*Poly*	86.73 (±1.40)	**98.38** (±0.44)	89.66 (±2.05)	99.19 (±0.28)
	TGAN-Skip	86.44 (±1.11)	97.94 (±0.49)	90.22 (±1.74)	99.34 (±0.20)
	TGAN-WGAN	86.84 (±1.62)	98.33 (±0.64)	**90.88** (±1.40)	**99.45** (±0.15)
	TGAN	86.59 (±1.62)	98.23 (±0.36)	90.63 (±1.69)	99.39 (±0.17)
	DOPING	0.10 (±0.08)	64.52 (±31.53)	6.17 (±0.47)	62.07 (±2.61)
	SMOTE	0.09 (±0.07)	70.12 (±47.90)	8.45 (±5.29)	62.97 (±0.80)
	CatSW	**9.58** (±0.40)	22.55 (±0.56)	6.13 (±0.45)	62.53 (±1.19)
	polynom-fit-SMOTE	0.48 (±0.26)	68.09 (±6.01)	**18.60** (±17.23)	60.62 (±4.75)
	SMOTE-NC	0.09 (±0.07)	70.12 (±47.90)	8.45 (±5.29)	62.97 (±0.80)
safe drive	CTGAN	0.27 (±0.21)	46.96 (±40.87)	5.90 (±1.19)	60.71 (±4.01)
	AE-*SMOTE*	0.41 (±0.14)	68.25 (±19.31)	6.21 (±0.15)	62.99 (±0.29)
	AE-*Poly*	0.35 (±0.19)	**71.91** (±28.02)	18.10 (±20.69)	**63.36** (±0.39)
	TGAN-Skip	3.89 (±0.76)	24.79 (±5.09)	5.23 (±0.66)	59.40 (±3.86)
	TGAN-WGAN	1.21 (±1.66)	33.73 (±29.69)	5.49 (±0.90)	58.35 (±5.40)
	TGAN	2.62 (±2.62)	41.99 (±40.01)	5.27 (±1.02)	58.68 (±4.92)
	DOPING	47.69 (±1.43)	**75.54** (±0.52)	**56.36** (±1.38)	**78.46** (±0.83)
	SMOTE	53.76 (±1.32)	73.38 (±1.33)	55.16 (±1.12)	77.36 (±0.90)
	CatSW	**54.54** (±0.97)	65.22 (±0.61)	56.14 (±1.22)	78.34 (±0.71)
	polynom-fit-SMOTE	51.55 (±1.08)	74.72 (±0.61)	55.19 (±1.50)	77.11 (±2.53)
	SMOTE-NC	53.23 (±1.33)	73.72 (±1.17)	55.79 (±1.50)	77.99 (±0.74)
credit default	CTGAN	51.33 (±1.29)	74.32 (±1.29)	54.57 (±3.35)	77.26 (±1.41)
	AE-*SMOTE*	50.90 (±1.34)	74.94 (±1.01)	55.38 (±1.48)	77.82 (±0.85)
	AE-*Poly*	51.14 (±1.10)	75.21 (±0.76)	56.00 (±1.08)	78.15 (±0.80)
	TGAN-Skip	50.33 (±2.35)	73.71 (±0.53)	52.16 (±0.80)	74.30 (±1.21)
	TGAN-WGAN	48.86 (±1.97)	73.38 (±1.56)	52.87 (±1.77)	74.99 (±2.78)
	TGAN	49.35 (±1.81)	73.92 (±2.03)	53.65 (±4.05)	76.91 (±2.85)

Table 4: Comparison of minority oversampling methods.

		Metric			
Dataset	**Method**	**F1 - score**	**G - score**	**PR - AUC**	**ROC - AUC**
	DOPING	35.37 (±12.57)	81.41 (±15.92)	40.89 (±13.12)	76.64 (±8.64)
	SMOTE	37.65 (±9.28)	81.33 (±13.29)	36.54 (±9.42)	75.90 (±6.95)
	CatSW	34.71 (±9.45)	50.81 (±9.48)	35.18 (±10.88)	78.47 (±9.46)
	polynom-fit-SMOTE	34.34 (±12.29)	79.49 (±15.19)	40.25 (±12.85)	78.93 (±8.34)
	SMOTE-NC	- (±−)	- (±−)	- (±−)	- (±−)
hiva	CTGAN	36.65 (±11.27)	79.93 (±9.21)	40.48 (±11.69)	76.42 (±6.67)
	AE-*SMOTE*	37.05 (±11.35)	85.87 (±8.49)	**42.15** (±13.08)	**79.92** (±6.35)
	AE-*Poly*	**39.12** (±12.26)	**86.33** (±8.65)	41.51 (±13.95)	78.70 (±8.29)
	TGAN-Skip	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN-WGAN	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN	- (±−)	- (±−)	- (±−)	- (±−)
	DOPING	95.00 (±1.84)	96.84 (±1.31)	97.73 (±1.69)	99.90 (±0.05)
	SMOTE	95.76 (±1.31)	97.11 (±0.82)	97.91 (±1.11)	99.91 (±0.04)
	CatSW	93.82 (±0.88)	94.18 (±0.99)	97.97 (±1.21)	99.91 (±0.05)
	polynom-fit-SMOTE	96.15 (±0.92)	97.08 (±1.00)	98.18 (±0.96)	99.92 (±0.05)
	SMOTE-NC	95.37 (±1.16)	96.50 (±0.75)	97.58 (±1.11)	99.90 (±0.05)
sylva	CTGAN	95.51 (±1.46)	97.27 (±0.95)	97.72 (±1.02)	99.90 (±0.05)
	AE-*SMOTE*	96.12 (±1.03)	97.35 (±1.17)	**98.48** (±0.75)	**99.93** (±0.03)
	AE-*Poly*	**96.34** (±1.06)	97.36 (±0.90)	98.38 (±0.91)	99.92 (±0.04)
	TGAN-Skip	95.33 (±1.33)	97.25 (±0.96)	97.79 (±1.17)	99.90 (±0.06)
	TGAN-WGAN	95.87 (±1.03)	97.35 (±0.79)	97.63 (±1.19)	99.91 (±0.04)
	TGAN	95.50 (±1.35)	**97.38** (±0.96)	97.85 (±1.23)	99.91 (±0.05)
	DOPING	69.70 (±1.25)	**88.15** (±0.96)	81.18 (±1.51)	96.74 (±0.27)
	SMOTE	73.10 (±1.39)	85.98 (±1.61)	81.61 (±2.82)	96.90 (±0.46)
	CatSW	58.92 (±4.59)	66.23 (±3.96)	72.85 (±4.16)	95.28 (±0.83)
	polynom-fit-SMOTE	70.46 (±2.41)	85.12 (±1.92)	79.43 (±1.65)	96.42 (±0.20)
	SMOTE-NC	73.10 (±1.39)	85.98 (±1.61)	81.61 (±2.82)	96.90 (±0.46)
satimage	CTGAN	70.04 (±1.72)	87.59 (±1.51)	80.47 (±2.17)	96.65 (±0.41)
	AE-*SMOTE*	**73.95** (±1.55)	87.17 (±1.45)	**82.52** (±2.20)	**96.94** (±0.24)
	AE-*Poly*	72.71 (±2.81)	86.47 (±1.74)	81.20 (±1.36)	96.71 (±0.25)
	TGAN-Skip	68.02 (±3.10)	86.69 (±1.85)	79.62 (±2.78)	96.39 (±0.47)
	TGAN-WGAN	69.24 (±1.88)	87.10 (±1.24)	80.36 (±1.65)	96.49 (±0.34)
	TGAN	69.72 (±1.47)	87.04 (±1.60)	80.64 (±2.19)	96.60 (±0.24)
	DOPING	80.62 (±3.65)	92.19 (±1.96)	**87.89** (±4.14)	99.01 (±0.47)
	SMOTE	81.52 (±3.63)	89.83 (±2.74)	86.98 (±4.70)	98.97 (±0.39)
	CatSW	75.84 (±5.03)	81.70 (±4.84)	84.30 (±6.14)	98.73 (±0.30)
	polynom-fit-SMOTE	**83.55** (±3.23)	91.74 (±2.53)	86.80 (±5.52)	99.02 (±0.62)
	SMOTE-NC	81.52 (±3.63)	89.83 (±2.74)	86.98 (±4.70)	98.97 (±0.39)
hypothyroid	CTGAN	80.44 (±3.49)	91.86 (±2.67)	87.34 (±4.56)	99.06 (±0.52)
	AE-*SMOTE*	81.44 (±3.10)	**92.25** (±1.61)	87.51 (±3.90)	**99.08** (±0.33)
	AE-*Poly*	81.12 (±2.34)	92.21 (±1.70)	87.36 (±3.32)	99.03 (±0.50)
	TGAN-Skip	81.52 (±4.77)	90.12 (±2.36)	84.03 (±2.92)	98.78 (±0.46)
	TGAN-WGAN	79.28 (±2.26)	89.57 (±1.00)	85.49 (±5.69)	98.93 (±0.42)
	TGAN	80.08 (±3.07)	90.47 (±3.43)	87.00 (±5.14)	99.02 (±0.44)
	DOPING	97.55 (±1.58)	99.51 (±0.34)	99.21 (±0.71)	99.77 (±0.30)
	SMOTE	97.72 (±0.87)	99.02 (±0.80)	99.81 (±0.20)	99.97 (±0.03)
	CatSW	97.97 (±1.55)	98.42 (±1.10)	99.18 (±1.97)	99.78 (±0.55)
	polynom-fit-SMOTE	98.47 (±0.85)	**99.64** (±0.33)	99.66 (±0.45)	99.93 (±0.10)
	SMOTE-NC	97.72 (±0.87)	99.02 (±0.80)	99.81 (±0.20)	99.97 (±0.03)
segment0	CTGAN	97.56 (±1.39)	99.41 (±0.51)	99.42 (±0.58)	99.87 (±0.14)
	AE-*SMOTE*	**98.70** (±0.73)	99.25 (±0.43)	**99.88** (±0.17)	**99.98** (±0.03)
	AE-*Poly*	98.24 (±1.10)	99.33 (±0.38)	99.75 (±0.26)	99.95 (±0.06)
	TGAN-Skip	97.07 (±1.18)	98.50 (±1.16)	99.54 (±0.47)	99.90 (±0.11)
	TGAN-WGAN	97.39 (±1.69)	98.74 (±1.03)	99.65 (±0.30)	99.93 (±0.06)
	TGAN	97.92 (±1.07)	99.20 (±0.45)	99.47 (±0.68)	99.85 (±0.21)
	DOPING	0.00 (±0.00)	0.00 (±0.00)	19.67 (±10.65)	53.86 (±5.98)
	SMOTE	4.76 (±12.60)	14.22 (±37.61)	13.77 (±11.77)	65.44 (±17.64)
	CatSW	12.33 (±14.53)	21.91 (±23.09)	11.26 (±12.67)	**77.44** (±10.28)
	polynom-fit-SMOTE	**16.12** (±12.06)	**31.96** (±23.00)	14.72 (±9.38)	70.29 (±10.05)
	SMOTE-NC	- (±−)	- (±−)	- (±−)	- (±−)
poker 8 9 vs 5	CTGAN	1.83 (±1.04)	8.92 (±4.19)	**38.03** (±4.88)	55.76 (±7.07)
	AE-*SMOTE*	5.80 (±5.40)	17.60 (±6.82)	23.16 (±6.54)	66.24 (±7.23)
	AE-*Poly*	6.25 (±7.29)	18.33 (±8.08)	22.44 (±8.93)	63.40 (±8.64)
	TGAN-Skip	2.20 (±2.74)	7.75 (±7.88)	23.51 (±23.61)	53.30 (±12.48)
	TGAN-WGAN	2.72 (±3.99)	9.43 (±6.71)	3.37 (±4.59)	53.86 (±8.67)
	TGAN	0.00 (±0.00)	0.00 (±0.00)	16.62 (±19.90)	47.92 (±13.42)
	DOPING	87.55 (±2.85)	**93.76** (±1.46)	94.45 (±1.33)	99.30 (±0.14)
	SMOTE	87.75 (±2.06)	92.46 (±1.66)	94.36 (±1.27)	99.31 (±0.14)
	CatSW	84.24 (±3.09)	87.04 (±3.47)	94.44 (±1.48)	99.28 (±0.12)
	polynom-fit-SMOTE	**88.37** (±2.58)	93.25 (±1.44)	94.48 (±2.17)	99.28 (±0.23)
	SMOTE-NC	87.75 (±2.06)	92.46 (±1.66)	94.36 (±1.27)	99.31 (±0.14)
page blocks0	CTGAN	87.64 (±1.78)	93.34 (±2.22)	94.56 (±1.98)	99.29 (±0.22)
	AE-*SMOTE*	88.24 (±2.08)	93.31 (±1.79)	**94.73** (±0.97)	99.31 (±0.15)
	AE-*Poly*	88.22 (±2.23)	93.63 (±1.71)	94.66 (±1.89)	**99.33** (±0.17)
	TGAN-Skip	87.05 (±1.94)	92.90 (±1.63)	93.65 (±1.97)	99.08 (±0.33)
	TGAN-WGAN	88.09 (±1.83)	93.42 (±1.37)	93.67 (±2.22)	99.14 (±0.30)
	TGAN	87.56 (±2.08)	93.44 (±1.43)	94.00 (±1.91)	99.20 (±0.21)
	DOPING	95.84 (±7.76)	99.95 (±0.09)	100.00 (±0.00)	100.00 (±0.00)
	SMOTE	98.90 (±2.91)	99.95 (±0.09)	100.00 (±0.00)	100.00 (±0.00)
	CatSW	87.20 (±9.55)	93.23 (±9.98)	98.00 (±2.71)	99.96 (±0.06)
	polynom-fit-SMOTE	98.90 (±2.91)	99.98 (±0.04)	100.00 (±0.00)	100.00 (±0.00)
	SMOTE-NC	98.90 (±2.91)	99.95 (±0.09)	100.00 (±0.00)	100.00 (±0.00)
kddcup buffer overflow vs back	CTGAN	97.40 (±4.44)	99.97 (±0.06)	100.00 (±0.00)	100.00 (±0.00)
	AE-*SMOTE*	100.00 (±0.00)	100.00 (±0.00)	100.00 (±0.00)	100.00 (±0.00)
	AE-*Poly*	100.00 (±0.00)	100.00 (±0.00)	100.00 (±0.00)	100.00 (±0.00)
	TGAN-Skip	95.24 (±12.60)	99.95 (±0.13)	100.00 (±0.00)	100.00 (±0.00)
	TGAN-WGAN	95.36 (±6.04)	98.05 (±5.05)	97.27 (±3.90)	99.96 (±0.10)
	TGAN	97.14 (±7.56)	99.97 (±0.09)	100.00 (±0.00)	100.00 (±0.00)

Table 5: Comparison of minority oversampling methods.

		Metric			
Dataset	**Method**	**F1 - score**	**G - score**	**PR - AUC**	**ROC - AUC**
	DOPING	98.70 (± 2.23)	99.96 (± 0.08)	99.90 (± 0.26)	100.00 (± 0.01)
	SMOTE	98.64 (± 2.32)	99.96 (± 0.08)	100.00 (± 0.00)	100.00 (± 0.00)
	CatSW	97.46 (± 3.54)	98.10 (± 2.38)	99.89 (± 0.30)	100.00 (± 0.01)
	polynom-fit-SMOTE	98.76 (± 2.12)	98.78 (± 2.08)	99.95 (± 0.14)	100.00 (± 0.01)
	SMOTE-NC	98.64 (± 2.32)	99.96 (± 0.08)	100.00 (± 0.00)	100.00 (± 0.00)
kddcup guess passwd vs satan	CTGAN	**100.00** (± 0.00)	**100.00** (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)
	AE-*SMOTE*	99.32 (± 1.80)	99.98 (± 0.06)	99.95 (± 0.14)	100.00 (± 0.01)
	AE-*Poly*	99.32 (± 1.80)	99.98 (± 0.06)	99.95 (± 0.14)	100.00 (± 0.01)
	TGAN-Skip	98.08 (± 2.40)	98.76 (± 2.06)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN-WGAN	96.59 (± 3.72)	99.30 (± 1.57)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN	98.64 (± 2.32)	99.96 (± 0.08)	99.86 (± 0.37)	99.99 (± 0.02)
	DOPING	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)
	SMOTE	95.24 (± 12.60)	99.96 (± 0.12)	99.64 (± 0.94)	99.99 (± 0.01)
	CatSW	94.84 (± 9.68)	96.56 (± 5.94)	97.22 (± 7.37)	98.19 (± 4.78)
	polynom-fit-SMOTE	95.92 (± 6.97)	99.96 (± 0.08)	99.64 (± 0.94)	99.99 (± 0.01)
	SMOTE-NC	95.24 (± 12.60)	99.96 (± 0.12)	99.64 (± 0.94)	99.99 (± 0.01)
kddcup land vs satan	CTGAN	91.98 (± 10.79)	95.43 (± 5.60)	97.77 (± 4.02)	99.97 (± 0.06)
	AE-*SMOTE*	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)
	AE-*Poly*	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN-Skip	95.24 (± 5.94)	95.48 (± 5.64)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN-WGAN	95.24 (± 5.94)	95.48 (± 5.64)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN	96.83 (± 5.42)	96.98 (± 5.15)	100.00 (± 0.00)	100.00 (± 0.00)
	DOPING	84.35 (± 13.67)	99.89 (± 0.09)	98.30 (± 4.50)	99.67 (± 0.87)
	SMOTE	97.14 (± 7.56)	99.95 (± 0.06)	100.00 (± 0.00)	100.00 (± 0.00)
	CatSW	91.34 (± 9.91)	91.99 (± 8.95)	98.15 (± 4.88)	99.98 (± 0.04)
	polynom-fit-SMOTE	96.37 (± 6.26)	99.95 (± 0.06)	100.00 (± 0.00)	100.00 (± 0.00)
	SMOTE-NC	97.14 (± 7.56)	99.95 (± 0.06)	100.00 (± 0.00)	100.00 (± 0.00)
kddcup rootkit imap vs back	CTGAN	91.84 (± 7.64)	99.94 (± 0.09)	99.20 (± 2.13)	99.99 (± 0.02)
	AE-*SMOTE*	95.92 (± 6.97)	99.97 (± 0.06)	100.00 (± 0.00)	100.00 (± 0.00)
	AE-*Poly*	96.37 (± 6.26)	99.97 (± 0.06)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN-Skip	92.29 (± 7.30)	98.44 (± 3.97)	100.00 (± 0.00)	100.00 (± 0.00)
	TGAN-WGAN	90.25 (± 6.76)	98.43 (± 3.96)	94.70 (± 9.63)	99.93 (± 0.14)
	TGAN	93.20 (± 12.85)	99.95 (± 0.09)	98.09 (± 5.06)	99.97 (± 0.09)
	DOPING	65.97 (± 2.97)	81.18 (± 1.49)	77.46 (± 2.51)	90.00 (± 1.40)
	SMOTE	66.77 (± 3.40)	78.62 (± 2.82)	76.30 (± 3.42)	89.57 (± 1.72)
	CatSW	66.63 (± 4.15)	72.46 (± 4.26)	75.42 (± 4.65)	89.27 (± 2.19)
	polynom-fit-SMOTE	**67.21** (± 3.90)	**81.21** (± 1.80)	77.78 (± 2.42)	90.12 (± 1.32)
	SMOTE-NC	66.77 (± 3.40)	78.62 (± 2.82)	76.30 (± 3.42)	89.57 (± 1.72)
ada	CTGAN	65.95 (± 2.71)	80.49 (± 1.28)	77.38 (± 2.64)	90.01 (± 1.40)
	AE-*SMOTE*	66.51 (± 3.35)	81.01 (± 1.90)	**77.88** (± 2.41)	90.11 (± 1.26)
	AE-*Poly*	66.59 (± 2.37)	81.18 (± 1.24)	77.85 (± 2.17)	**90.19** (± 1.30)
	TGAN-Skip	65.22 (± 3.58)	78.31 (± 2.30)	75.37 (± 2.92)	88.74 (± 1.83)
	TGAN-WGAN	65.12 (± 3.14)	79.17 (± 1.77)	75.77 (± 2.55)	89.22 (± 1.58)
	TGAN	65.86 (± 3.36)	78.25 (± 2.27)	75.41 (± 3.07)	88.73 (± 1.56)
	DOPING	0.00 (± 0.00)	0.00 (± 0.00)	1.10 (± 0.35)	68.75 (± 5.49)
	SMOTE	7.97 (± 6.12)	21.68 (± 14.13)	9.42 (± 10.37)	75.18 (± 6.57)
	CatSW	3.19 (± 1.21)	12.61 (± 2.70)	3.28 (± 3.37)	73.92 (± 9.54)
	polynom-fit-SMOTE	5.25 (± 5.65)	15.99 (± 8.24)	3.26 (± 6.40)	73.60 (± 9.44)
	SMOTE-NC	7.97 (± 6.12)	21.68 (± 14.13)	9.42 (± 10.37)	75.18 (± 6.57)
abalone19	CTGAN	2.86 (± 7.56)	7.12 (± 18.84)	4.52 (± 6.64)	66.14 (± 8.95)
	AE-*SMOTE*	4.22 (± 4.02)	12.21 (± 11.52)	**12.24** (± 9.48)	73.54 (± 6.35)
	AE-*Poly*	4.37 (± 2.43)	15.09 (± 5.13)	11.48 (± 18.39)	72.24 (± 6.26)
	TGAN-Skip	1.36 (± 3.60)	3.68 (± 9.73)	1.44 (± 0.55)	72.97 (± 5.81)
	TGAN-WGAN	1.36 (± 1.99)	5.43 (± 7.23)	1.19 (± 0.51)	69.73 (± 9.53)
	TGAN	0.00 (± 0.00)	0.00 (± 0.00)	1.25 (± 0.48)	71.05 (± 4.95)
	DOPING	0.00 (± 0.00)	0.00 (± 0.00)	6.98 (± 5.23)	64.04 (± 7.16)
	SMOTE	18.74 (± 12.08)	35.92 (± 7.66)	12.19 (± 8.28)	73.19 (± 11.94)
	CatSW	14.04 (± 7.77)	26.62 (± 13.06)	10.70 (± 6.59)	69.44 (± 9.54)
	polynom-fit-SMOTE	12.11 (± 11.34)	25.40 (± 12.76)	9.72 (± 5.41)	**73.28** (± 8.32)
	SMOTE-NC	18.74 (± 12.08)	35.92 (± 7.66)	12.19 (± 8.28)	73.19 (± 11.94)
winequality red 4	CTGAN	11.96 (± 15.58)	30.94 (± 14.33)	10.96 (± 12.29)	68.00 (± 7.79)
	AE-*SMOTE*	**21.28** (± 12.44)	**44.18** (± 4.89)	**14.26** (± 7.94)	72.55 (± 8.70)
	AE-*Poly*	14.48 (± 8.27)	36.06 (± 28.72)	13.83 (± 10.89)	70.85 (± 10.77)
	TGAN-Skip	8.37 (± 5.38)	22.16 (± 8.01)	7.46 (± 3.04)	68.54 (± 8.47)
	TGAN-WGAN	10.33 (± 5.96)	26.27 (± 9.14)	8.86 (± 4.80)	69.00 (± 6.29)
	TGAN	4.64 (± 6.10)	16.71 (± 22.83)	8.95 (± 9.92)	66.99 (± 8.58)
	DOPING	98.71 (± 0.58)	99.65 (± 0.52)	99.33 (± 0.52)	99.77 (± 0.28)
	SMOTE	99.14 (± 0.61)	99.78 (± 0.24)	99.60 (± 0.38)	99.92 (± 0.11)
	CatSW	98.57 (± 0.77)	98.84 (± 0.63)	**99.74** (± 0.23)	99.96 (± 0.07)
	polynom-fit-SMOTE	98.98 (± 0.85)	99.70 (± 0.36)	99.73 (± 0.28)	**99.96** (± 0.06)
	SMOTE-NC	99.14 (± 0.61)	99.78 (± 0.24)	99.60 (± 0.38)	99.92 (± 0.11)
fabert	CTGAN	99.07 (± 0.65)	99.72 (± 0.34)	99.41 (± 0.43)	99.75 (± 0.26)
	AE-*SMOTE*	99.16 (± 0.58)	**99.84** (± 0.14)	99.50 (± 0.31)	99.89 (± 0.08)
	AE-*Poly*	**99.16** (± 0.47)	99.82 (± 0.20)	99.47 (± 0.32)	99.87 (± 0.23)
	TGAN-Skip	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
	TGAN-WGAN	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
	TGAN	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
	DOPING	88.79 (± 1.62)	96.76 (± 1.16)	91.81 (± 3.48)	96.29 (± 1.09)
	SMOTE	88.93 (± 1.64)	96.80 (± 0.91)	91.27 (± 2.72)	95.99 (± 1.24)
	CatSW	88.01 (± 2.55)	93.93 (± 2.09)	91.48 (± 2.65)	96.69 (± 0.88)
	polynom-fit-SMOTE	88.56 (± 2.00)	96.77 (± 1.08)	92.16 (± 3.17)	96.45 (± 1.09)
	SMOTE-NC	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
ads	CTGAN	89.05 (± 1.73)	96.76 (± 0.73)	92.86 (± 1.87)	**97.09** (± 0.72)
	AE-*SMOTE*	89.05 (± 1.70)	**97.17** (± 0.64)	92.92 (± 2.81)	96.91 (± 0.86)
	AE-*Poly*	**89.20** (± 1.98)	97.06 (± 0.67)	**93.10** (± 2.67)	96.91 (± 0.72)
	TGAN-Skip	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
	TGAN-WGAN	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)	- ($\pm -$)
	TGAN	0.00 (± 0.00)	0.00 (± 0.00)	0.00 (± 0.00)	0.00 (± 0.00)

Table 6: Comparison of minority oversampling methods.

		Metric			
Dataset	**Method**	**F1 - score**	**G - score**	**PR - AUC**	**ROC - AUC**
	DOPING	87.70 (±3.85)	92.79 (±2.83)	89.91 (±3.70)	98.47 (±0.59)
	SMOTE	87.49 (±4.01)	92.22 (±2.23)	91.13 (±3.61)	98.82 (±0.49)
	CatSW	84.56 (±3.16)	88.04 (±2.04)	89.61 (±3.64)	98.55 (±0.50)
	polynom-fit-SMOTE	86.93 (±3.51)	91.73 (±3.01)	88.99 (±4.00)	98.65 (±0.43)
	SMOTE-NC	87.49 (±4.01)	92.22 (±2.23)	91.13 (±3.61)	98.82 (±0.49)
sick euthyroid	CTGAN	87.57 (±3.32)	92.61 (±2.65)	89.32 (±3.75)	98.42 (±0.67)
	AE-*SMOTE*	88.70 (±2.65)	**93.75** (±3.25)	89.43 (±3.34)	98.44 (±0.86)
	AE-*Poly*	**89.08** (±3.30)	93.49 (±2.64)	89.06 (±5.50)	98.64 (±0.78)
	TGAN-Skip	86.44 (±1.96)	91.65 (±1.60)	88.20 (±4.64)	97.82 (±0.91)
	TGAN-WGAN	87.52 (±3.93)	92.76 (±2.90)	88.03 (±4.36)	98.06 (±0.88)
	TGAN	86.61 (±3.33)	91.71 (±2.63)	86.61 (±6.22)	98.02 (±1.03)
	DOPING	87.26 (±3.15)	97.44 (±0.68)	96.79 (±0.78)	99.57 (±0.29)
	SMOTE	91.38 (±1.52)	97.57 (±0.78)	97.14 (±0.86)	99.59 (±0.25)
	CatSW	88.44 (±3.54)	90.39 (±3.38)	96.54 (±1.03)	99.53 (±0.24)
	polynom-fit-SMOTE	**91.57** (±1.69)	97.30 (±0.81)	97.50 (±0.76)	99.66 (±0.19)
	SMOTE-NC	91.38 (±1.52)	97.57 (±0.78)	97.14 (±0.86)	99.59 (±0.25)
ISOLET	CTGAN	88.24 (±1.72)	97.38 (±0.99)	97.05 (±0.71)	99.62 (±0.20)
	AE-*SMOTE*	89.46 (±1.86)	97.51 (±0.65)	97.23 (±0.71)	99.62 (±0.20)
	AE-*Poly*	89.92 (±2.27)	**97.66** (±0.90)	**97.56** (±0.60)	**99.69** (±0.11)
	TGAN-Skip	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN-WGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	DOPING	47.69 (±8.13)	81.65 (±8.13)	55.28 (±5.82)	91.12 (±2.59)
	SMOTE	55.58 (±6.46)	79.17 (±6.91)	56.72 (±5.39)	90.99 (±2.76)
	CatSW	43.49 (±4.07)	54.02 (±3.33)	51.58 (±6.74)	89.87 (±2.78)
	polynom-fit-SMOTE	53.54 (±5.75)	77.79 (±3.24)	56.30 (±5.80)	90.67 (±2.74)
	SMOTE-NC	55.58 (±6.46)	79.17 (±6.91)	56.72 (±5.39)	90.99 (±2.76)
us crime	CTGAN	48.84 (±5.34)	81.45 (±7.92)	57.20 (±5.44)	90.91 (±2.89)
	AE-*SMOTE*	51.94 (±6.02)	82.26 (±7.32)	58.02 (±4.99)	91.46 (±2.80)
	AE-*Poly*	53.30 (±6.14)	**83.85** (±5.96)	**59.01** (±4.65)	**91.56** (±2.31)
	TGAN-Skip	51.82 (±6.47)	80.76 (±5.09)	56.60 (±6.25)	90.77 (±3.22)
	TGAN-WGAN	50.39 (±6.91)	79.46 (±6.42)	56.66 (±7.31)	90.57 (±2.11)
	TGAN	50.16 (±6.15)	80.17 (±5.70)	54.78 (±5.65)	90.63 (±1.75)
	DOPING	0.00 (±0.00)	0.00 (±0.00)	9.00 (±2.33)	54.38 (±5.02)
	SMOTE	12.32 (±3.92)	28.14 (±2.47)	10.08 (±2.33)	59.04 (±2.66)
	CatSW	14.87 (±1.64)	28.53 (±2.29)	8.57 (±1.45)	53.77 (±4.60)
	polynom-fit-SMOTE	11.40 (±4.13)	33.24 (±7.46)	10.55 (±2.34)	58.74 (±6.09)
	SMOTE-NC	12.32 (±3.92)	28.14 (±2.47)	10.08 (±2.33)	59.04 (±2.66)
yeast ml8	CTGAN	12.52 (±2.01)	33.30 (±4.42)	10.11 (±1.86)	58.08 (±3.66)
	AE-*SMOTE*	**15.36** (±3.66)	34.84 (±9.50)	**25.24** (±8.93)	**59.48** (±4.46)
	AE-*Poly*	11.79 (±6.15)	**42.44** (±35.13)	19.03 (±23.72)	57.44 (±4.00)
	TGAN-Skip	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN-WGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	DOPING	8.25 (±12.14)	57.12 (±40.33)	28.52 (±10.14)	78.62 (±4.32)
	SMOTE	30.82 (±5.94)	66.77 (±17.94)	30.33 (±7.35)	79.35 (±3.02)
	CatSW	24.06 (±2.59)	38.83 (±1.57)	20.76 (±6.38)	74.12 (±2.93)
	polynom-fit-SMOTE	11.41 (±7.94)	57.52 (±45.77)	25.35 (±8.43)	77.63 (±2.12)
	SMOTE-NC	30.82 (±5.94)	66.77 (±17.94)	30.33 (±7.35)	79.35 (±3.02)
scene	CTGAN	9.09 (±10.90)	61.19 (±42.97)	29.12 (±8.16)	76.84 (±3.34)
	AE-*SMOTE*	27.71 (±5.89)	73.03 (±10.35)	30.94 (±8.16)	**79.90** (±1.98)
	AE-*Poly*	19.77 (±7.86)	**76.87** (±23.98)	**31.64** (±7.65)	79.40 (±2.37)
	TGAN-Skip	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN-WGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	TGAN	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
	DOPING	87.85 (±4.24)	94.66 (±1.82)	95.01 (±1.65)	99.49 (±0.22)
	SMOTE	89.93 (±4.01)	94.70 (±3.31)	95.99 (±2.29)	99.63 (±0.27)
	CatSW	83.58 (±6.13)	86.80 (±4.65)	93.30 (±4.08)	99.24 (±0.69)
	polynom-fit-SMOTE	87.35 (±3.67)	92.76 (±1.33)	92.68 (±2.82)	99.03 (±0.66)
	SMOTE-NC	89.93 (±4.01)	94.70 (±3.31)	95.99 (±2.29)	99.63 (±0.27)
thyroid sick	CTGAN	87.24 (±5.07)	94.58 (±2.60)	93.41 (±3.83)	99.20 (±0.57)
	AE-*SMOTE*	89.57 (±4.34)	**95.47** (±1.87)	95.57 (±2.43)	99.45 (±0.45)
	AE-*Poly*	89.18 (±4.45)	95.02 (±1.63)	95.33 (±2.59)	99.45 (±0.45)
	TGAN-Skip	84.12 (±4.04)	92.00 (±1.91)	92.68 (±5.26)	99.01 (±0.35)
	TGAN-WGAN	85.64 (±3.72)	93.09 (±1.51)	92.53 (±5.13)	98.86 (±0.61)
	TGAN	86.90 (±4.44)	92.79 (±2.16)	94.33 (±4.09)	99.25 (±0.39)
	DOPING	2.96 (±1.89)	55.32 (±40.72)	16.26 (±4.61)	73.61 (±2.49)
	SMOTE	20.66 (±1.40)	55.25 (±17.33)	16.07 (±2.22)	73.41 (±1.61)
	CatSW	**21.02** (±1.35)	34.79 (±1.28)	17.66 (±1.55)	73.89 (±1.86)
	polynom-fit-SMOTE	9.43 (±4.23)	54.76 (±12.25)	16.23 (±2.08)	73.71 (±2.03)
	SMOTE-NC	20.66 (±1.40)	55.25 (±17.33)	16.07 (±2.22)	73.41 (±1.61)
coil 2000	CTGAN	2.09 (±2.41)	42.25 (±43.88)	**20.23** (±14.73)	73.16 (±3.65)
	AE-*SMOTE*	4.81 (±3.86)	58.49 (±24.57)	16.31 (±2.97)	73.36 (±1.49)
	AE-*Poly*	6.53 (±3.11)	**64.87** (±16.21)	17.00 (±2.38)	**74.64** (±2.43)
	TGAN-Skip	9.41 (±3.57)	39.67 (±39.54)	15.07 (±3.61)	69.39 (±9.41)
	TGAN-WGAN	12.49 (±4.09)	38.56 (±41.00)	15.16 (±3.65)	72.29 (±4.76)
	TGAN	11.71 (±3.00)	39.28 (±13.17)	14.09 (±3.60)	65.86 (±9.75)
	DOPING	6.55 (±9.16)	25.69 (±33.75)	19.77 (±6.55)	**79.89** (±4.27)
	SMOTE	20.10 (±6.65)	44.72 (±21.54)	18.28 (±6.42)	79.49 (±4.11)
	CatSW	19.55 (±4.72)	34.50 (±5.13)	17.29 (±5.37)	75.84 (±8.68)
	polynom-fit-SMOTE	20.28 (±7.77)	**57.22** (±19.44)	**30.19** (±16.31)	70.82 (±12.93)
	SMOTE-NC	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
solar flare M0	CTGAN	23.65 (±5.60)	48.47 (±36.59)	19.12 (±9.88)	74.63 (±4.89)
	AE-*SMOTE*	**27.70** (±5.59)	55.17 (±7.93)	19.11 (±6.77)	78.37 (±5.03)
	AE-*Poly*	26.26 (±9.48)	52.95 (±9.81)	19.75 (±3.25)	77.87 (±5.13)
	TGAN-Skip	20.03 (±11.37)	43.50 (±30.35)	20.84 (±10.10)	75.71 (±4.81)
	TGAN-WGAN	18.02 (±9.72)	45.10 (±23.88)	18.34 (±6.93)	71.41 (±5.94)
	TGAN	9.88 (±9.57)	45.82 (±45.31)	20.25 (±9.68)	73.45 (±6.29)

Table 7: Comparison of minority oversampling methods.

		Metric			
Dataset	**Method**	**F1 - score**	**G - score**	**PR - AUC**	**ROC - AUC**
wine quality	*DOPING*	29.13 (±6.22)	80.57 (±4.87)	33.69 (±4.42)	83.58 (±1.61)
	SMOTE	39.00 (±6.01)	68.37 (±10.34)	32.35 (±9.82)	83.20 (±1.98)
	CatSW	29.73 (±2.49)	44.31 (±1.93)	26.85 (±5.58)	82.52 (±2.52)
	polynom-fit-SMOTE	36.70 (±6.83)	64.53 (±7.64)	28.43 (±7.22)	83.11 (±2.44)
	SMOTE-NC	39.00 (±6.01)	68.37 (±10.34)	32.35 (±9.82)	83.20 (±1.98)
	CTGAN	32.35 (±5.86)	74.70 (±7.43)	32.14 (±5.52)	83.59 (±3.48)
	AE-*SMOTE*	32.45 (±3.32)	**83.81** (±11.56)	**33.96** (±7.22)	82.65 (±2.80)
	AE-*Poly*	29.84 (±8.64)	80.67 (±10.60)	32.49 (±6.78)	**84.40** (±2.30)
	TGAN-Skip	29.96 (±6.28)	66.51 (±9.00)	27.56 (±5.21)	82.54 (±0.86)
	TGAN-WGAN	28.81 (±7.51)	71.59 (±10.78)	32.30 (±7.17)	82.63 (±1.92)
	TGAN	27.34 (±8.67)	68.84 (±10.88)	27.86 (±6.14)	83.01 (±2.44)
webpage	*DOPING*	**79.70** (±1.64)	**94.48** (±1.39)	84.14 (±1.87)	97.80 (±0.45)
	SMOTE	79.49 (±1.77)	93.82 (±1.54)	84.42 (±1.79)	98.04 (±0.52)
	CatSW	63.41 (±5.35)	70.73 (±4.33)	79.24 (±3.76)	97.82 (±0.62)
	polynom-fit-SMOTE	78.88 (±2.42)	94.21 (±2.07)	**84.87** (±1.92)	**98.14** (±0.52)
	SMOTE-NC	- (±−)	- (±−)	- (±−)	- (±−)
	CTGAN	78.40 (±2.02)	94.03 (±2.01)	83.08 (±2.38)	97.31 (±0.74)
	AE-*SMOTE*	76.35 (±2.06)	88.31 (±2.42)	81.76 (±2.28)	97.78 (±0.66)
	AE-*Poly*	75.68 (±2.77)	86.33 (±2.95)	81.50 (±2.00)	97.80 (±0.43)
	TGAN-Skip	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN-WGAN	- (±−)	- (±−)	- (±−)	- (±−)
	TGAN	- (±−)	- (±−)	- (±−)	- (±−)
ozone level	*DOPING*	4.72 (±9.02)	19.85 (±37.87)	21.71 (±10.98)	88.25 (±3.00)
	SMOTE	37.34 (±11.82)	63.74 (±21.92)	32.05 (±11.46)	91.34 (±2.12)
	CatSW	30.41 (±7.37)	44.37 (±6.39)	**32.96** (±10.89)	89.27 (±3.28)
	polynom-fit-SMOTE	29.38 (±7.63)	54.67 (±13.63)	22.14 (±4.83)	89.18 (±3.49)
	SMOTE-NC	37.34 (±11.82)	63.74 (±21.92)	32.05 (±11.46)	91.34 (±2.12)
	CTGAN	10.29 (±2.23)	32.81 (±12.89)	21.53 (±11.46)	86.04 (±7.96)
	AE-*SMOTE*	27.35 (±14.95)	**72.37** (±18.67)	30.61 (±8.95)	89.06 (±3.19)
	AE-*Poly*	21.15 (±13.19)	59.85 (±34.17)	30.07 (±5.71)	88.82 (±5.30)
	TGAN-Skip	6.14 (±5.86)	32.75 (±33.01)	16.54 (±11.97)	80.31 (±3.45)
	TGAN-WGAN	12.48 (±7.47)	54.75 (±27.34)	19.90 (±4.86)	87.62 (±3.60)
	TGAN	7.60 (±6.43)	24.68 (±18.47)	22.75 (±16.68)	83.35 (±9.37)
kc1	*DOPING*	40.20 (±8.74)	71.68 (±4.95)	46.41 (±7.18)	81.09 (±2.29)
	SMOTE	47.03 (±3.41)	69.57 (±7.91)	45.76 (±5.31)	81.04 (±2.19)
	CatSW	43.65 (±2.52)	53.47 (±1.89)	41.43 (±8.37)	79.93 (±3.36)
	polynom-fit-SMOTE	45.03 (±6.38)	71.51 (±4.44)	47.09 (±6.94)	81.83 (±2.60)
	SMOTE-NC	47.03 (±3.41)	69.57 (±7.91)	45.76 (±5.31)	81.04 (±2.19)
	CTGAN	38.96 (±5.29)	72.66 (±4.31)	47.21 (±6.65)	81.84 (±2.93)
	AE-*SMOTE*	43.50 (±2.26)	**73.40** (±4.31)	**50.55** (±7.63)	82.12 (±2.63)
	AE-*Poly*	43.26 (±5.30)	73.39 (±4.13)	48.42 (±5.72)	**82.36** (±2.52)
	TGAN-Skip	37.60 (±6.61)	69.44 (±3.43)	42.91 (±4.02)	80.41 (±2.33)
	TGAN-WGAN	36.91 (±4.53)	70.09 (±5.25)	43.87 (±4.78)	80.12 (±2.04)
	TGAN	42.24 (±6.95)	73.12 (±5.03)	46.55 (±5.95)	81.04 (±2.22)

C DATASETS

We provide a overview of all the public datasets used for the evaluation of the proposed method. Links are provided for each dataset in each row Dua & Graff (2017), Alcala-Fdez et al. (2011), Lemaitre et al. (2016).

ID	Dataset	Categorical	Continuous	Imbalance Ratio	Num Samples
0	solar flare M0	32	0	1:21	1389
1	winequality red 4	0	11	1:31	1599
2	kddcup land vs satan	0	30	1:78	1610
3	kddcup guess passwd vs satan	0	38	1:32	1642
4	us crime	0	100	1:14	1994
5	poker 8 9 vs 5	25	0	1:84	2075
6	kc1	0	21	1:7	2109
7	kddcup rootkit imap vs back	0	47	1:102	2225
8	kddcup buffer overflow vs back	0	31	1:75	2233
9	segment0	0	23	1:8	2308
10	scene	0	294	1:15	2407
11	yeast ml8	0	103	1:15	2417
12	ozone level	0	72	1:36	2536
13	hypothyroid	0	25	1:22	3163
14	sick euthyroid	36	6	1:12	3163
15	ads	1555	0	1:8	3279
16	thyroid sick	45	7	1:17	3772
17	hiva	1617	0	1:29	3845
18	ada	0	47	1:5	4147
19	abalone19	0	8	1:131	4174
20	wine quality	0	11	1:28	4898
21	page blocks0	0	10	1:11	5472
22	satimage	0	36	1:11	6435
23	ISOLET	0	617	1:14	7797
24	coil 2000	57	28	1:18	9822
25	sylva	172	40	1:17	13086
26	letters	0	16	1:28	19999
27	fabert	0	800	1:17	24711
28	credit default	9	14	1:6	30000
29	webpage	300	0	1:36	34780
30	adult	8	6	1:14	48840
31	protein homo	0	74	1:113	145751
32	fraud	0	29	1:580	284807
33	covtype	44	10	1:48	581011
34	safe drive	14	43	1:28	595212

- **Letters**: This dataset is a multi-class dataset converted into an imbalanced binary dataset by merging all letters besides the least frequent letter (h) to be labeled as class 0 and h as class 1.
- **Forest Cover Type**: This dataset is a multiclass classification problem with 7 clases, we convert this dataset into an imbalanced binary problem by merging the least frequent classes $\{4, 5\}$ as class 1 and the rest $\{1, 2, 3, 6, 7\}$ as class 0. https://archive.ics.uci.edu/ml/datasets/covertype

D SOFTWARE

To ensure reproducibility, all experiments were run using the same random seed (42) and software versions.

Table 8: Python dependencies.

Dependency	**Version**
python	3.6.1
pytorch	1.0.0
cuda100	1.0
numpy	1.15.4
pandas	0.24.1
scikit-learn	0.20.1
scipy	1.1.0
tqdm	4.28.1
matplotlib	3.0.1
imbalanced-learn	0.7.0