Conditional Data Synthesis with Deep Generative Models for Imbalanced Dataset Oversampling

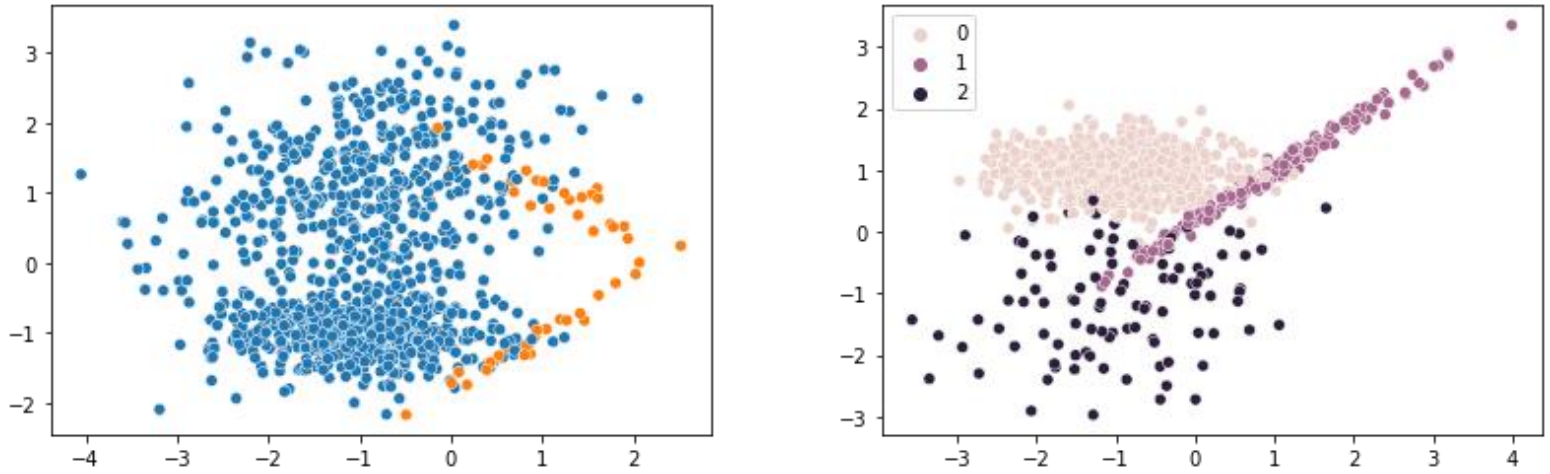
L. Akritidis[1], A. Fevgas[2], M. Alamaniotis[3], P. Bozanis[1]

[1]School of Science and Technology, International Hellenic University
[2]Department of Electrical and Computer Engineering, University of Thessaly
[3]Department of Electrical and Computer Engineering, University of Texas at San Antonio

33rd IEEE International Conference on Tools with Artificial Intelligence
November 2023
Atlanta, USA

Imbalanced datasets

- Class imbalance problem: the uneven distribution of the training data samples to the classes of a dataset.



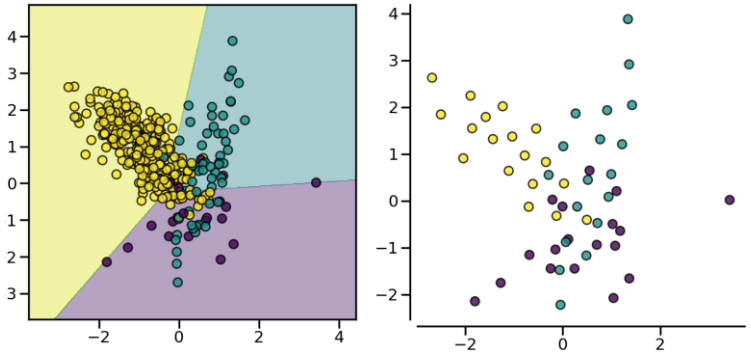
- Very significant problem, appears everywhere:
  - Cybersecurity, Bioinformatics, Natural Language Processing, Image Processing.
- Severely degrades classification performance.

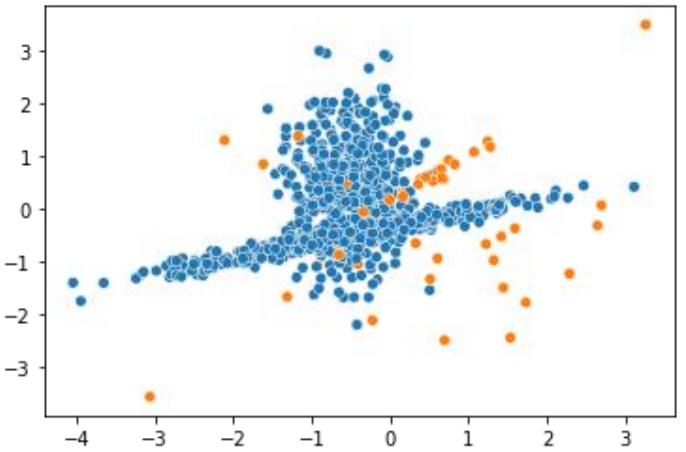
Data imbalance is critical

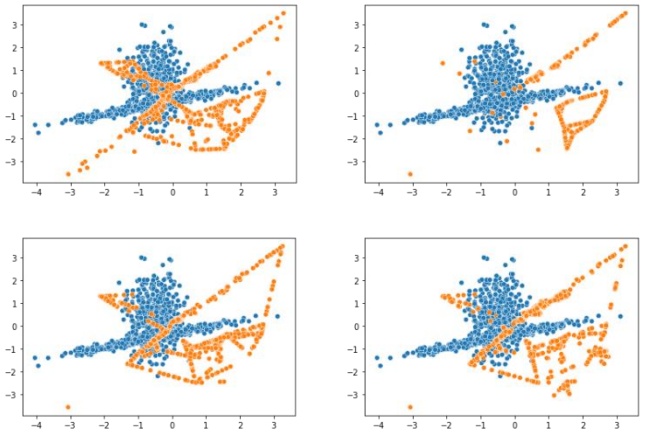
- Severely degrades classification performance.
- Predictors become strongly biased towards the majority class.
- They cannot effectively learn the minority classes.
- Measures are required to confront class imbalance before training machine learning models.

Confronting data imbalance

- Oversampling: creates synthetic samples for the minority class/es.
- Undersampling: reduces the population of the majority class samples.







- Hybrid sampling: combines over- & under-sampling.
- Cost-sensitive learning.
- Ensemble learning (bagging, boosting).

SafeBorder-GAN, SafeBorder-VAE

- We introduce an oversampling technique to mitigate data imbalance using deep generative networks.
- We focus on Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs).
- We propose two variants of GAN and VAE, termed SB-GAN and SB-VAE, respectively.
  - SB: SafeBorder sampling.

Generative Adversarial Networks (GANs)

- Inside a GAN, two Neural Networks compete each other: a Discriminator D and a Generator G.
  - D is a binary classifier trained to distinguish whether its input data samples are real or not.
  - G learns to synthesize artificial data instances with the aim of deceiving D, so that its output is classified as real.
- Formally, D and G play the following zero-sum game:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_z(z)}\left[\log\left(1 - D\left(G(\mathbf{z})\right)\right)\right]$$

- Conditional GANs learn to synthesize data of a specific class $\mathbf{y}$:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_z(z)}\left[\log\left(1 - D\left(G(\mathbf{z}|\mathbf{y})\right)\right)\right]$$

Variational Autoencoders (VAEs)

- VAEs implement the Encoder E-Decoder D architecture.
  - E encodes its input $\mathbf{x}$ as a probability distribution $p_\theta(\mathbf{x}|\mathbf{z})$ over a latent space $\mathbf{Z}$.
  - D is trained to generate meaningful data from samples taken randomly from $p_\theta(\mathbf{x}|\mathbf{z})$.
  - $p_\theta(\mathbf{x}|\mathbf{z})$: We usually try to learn a standard Gaussian with $\mu = 0, \sigma = 1$.
- The loss function quantifies the reconstruction error of VAE and the divergence between the learned distribution and a standard Gaussian.

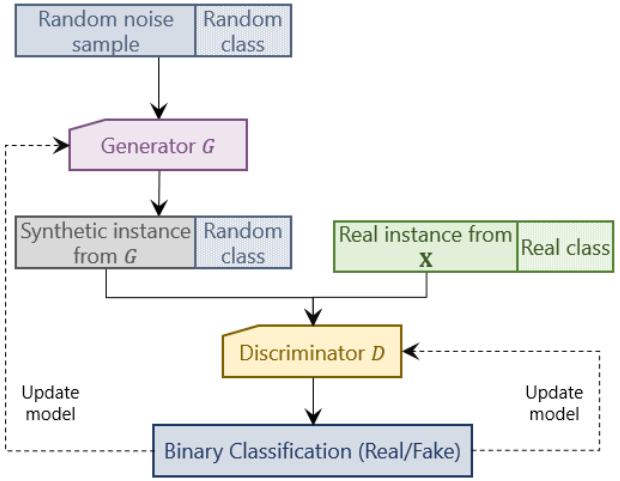
$$\text{Loss}_{\text{VAE}} = -KL\big(q_\phi(\mathbf{z}|\mathbf{x})\big|p_\theta(\mathbf{z})\big) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]$$

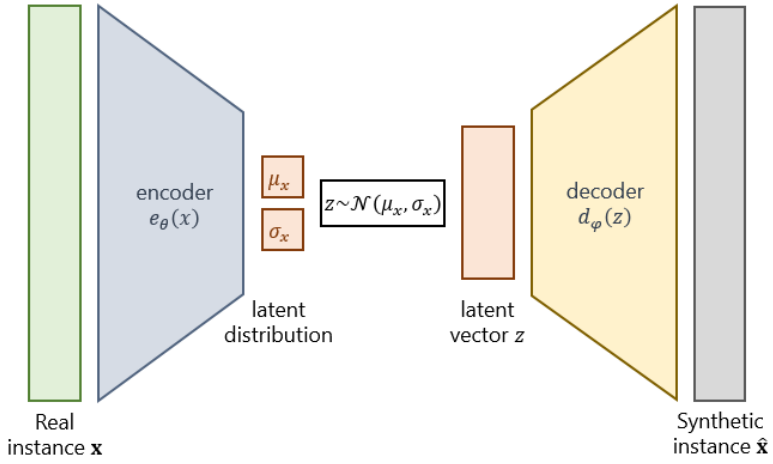
- In a conditional VAE, the loss function is formulated as follows:

$$\text{Loss}_{\text{CVAE}} = -KL\big(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})\big|p_\theta(\mathbf{z})\big) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})]$$

Conditional Generators

- SB-GAN and SB-VAE are conditional generators.
- They can create artificial samples belonging to a particular class.
  - The one-hot-encoded class representation is concatenated with the input vector.
  - The concatenated representation is fed to both Discriminator & Generator (in SB-GAN), or Encoder & Decoder (in SB-VAE).





SafeBorder Sampling

- SB-GAN and SB-VAE are fed not with the entire minority classes, but only with carefully selected data instances. The selection criterion depends on the classes of the neighboring data points.
- A preprocessing layer identifies the outlier and noisy samples and excludes them from training.
- In case all (or the majority of) the neighbors of a training sample belong to different classes, then the sample is considered as an outlier.
  - it does not comply with the distribution of its own class.
- Such samples are ignored during training, thus improving the learned class distributions.

SafeBorder Sampling Algorithm

- The Safe-Borderline (SB) technique specifies a hyper-sphere S of radius r around each minority sample $\mathbf{x}_j^{(i)}$.
- Then, $\mathbf{x}_j^{(i)}$ is tagged according to the classes of the elements inside its surrounding hyper-sphere S:
  - *Outlier*: If all samples in S belong to a different class than $\mathbf{x}_j^{(i)}$.
  - *Isolated*: If S is empty.
  - *Borderline*: If S contains mixed samples from all classes.
  - *Safe*, or *Core*: If all the samples inside S belong to the same class as $\mathbf{x}_j^{(i)}$.

Algorithm 1 SafeBorder sampling

Input: Training set $\mathbf{X}$, radius r, α
Output: Safe-Borderline set $\mathbf{X}_{SB}$

$\mathbf{X}^{(i)} \leftarrow [\,]$ *// minority class examples*
for each training example $\mathbf{x}_j \in \mathbf{X}$ **do**
  if $y_j == y^{(i)}$ **then** $\mathbf{X}^{(i)} \leftarrow \mathbf{X}^{(i)}.append(\mathbf{x})$

$\mathbf{X}_{iso}^{(i)} \leftarrow [\,]$ *// isolated samples*
$\mathbf{X}_{out}^{(i)} \leftarrow [\,]$ *// outliers*
$\mathbf{X}_{sf}^{(i)} \leftarrow [\,]$ *// safe samples*
$\mathbf{X}_{b}^{(i)} \leftarrow [\,]$ *// borderline samples*
for each minority sample $\mathbf{x}_j \in \mathbf{X}^{(i)}$ **do**
  $\mathbf{X}_{j,nn} \leftarrow \text{FetchPointsInRadius}(\mathbf{X}, \mathbf{x}_j, r)$
  if $|\mathbf{X}_{j,nn}| == 0$ **then**
    $\mathbf{X}_{iso}^{(i)} \leftarrow \mathbf{X}_{iso}^{(i)}.append(\mathbf{x}_j)$
    break
  $outlier \leftarrow$ True
  $safe \leftarrow$ True
  for each sample $\mathbf{x}_k \in \mathbf{X}_{j,nn}$ inside radius of $\mathbf{x}_j$ **do**
    if $y_k == y_j$ **then**
      $outlier \leftarrow$ False
    else
      $safe \leftarrow$ False
  if $outlier ==$ True **then** $\mathbf{X}_{out}^{(i)} \leftarrow \mathbf{X}_{out}^{(i)}.append(\mathbf{x}_j)$
  else if $safe ==$ True **then** $\mathbf{X}_{sf}^{(i)} \leftarrow \mathbf{X}_{sf}^{(i)}.append(\mathbf{x}_j)$
  else $\mathbf{X}_{b}^{(i)} \leftarrow \mathbf{X}_{b}^{(i)}.append(\mathbf{x}_j)$

$\mathbf{X}_{SB} \leftarrow \mathbf{X}_{sf}^{(i)}$
if $|\mathbf{X}_{SB}| < \alpha|\mathbf{X}^{(i)}|$ **then** $\mathbf{X}_{SB} \leftarrow \mathbf{X}_{SB}.extend(\mathbf{X}_{b}^{(i)})$
if $|\mathbf{X}_{SB}| < \alpha|\mathbf{X}^{(i)}|$ **then** $\mathbf{X}_{SB} \leftarrow \mathbf{X}_{SB}.extend(\mathbf{X}_{iso}^{(i)})$
return $\mathbf{X}_{SB}$

Experiments

- We tested the performance of various classifiers on data augmented by SB-GAN and SB-VAE.
  - 4 typical models were used: Feed-Forward Net (FFNN), Support Vector Machines (SVM), Random Forest (RF), Logistic Regression (LR).
  - Results were 5-fold cross-validated.
- We utilized four popular imbalanced datasets (the last column denotes the imbalance ratio).

TABLE I
BENCHMARK DATASETS

Name	Samples	Dimensions	IR
Rice Type	18185	10	1:1.2
Smoke Detection	62630	14	1:2.5
Credit Card Default	30000	25	1:3.5
Surgical	14635	24	1:3.0

Model architectures

- SB-GAN:
  - Discriminator: Two fully connected layers (64/32 neurons, LeakyReLU) + Dropout(0.5) + Binary Cross Entropy loss function.
  - Generator: Two residual blocks (fully connected layers (64/32 neurons, LeakyReLU) + BatchNorm).

- SB-VAE:
  - Encoder: Two fully connected layers (64/32 neurons, LeakyReLU) + Dropout(0.5).
  - Decoder: Two fully connected layers (32/64 neurons, LeakyReLU).
  - Latent dimensionality: 8.

State-of-the-art

- Random Oversampling (ROS)
- Synthetic Minority Oversampling Technique (SMOTE)
- Borderline SMOTE (BRD-SMOTE)
- SVM-SMOTE
- k-Means SMOTE (KMN-SMOTE)
- Adaptive Synthetic Sampling (ADASYN)
- A typical GAN (same architecture as SB-GAN)
- A typical VAE (same architecture as SB-VAE)

Results (Rice Type Classification dataset)

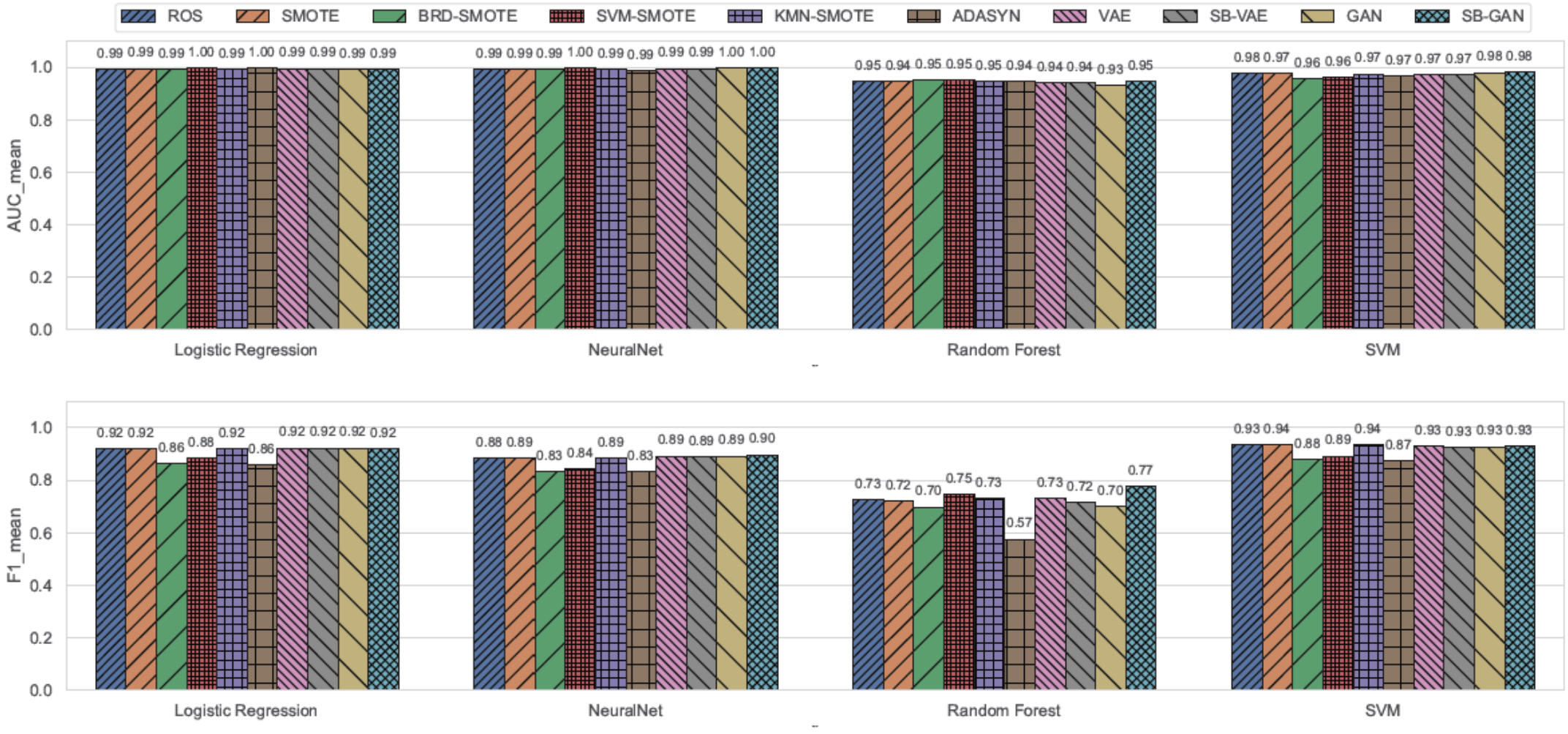


Fig. 2. Performance comparison for different classifiers and oversampling techniques for the Rice Type Classification dataset.

Results (Smoke Detection dataset)

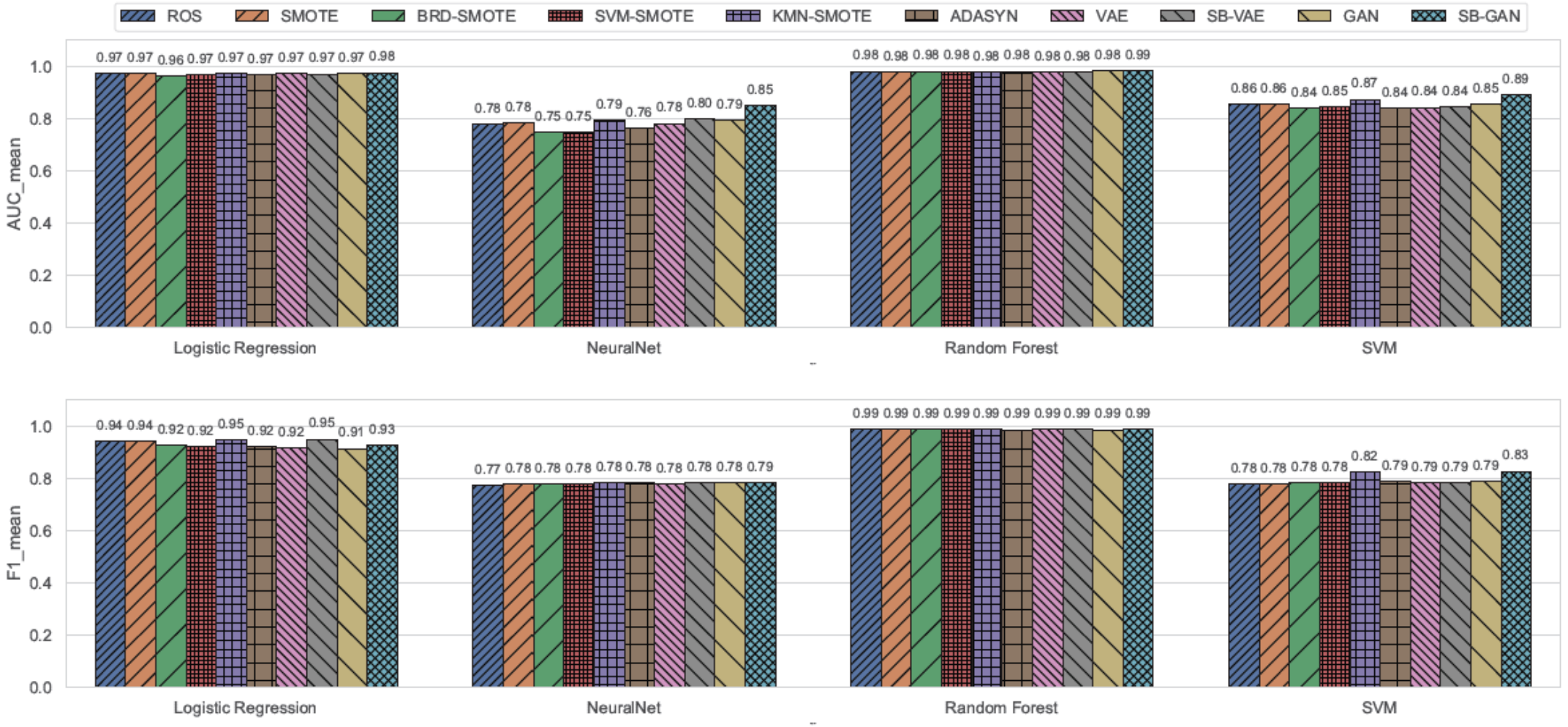


Fig. 3. Performance comparison for different classifiers and oversampling techniques for the Smoke Detection dataset.

Results (Credit Card Default dataset)

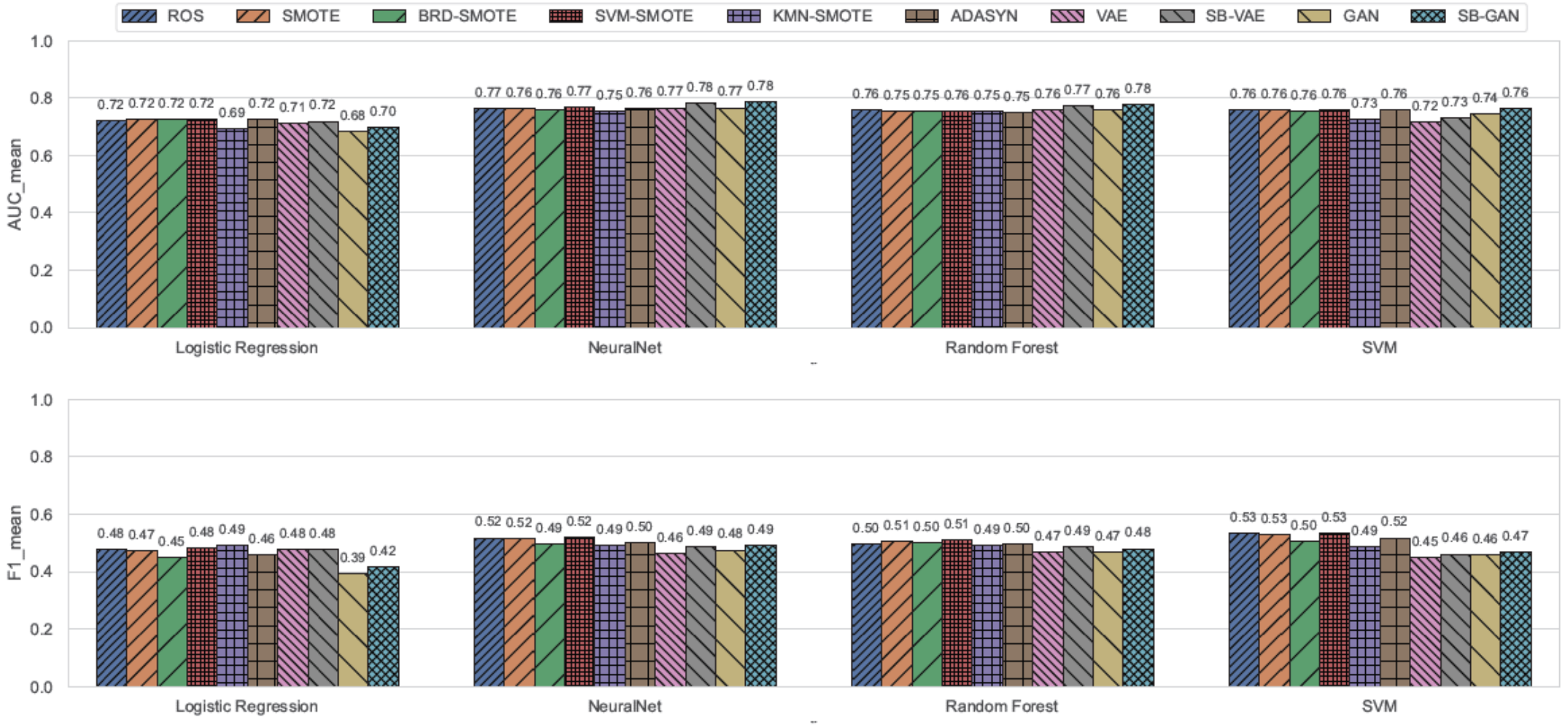


Fig. 4. Performance comparison for different classifiers and oversampling techniques for the Credit Card Default dataset.

Results (Surgical dataset)

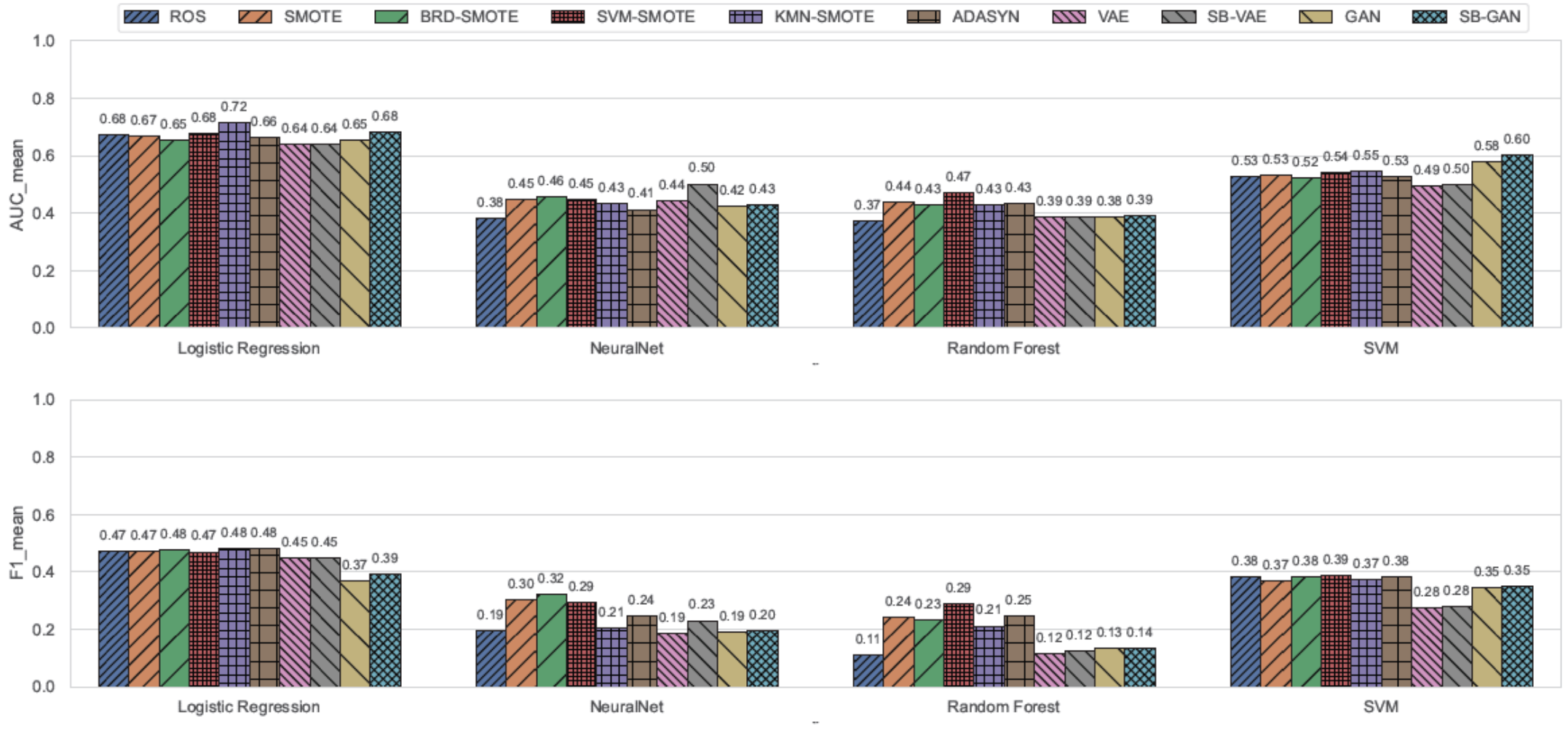


Fig. 5. Performance comparison for different classifiers and oversampling techniques for the Surgical dataset.

Conclusions

- In each dataset, we had a different classification model as a winner. However, in 3 out of 4 cases, the winners maximized their performance in combination with SB-GAN.
- SB-GAN outperformed the typical GAN model by 1% to about 8%.
- SB-VAE defeated VAE by margins ranging between 1% and 6.5%.
- Regardless of the applied oversampling technique, the performance of all classifiers drops as the imbalance ratio of a dataset increases.
- Although worse than SB-GAN, the older oversampling techniques are still very effective.

Thank you for watching

Thank you!

e-mail: lakritidis@ihu.gr

Github repository: https://github.com/lakritidis/imbalanced