A Comparative Study Of The Wasserstein Distance Generative Adversarial Network And SMOTE Density-based Over-sampling Approaches In Addressing Class Imbalance

---

Kwanda Sydwell Ngwenduna

Supervisor:
Mr. Rendani Mbuvha

A research report submitted in partial fulfillment of the requirements for the degree of Master of Science in the field of e-Science

in the

School of Computer Science and Applied Mathematics
University of the Witwatersrand, Johannesburg

29 May 2020

Declaration

I, Kwanda Sydwell Ngwenduna, declare that this research report is my own, unaided work. It is being submitted for the degree of Master of Science in the field of e-Science at the University of the Witwatersrand, Johannesburg. It has not been submitted for any degree or examination at any other university. Where the discussion has been informed by previously-submitted work, this has been indicated as such. All material from external sources has been appropriately referenced.

Kwanda Sydwell Ngwenduna

29 May 2020

"…in the process of training generative models, we will endow the computer with the understanding of the world and what is made up of."

OpenAI

Abstract

In binary classification problems, class imbalance occurs if one of the classes has overwhelmingly more instances than others. This causes a significant bias in the accuracy of Machine Learning (ML) classifiers. A pioneering and popular approach to alleviate class imbalance is the Synthetic Minority Over-sampling TEchnique (SMOTE). However, SMOTE is less reliant on the true underlying probability distribution of the minority class data. Probability density estimation approaches have recently been adopted, but most of these postulate the unknown probability distribution of the minority class, which can be subjective and inappropriate. Generative Adversarial Networks (GANs) can sample from the true underlying probability distribution without explicitly specifying its form. GANs have been used to create realistic samples and outperforms other deep generative models. However, there has been limited theoretical and empirical reviews comparing generative models such as GANs and other SMOTE density-based approaches for alleviating class imbalance, especially for tabular data sets akin to most financial institutions.

This report compares Wasserstein Conditional GAN with gradient penalty (WCGAN-GP) to density-based SMOTE approaches for synthetic minority sample generation on a number of imbalanced data sets. A Logistic Regression (LR) model is trained to detect minority cases on the imbalanced and over-sampled data sets, compared using Precision, Recall, F1-Score and the Receiver Operating Characteristic (ROC) curve on a testing data set. On average, WCGAN-GP yields better results, followed by SMOTE, with RWO and PDFOS having the worst performance than the Baseline. WCGAN-GP shows a statistically superior predictive performance over SMOTE density estimation techniques on 4 of the 5 data sets used. These results show a significant potential for GANs as an alternative to SMOTE density techniques, useful for new sample creation, data augmentation and boosting classification models.

Keywords: Class imbalance, GAN, PDFOS, RWO, SMOTE, WCGAN-GP.

Acknowledgements

I sincerely express my heart-felt gratitude to Mr. Rendani Mbuvha for his supervision, invaluable experience and assisting me in this work. It was marvelous to have such a high calibre individual with vested interest and willingness to see this work throughout. Mr. Rendani Mbuvha continues to be my inspiration and significantly motivates me in pursuing research and my interests.

This work would not have been possible without the financial support and guidance of DST-CSIR National e-Science Postgraduate Teaching and Training Platform. In the same wavelength, I remain forever grateful to God's bountiful mercies and to my beloved parents for inspiring and believing in me in all my pursuits and successes.

Contents

List of Figures

List of Tables

List of Abbreviations

Adadelta	Adaptive delta
AdaGrad	Adaptive gradient
Adam	Adaptive Moment estimation
ADASYN	ADAptive SYNthetic sampling
AE	Autoencoder
AHC	Agglomerative Hierarchical Clustering
ANN	Artificial Neural Network
AUC	Area Under the ROC Curve
AUPRC	Area Under the Precision-Recall Curve
BAGAN	Balanced Generative Adversarial Network
Bagging	Bootstrapped Aggregating
BE	Best Estimate
BEGAN	Boundary Equilibrium Generative Adversarial Network
BGD	Batch Gradient Descent
BN	Bayesian Network
cGAN	Conditional Generative Adversarial Network
CLT	Central Limit Theorem
CV	Cross Validation
D	Discriminator
DAE	Denoising Autoencoder
DEAGO	Denoising Autoencoder Generative Over-sampling
DBN	Deep Belief Network
DBSCAN	Density-Based Spatial Clustering of Applications Noise
DBSMOTE	Density-Based SMOTE
DCGAN	Deep Convolutional GAN
DRAGAN	Deep Regret Analytic GAN
DT	Decision Tree
EM	Earth mover
FN	False Negative
FP	False Positive
FVBN	Fully Visible Belief Network
G	Generator
GD	Gradient Descent
GAMO	Generative Adversarial Minority Over-sampling
GAN	Generative Adversarial Network

GBM	Gradient Boosting Machine
GMMN	Generative Moment Matching Network
GP	Gradient Penalty
GSN	Generative Stochastic Network
ICA	Independent Component Analysis
InfoGAN	Informative GAN
IPM	Integral Probability Metric
IR	Imbalanced Ratio
ISOMAP	Isometric Mapping
JS	Jensen-Shannon
KDE	Kernel Density Estimate
K-NN	K-Nearest Neighbor
KL	Kullback-Leibler
LLE	Locally Linear Embedding
LR	Logistic Regression
LS-GAN	Least-Squares GAN
LSGAN	Loss Sensitive GAN
MCC	Mattthew's correlation coefficient
medGAN	Medical GAN
MCMC	Markov chain Monte Carlo
MISE	Mean Integrated Squared Error
ML	Machine Learning
MLE	Maximum Likelihood Estimator
MLP	Multi-Layer Perceptron
MMD	Maximum Mean Discrepancy
MM-GAN	MiniMax GAN
MWMOTE	Majority Weighted Minority Over-sampling Technique
NAdam	Nesterov Adaptive moment estimation
NAG	Nesterov Adaptive Gradient
NN	Nearest Neighbor
NS-GAN	Non-Saturating GAN
PAUC	Partial Area under the Curve
PReLU	Parametric Rectified Linear Unit
PCA	Principal Component Analysis
PMCM	Pairwise Multiple Comparison of Mean Ranks Package
PDF	Probability Density Function
PDFOS	Probability Density Function estimation based Over-Sampling
PROGAN	Progressive Growing GAN
PW	Parzian-window
RACOG	Rapidly Converging Gibbs
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit

RGAN	Relativistic GAN
RMSprop	Root Mean Square propagation
ROC	Receiver Operating Characteristic
ROS	Random Over-Sampling
RWO	Random Walk Over-sampling
SGD	Stochastic Gradient Descent
SMOTE	Synthetic Minority Over-sampling TEchnique
SNE	Stochastic Neighbor Embedding
SOMO	Self Organizing Map Over-sampling
tanh	Hyperbolic tangent function
TN	True Negative
TP	True Positive
t-SNE	t-distributed Stochastic Neighbor Embedding
VAE	Variational Autoencoder
WGAN	Wasserstein GAN
WGAN-GP	Wasserstein GAN with Gradient Penalty
WCGAN	Wasserstein Conditional GAN

List of Symbols

α	Random number extracted from the Uniform distribution $[0, 1]$
ε	A small value which avoids division by zero in a GD optimiser
σ	Standard deviation of an attribute in the training data
μ	Mean of the minority class data
γ	Momentum term in a GD optimiser
η	Learning rate
λ	Regularization parameter
θ_i	Weight of an ANN for feature i
ψ	Standard Normal distribution with zero mean and unit variance
c	Clipping range done in WGAN
d	Number of features in the original data set
D	Discriminator with parameter θ_d
$E(x)$	Energy function for a Boltzmann machine
f	A Lipschits function
$g(.)$	Activation function
G	Generator with parameter θ_g
h	Smoothing parameter (bandwidth)
$h_\theta(x^{(i)})$	Predicted target for sample i
$J(\theta)$	Value function to be optimised in the GAN architecture
$\frac{\partial J(\theta_{t,i})}{\partial \theta_{t,i}}$	Gradient of $J(\theta)$ with respect to θ in an ANN at time step t
k	Number of nearest neighbors in SMOTE
K	Non-negative differentiable kernel function
m	Number of minority class cases
$\hat{m}_t$	Bias-corrected estimates of the first moment m_t of $\frac{\partial J(\theta)_{t,i}}{\partial \theta_{t,i}}$
N	Number of examples in the data set
$N^d(0, 1)$	multivariate d-Gaussian distribution with mean 0 and variance 1
p	Probability distribution
p_g	The generator's data distribution over x
$p_{data}(x)$	Unknown PDF generating the feature vector X
$p_z(z)$	A prior PDF generating latent representations z
r	Random number extracted from the standard Gaussian distribution $N(0, 1)$
R	An upper-triangular matrix of U i.e. Cholesky decomposition
s	Synthetic data instance generated using SMOTE

T	Number of synthetic fraud cases to over-sample
U	A positive-definite unbiased co-variance matrix of the minority class
$\hat{v}_t$	Bias-corrected estimates of the second moment v_t of $\frac{\partial J(\theta_{t,i})}{\partial \theta_{t,i}}$
$\hat{w}_i(j)$	Minority class instance generated using RWO
W_i	An instance representing a minority class for attribute i
x	Data point
X	Original feature vector
Y	Actual target value
z	Latent noise i.e. random noise
Z	Latent noise space

1 Introduction

This chapter introduces the problem of class imbalance, gives the research aims and objectives of the research, research questions and an outline of the research report.

1.1 Background

Whilst Machine Learning (ML) has gained wide applications, class imbalance remains an ubiquitous problem [72]. In binary classification problems, this occurs when one of the classes has overwhelmingly more instances than others. ML classifiers tend to have skewed accuracy towards the majority class when the data is imbalanced [104]. This results in giving a false sense of performance, thereby underestimating the most important class of interest (usually the minority class containing far fewer instances) [58]. This is problematic as misclassifying a minority class can result in significant misclassification costs than for the majority case [104].

Techniques exist to alleviate class imbalance and these techniques include re-sampling, algorithmic-level, cost-sensitive learning, ensembles and generative modelling [41, 49, 58, 72]. Re-sampling techniques modify the training data such that the distribution of the classes is evenly balanced where the majority or minority class is either under-sampled or over-sampled. Over-sampling has been the most frequently used technique than under-sampling since under-sampling eliminates important information in the majority class [72]. A pioneering and popular method to alleviate class imbalance has been the Synthetic Minority Over-sampling TEchnique (SMOTE) method [27]. However, SMOTE suffers from over-fitting, over-lapping classes, noisy examples, is less reliant on the true probability distribution, alters the original distribution of the minority classes and this may not be desirable [38, 49, 73, 179].

Techniques exist to overcome these issues, however, there is no consensus on which technique is superior and which one is appropriate in a given domain setting

[49]. There have also been few empirical studies which compare, evaluate and synthesize SMOTE and its variants. There have been few approaches which sample explicitly from the minority class distribution using density-based approaches, without postulating a probability distribution of the minority class. Current density-based approaches may be subjective as they need to specify in advance the format and structure of the minority class distribution. Generative models offer a significant alternative yet these models have not been thoroughly explored in imbalanced learning.

Implicit generative models can approximate the true underlying minority class distribution and we can sample from this probability distribution without knowing its form in advance. The idea is to not explicitly model density but instead just sample to generate new instances. We want to sample from a complex probability distribution but we cannot do this directly. However, we can sample from something simple (noise) and then learn a transformation to the training distribution using Generative Adversarial Network (GAN) [68]. GAN provides an alternative model-based approach to create synthetic examples in a single pass. A GAN is a combination of neural networks [149] that generate synthetic data given certain input data [66].

GANs have been highly successful in anime character creation [88], video generation [168], images [21, 90, 184], celebrity faces [6], super resolutions [180], text to images [145], music generation [173], learning joint distributions and imputing missing data [176]. This technique can generate seemingly natural and new data samples using the true probability distribution of the data [50]. GANs are notable better than other generative models due to the quality of samples they generate [67, 79]. A popular GAN, called the Wasserstein GAN (WGAN) [7] is evaluated against recent SMOTE density-based approaches, for over-sampling the minority cases observed in a number of imbalanced data sets.

1.2 Research Aims and Objectives

The aim of this work is to accurately over-sample the minority class of interest in a number of public available data sets using an implicit generative modelling approach and then boost a Logistic Regression (LR) model for binary classification [116]. This work covers the following objectives:

- Review over-sampling approaches for alleviating class imbalance and categorize the approaches according to their taxonomies, including generative modelling;
- Train a LR on 5 imbalanced data sets from the Machine Learning Repository;
- Use SMOTE [27], Random Walk Over-sampling (RWO) [179] and PDF estimation based Over-Sampling (PDFOS) [59] to over-sample the minority cases in each data set; and
- Through the Precision, Recall, F1-Score, Area under the Receiver Operating Characteristic (ROC) curve (AUC) [71] and Area under the Precision-Recall curve (AUPRC), compare these techniques against the Conditional WGAN with gradient penalty (WCGAN-GP) [7, 69, 118].

1.3 Research Questions

This work addresses these research questions:

1. Can we improve the model yielded by LR to detect each minority case and address class imbalance using SMOTE?
2. Do SMOTE density estimation variants outperform traditional SMOTE in improving LR applied on the various experimental data sets?
3. Can GANs outperform the best performing SMOTE density estimation variant in improving LR applied on the various data sets?
4. Does the GAN provide a statistically significant performance over SMOTE, RWO and PDFOS?

1.4 Outline

The rest of the report is organised as follows. Chapter 2 reviews the literature on class imbalance. Chapter 3 describes how neural networks work, while Chapter 4 describes GANs. Then Chapter 5 describes SMOTE density estimation approaches. Chapter 6 outlines the experiments conducted. Chapter 7 presents the results and discusses them, while Chapter 8 gives conclusions, limitations and possible future work.

2 Literature Review

This chapter reviews the literature on imbalance learning, paying attention on synthetic over-sampling solutions and generative models.

2.1 Class Imbalance

2.1.1 Definition

Whilst ML has gained significant prevalence in the past few decades, class imbalance remains a pervasive problem [72]. Class imbalance occurs in a supervised classification problem when there is an unequal distribution observed in the target class of interest by a large margin [27, 49, 72]. Class imbalance occurs due to nature of the data space, data collection costs and limitations and absolute rarity [58, 72]. This research is concerned with binary classification problems as these have been the most studied.

Imbalanced data sets typically have accuracy bias towards the majority class when ML classifiers are trained and tested on them [27, 26, 72, 85]. This arises because ML classifiers are designed to improve the accuracy by reducing the misclassification error. Thus they do not necessarily take into account unequal class distributions. This problem causes a significant and an unexpected performance behaviour for most classifiers [104], creating the class imbalance issue.

2.1.2 Solutions

A number of solutions shown in Figure 2.1 [41, 48, 58, 72] exist dealing with this problem, the most common and influential being sampling approaches. Most studies have shown that ML classifiers show good accuracy for data sets that are balanced compared to those with class imbalance [12, 27]. This chapter reviews sampling approaches in detail, more especially synthetic over-sampling, as this is the

most influential and popular solution to alleviate class imbalance.

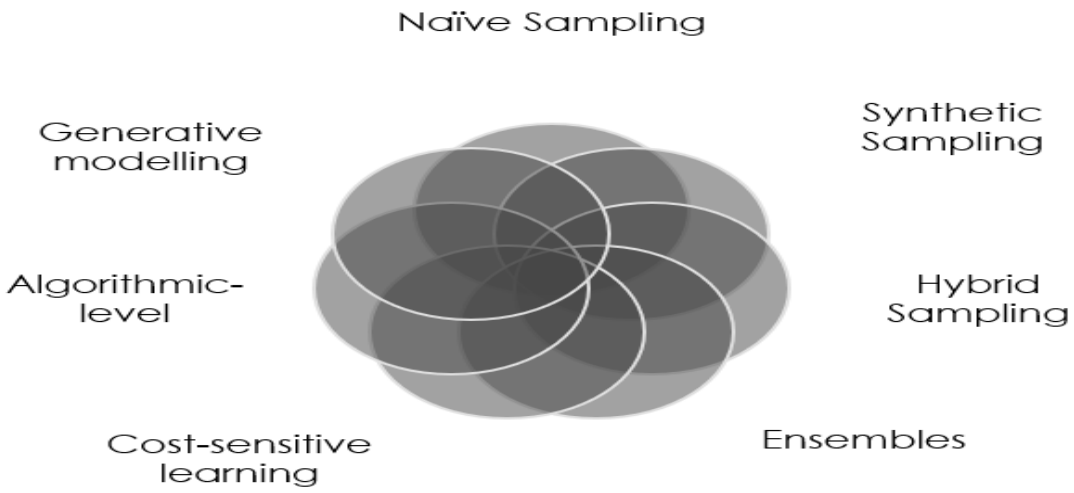


FIGURE 2.1: Taxonomy of techniques to alleviate class imbalance

Under-Sampling approaches discard useful information, reduce the amount of data set available for training and increase the variance of a classifier [12, 49, 72, 85, 86]. Essentially, generative modelling is argued to offer a possible different alternative to synthetic over-sampling techniques based on density estimation.

2.2 Random Over-sampling

Random Over-Sampling (ROS) randomly replicates minority class instances until the data set is balanced. ROS creates extra information so that no information is lost from the original data set [27]. However, this does not add any new information. This process induces over-fitting, thereby creating longer training and model complexity [72].

Fernández et al. [49] argue that ROS induces a higher weight and cost to the majority class instances, causing possible further model bias. Thus it might still be hard to classify and model correctly the clusters of minority cases using a classifier, especially in the case of over-lapping or even small disjuncts [12]. An informed over-sampling approach called synthetic over-sampling, improves ROS and has been the most studied and popular approach.

2.3 Synthetic Over-sampling

Chawla et al. [27] introduce a pioneering method in class imbalance popularised as SMOTE. This section reviews SMOTE and its variants.

2.3.1 SMOTE

SMOTE creates new minority cases by linearly interpolating between two nearest neighbour (NN) instances of the minority class [27]. Chawla et al. [27] demonstrate that SMOTE significantly improves the effectiveness of ML binary classifiers compared to ROS and Under-Sampling approaches. Over time, this technique has proved to be popular with researchers, becoming a pioneer in imbalanced learning.

However, SMOTE treats all minority class instances equally and does not consider the entire data space as a whole [41]. Moreover, this technique may also get confused between minority and majority instances if there are over-laps, creating noisy examples and further possible model bias. This technique also neglects the probability density function (PDF) of the overall minority class [38] whilst altering the original PDF [179]. As a result, SMOTE is notable known to over-generalise and over-fit.

Over the past decades, SMOTE variants have been proposed to alleviate some of these problems. These techniques can be divided into those that focus on synthetic data generation using clustering techniques, density estimation, feature extraction, ensembles, filtering approaches, kernel functions and more recently generative modelling [41, 49, 50, 58, 72]. This research is not able to list all of these techniques, however we review the most popular ones.

2.3.2 Borderline methods

Usually, the best candidates to be over-sampled are determined before generating synthetic examples using some heuristic.

Borderline-SMOTE [70] focuses on instances near the decision boundary and these are deemed more important. ADAptive SYNthetic sampling (ADASYN) [73] infers which points are more difficult to pick up and attempts to place a higher ratio of synthetic data close to these points. Safe-level-SMOTE [23] generates a safe level for each data point before over-sampling synthetic examples.

The above three techniques are some of the most popular and common SMOTE variants and baseline approaches to compare against newer alternatives. However, these approaches do not sample from the true underlying minority class distribution when generating new synthetic cases and they may alter the original distribution of the minority class. Density-based approaches account for the true global structure of the minority classes and may solve some of the issues above [38, 59, 179].

2.3.3 Clustering methods

These approaches adopt clustering in the minority class and over-sample following the centroids of the clusters, based on the notion that SMOTE does not take the cluster structure into account [10, 11, 42, 61, 126]. There are a number of these approaches which have been shown to be better than SMOTE.

Agglomerative Hierarchical Clustering (AHC) [63] was the first attempt adopting clustering where AHC is used to over-sample minority cases inside clusters. Other clustering-based approaches are reported such as Majority Weighted Minority Over-sampling Technique (MWMOTE) [11] and Density-Based SMOTE (DBSMOTE) [22], among many others. Clustering techniques also make assumptions around the number of centroids, density, distribution and connectivity. These add further assumptions and these models may fail to generate enough clusters especially with too few minority instances.

2.3.4 Feature extraction

Other algorithms adopt feature extraction before applying over-sampling in the reduced dimension. These approaches are based on the preservation of global distances or neighborhood relations between samples in the higher and lower dimension, where SMOTE is performed in the reduced space. These algorithms are particularly useful for inseparable and over-lapping classes and complex data sets [172]. The most common algorithm is t-distributed Stochastic Neighbor Embedding (t-SNE) [108, 172] although Principal Component Analysis (PCA) [83], Autoencoder (AE) [15] and Self-Organising Maps Over-sampling (SOMO) [61] also accomplish this.

t-SNE reveals the underlying probability distribution structure at different scales and preserving the global data structure better than other methods [108, 109, 172]. However, for data sets with too few variables or fewer samples, t-SNE and other clustering techniques may not be inappropriate [49, 178]. These techniques may also be time-consuming and complex and often are based on distance or neighborhood assumptions.

2.3.5 Probability distribution methods

There are some methods adopting a density-based approach for the minority class and then over-sample from the estimated probability distribution. This section reviews these methods.

2.3.5.1 Random Walk Sampling

Zhang and Li [179] present Random Walk Over-sampling (RWO) which generates synthetic minority examples by trying to preserve the variance and mean of the minority class. Zhang and Li [179] compare RWO, SMOTE and ROS and note that RWO was statistically better. However, RWO assumes the mean and variance of each attribute, which exists only for continuous variables and thus this technique may be arbitrary for non-continuous variables, despite its simplicity and ease of use.

2.3.5.2 Kernel functions

Gao et al. [59] estimate the probability density function (PDF) of the minority class using the Parzian-window (PW) kernel function, termed PDF estimation based Over-sampling (PDFOS). PDFOS uses Normal kernel functions to locally approximate the PDF of the minority class. PDFOS has good theoretical components, enabling its practical and good results [35].

However, this approach is notable complex on its mathematical properties and the Gaussian assumption may not always hold for some data sets. Mathew et al. [114] consider a similar PDFOS algorithm which incorporate ADASYN, called Kernel ADASYN. This approach focuses on the most important and difficult to classify points by using a PDFOS around these points.

2.3.5.3 RACOG

Das, Krishnan, and Cook [38] generate synthetic minority examples through a Gibbs sampler using the RApidly COnverging Gibbs algorithm (RACOG). Gibbs sampling obtains samples from a multivariate PDF when sampling is impossible through Markov chain Monte Carlo (MCMC) [62, 117]. Unlike Monte Carlo sampling methods that are able to draw independent samples from a PDF, MCMC methods draw samples where the next sample is highly dependent on the existing sample, called a Markov Chain [62, 117]. RACOG uses the Gibbs sampler to sample from a specified PDF. Chow-Liu's Algorithm [32] approximates the multivariate PDF of the data through second-order product approximations. Then the Gibbs sampler finds a stationary PDF for a sequence of sampled observations.

RACOG depends on two important parameters to find this stationary distribution: burn-in rate and the lag, computed using a convergence diagnostic test called the Raftery-Lewis test [141]. Thus RACOG has slow convergence and dependence between previous values, due to the requirement to build a Markov chain for each minority example.

2.3.5.4 DEAGO

Bellinger, Japkowicz, and Drummond [16] use a Denoising Autoencoder (DAE) [166, 167] for modeling a joint multivariate PDF of the minority class and then synthetically creating samples from the learned distribution. This algorithm is called DEnoising Autoencoder-based Generative Oversampling (DEAGO) and it can approximate the global structure of the minority distribution. Bellinger, Japkowicz, and Drummond [16] show that DEAGO outperforms SMOTE and SMOTE boosting techniques. However, this was not contrasted with other density estimation approaches.

2.3.5.5 Other Methods

There are other methods capable for estimating a PDF and sampling from it. Bayesian Networks (BNs) [9, 56, 150], copulas [162, 134] and generative models [67] may also be adopted. Copulas can be used to find a multivariate PDF for uncorrelated variables and then sample from that PDF. However, BNs and copulas are limited by

the PDF type, computational issues and strict assumptions, limiting their sample generation capacity.

Generative models have received a lot of interest in the last few years [67]. Generative models use training observations p_x to learn the model p_{model} that accurately mimics the observations produced by the model. Given the vast literature on generative models, we review them Section 2.5.

2.3.6 Ensembles

Another approach is the use of ensembles where a classifier's accuracy is increased by the use of training on different over-sampled data sets or different algorithms and combining outputs to a single outcome [105]. SMOTE has also been extended to include boosting and boostrapped aggregating (bagging). These approaches tend to improve the results of SMOTE. However, they can take a long time to compute and still do not solve the true data distribution issue.

2.4 Hybrid Methods

Other techniques include combining SMOTE with data cleaning techniques, informed under-sampling techniques [12] or greedy-filtering approaches [4, 87, 143]. These hybrid approaches can also eliminate redundant and noisy instances, thereby further improve accuracy.

2.5 Deep Generative Models

Generative models have received lots of interests from researchers for creating new samples. This section covers a taxonomy of these techniques, with a particular focus on GANs.

2.5.1 Definition

Given a data set with observations X, we assume that X has been generated from an unknown PDF p_{data}. A generative model p_{model} mimics p_{data} as close as possible. If this is achieved, then we can sample from p_{model} to generate realistic samples that

appear to have been drawn from p_{data}. We are satisfied if our model can also generate diverse samples that are suitable different from X. In some cases, the model can be estimated explicitly and sometimes it can generate samples implicitly. Other models are capable of doing both. GANs provide no estimate of the model but are capable of generating new data without knowing it. Goodfellow [66] provides a taxonomy of common deep generative models show in Figure 2.2, divided into implicit and explicit models. GANs are designed to remedy most of the disadvantages that come with explicit models and other Markov chain models.

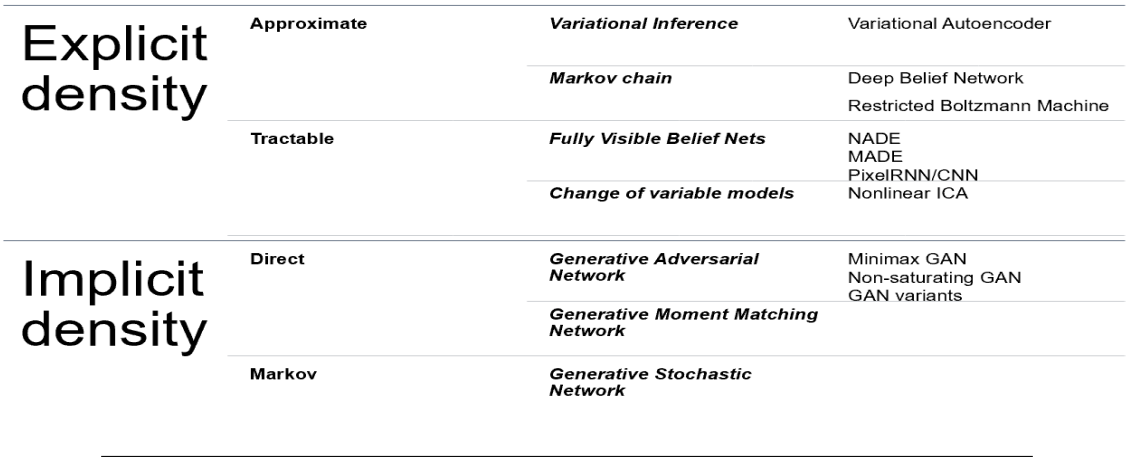


FIGURE 2.2: Taxonomy of generative models

2.5.2 Explicit models

Explicit models specify or approximate a parameterised log-likelihood representation of the data [68]. Parameters are then estimated and learned from the data and this requires a maximum likelihood estimation which integrates over the entire data space and this may be intractable [98]. These approximation techniques may not always yield the best results as some of them rely on Markov chains which are time-consuming [68].

Two popular tractable models are fully visible belief networks (FVBNs) [53] and nonlinear independent component analysis (ICA). Approximate methods improve on the design of tractable models which can be computational intensive and limited [68, 110, 146]. Approximate methods use either deterministic i.e. variational

inference or stochastic approximations i.e. MCMC approaches. Variational inference involves the use of Variational Autoencoders (VAEs) [92, 146] to approximate $p_{model}(x)$ using lower bounds.

2.5.2.1 FVBNs

FVBN estimates the PDF of the training data $p_{model}(x)$ into a decomposed product of one-dimensional probability distributions. This model outputs a probability for each possible value if x is discrete and outputs a network of parameters of a simple distribution if x is continuous. Using the generated model, sampling is done one step at a time, conditioned on all previous steps [68].

The problem with these models is their computational complexities as they need to generate one point at a time. Other problems include poor learning representations, over-emphasizing details over global data and not closely reflecting the true generation process [66]. Moreover, these models have been more useful for image synthesis than structured data sets such as tabular data [131]. GANs are known to provide new samples in parallel, thus yielding greater speed of generation [66, 98].

2.5.2.2 Non-linear ICA

Non-linear ICA involves defining some continuous non-linear transformations of data between high dimensions and lower dimensional spaces. The distribution of the data p_{model} is transformed into a distribution of a latent space z defined by $p_z(g)$ where g is some tractable transformed version of p_z. The challenge in ICA is finding tractable distributions in the latent space and these are limited [67]. GANs are known to have fewer restrictions than these models [17, 66, 68].

2.5.2.3 Variational Autoencoders

VAEs, along with FVBNs and GANs, are three of the most popular approaches for sample generation. VAEs are an extension to AEs [15, 95, 146]. AE learns useful representations of the data by encoding X into a compressed latent space z using $q(z|x)$ and then decoding z back into X using $p(x|z)$ by minimising the reconstruction error between the original data and the deconstructed data [15]. VAE maximizes the

following function :

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} \left[\log p(x|z) + \log p(z) - \log q(z) \right] \tag{2.1}$$

Unlike auto-regressive models, VAEs are normally easy to run in parallel during training and inference [67, 95, 146]. Conversely, they are normally harder to optimize than auto-regressive models [67, 110]. The encoder converts the input to latent space representations through the mean and variance and samples can be created from the learned representation. VAEs have been criticised to be generating blurry samples and are intractable [67, 153].

2.5.2.4 Boltzmann Machines

Boltzmann machines rely on the use of Markov chains to model $p_{model}(x)$ and to sample from it [1, 75, 152]. A Markov chain is a process that is used to generate samples by repeatedly drawing a sample from a transition operator [62]. A Boltzmann machine is an energy-based function defined as:

$$p_{model}(x) = \exp\left(-E(x)\right) / Z \tag{2.2}$$

where $E(x)$ is an energy function and Z is a normalizing factor to ensure that $p_{model}(x)$ sums to one [1, 67].

These methods include Restricted Boltzmann machine (RBM) [1] and Deep Belief Networks (DBNs) [76, 77]. DBNs and RBMs are generative stochastic neural networks that can estimate a PDF [1]. Samples are obtained through MCMC runs to convergence and this can be very expensive to run [98]. These models were pioneers in early 2006 for deep generative models but they have been rarely used because of poor scale-ability for higher dimension problems and high computational costs [67].

2.5.3 Implicit models

Implicit models learn to model the true distribution and define a stochastic procedure to directly generate new data from a latent space. These models can be trained indirectly without needing an explicit density function to be learned or defined.

Some of these models such as the Generative Stochastic Network (GSN) [17] involve MCMC methods which impose greater computational cost and often fail to scale to higher dimensional spaces [67]. Generative Adversarial Networks (GANs) [68] and Generative Moment Matching Networks (GMMNs) [98] are one of the few implicit probabilistic models capable of sampling in parallel and in a single step.

2.5.3.1 GANs

GANs were originally invented in a landmark paper by Ian Goodfellow in 2014 [68]. The setup of the framework uses an adversarial process to estimate the parameters of two artificial neural network (ANN) [149] models by iteratively and concomitantly training a discriminator (D) and a generator (G), as shown in Figure 2.3.

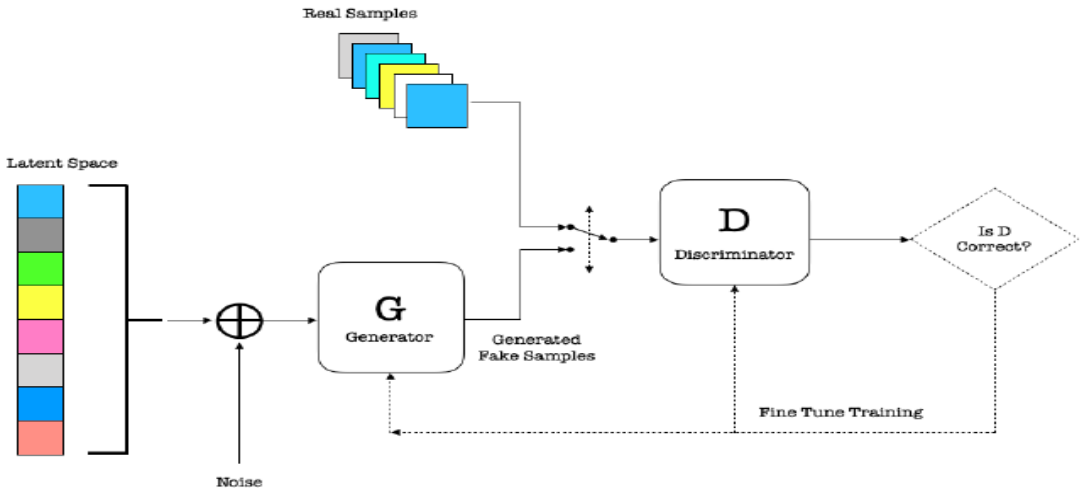


FIGURE 2.3: GAN operation

Through multiple cycles of generation and discrimination, both networks train each other, while simultaneously trying to outwit each other [68, 113, 130, 184]. GANs have two adversarial ANNs:

- G picks z from the prior latent space Z and then generates samples from this distribution using ANN;
- D receives generated samples from G and the true data examples, and must distinguish between the two for authenticity.

Both D and G are ANNs which play a zero-sum game, where G learns to produce realistic-looking samples and D learns to get better at discriminating between the generated samples and the true data. Once G is trained to optimality, it can create new samples and augment the training data set. GANs can sample in parallel better than other generative models, have fewer restrictions on the generator function, assume no use of Markov Chains, no variational bounds unlike VAE and produce subjectively better quality samples than other generative models [7, 66, 67, 68, 140, 153].

Whilst GANs are gaining popularity in many applications, they have notable issues. GANs are notoriously difficult to train properly, difficult to evaluate, the likelihood cannot be easily be computed, suffer from the vanishing gradient problem, mode collapse, boundary distortion and over-fitting [7, 67, 153].

Mode collapse is when many latent noise values z are mapped to the same data point x, leading to a lack of diversity in the samples that are created i.e. underfitting. The vanishing gradient problem occurs when D becomes perfect in its training without giving G the chance to improve. As a result, GANs may fail to converge and thereby leading to poor generated samples [7]. Figure 2.4 provides a non-exhaustive taxonomy of GAN variants and improved training, including common examples [36, 79, 82, 170].

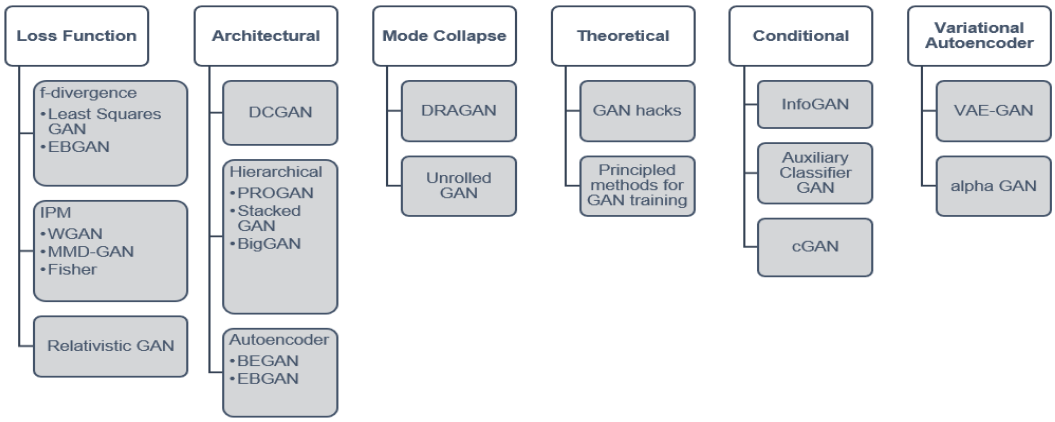


FIGURE 2.4: Taxonomy of GAN variants

Salimans et al. [153] look at ways to improve GANs (called hacks) while other authors propose variants to the vanilla GAN by changing the cost function, adding

gradient penalties (GPs), adding labels, avoiding over-fitting and finding better ways of optimising GANs. The first extension of GAN was the Conditional GAN (cGAN) which gave the generator the label in the latent space, making them class conditional [118]. Until the introduction of Deep Convolutional GAN (DCGAN) [140], training GANs was very unstable. DCGANs provide some further tricks using convolutional and deconvolutional layers. Since then, more variants and heuristics were proposed.

Wasserstein GAN (WGAN) [7] proposes a different loss function, becoming the most studied and widely used GAN architecture ever since [82]. WGAN has been shown to give better quality of generated synthetic data than the vanilla GAN and alleviating most of the GAN issues [7]. Gulrajani et al. [69] further amends WGAN through an addition of a GP to the cost function, coming with WGAN-GP.

In recent years, other loss functions which unify the GAN loss framework including f-divergence [129], Integral Probability Metrics (IPMs) [82] and Relativistic GANs (RGANs) [89], were proposed. The f-divergence measures the difference between p_{data} and p_g with a specific convex function f [129]. f-divergence GAN, IPMs and RGAN are considered unified frameworks suitable for stronger generalization to other loss-variants [111]. WGAN is a special class of IPMs and the most studied GAN. Other popular GAN loss variants include Least Squares GAN (LSGAN) [112], Boundary Equilibrium GAN (BEGAN) [19], Loss Sensitive GAN (LS-GAN) [138] and Energy-Based GAN (EBGAN) [182].

Other advanced GANs include the revolutionary Progressive Growing GAN (PROGAN) [90] which proposes a progressive growing and steps towards GAN performance. Other variants include Self-attention GAN [180] and BigGAN [21] which achieved tremendous performances on Imagenet data sets. There have been hybrids of GANs and VAEs where VAEs are used to encode the latent space to come up with VAE-GAN [95]. For further GAN reviews, Creswell et al. [36], Hitawala [79] and Hong et al. [82] provide a comparative overview. Lucic et al. [106] conduct an in-depth study on GANs and note no significant performance differences on the GANs studied.

Fiore et al. [50] compare GAN and ordinary SMOTE on a credit card fraud detection problem. Douzas and Bacao [41] apply cGAN [118] to improve the classification effectiveness on 12 data sets. Zheng et al. [183] compare GANs and deep learning methods for fraud detection in two large banks in China. Mariani et al. [113]

propose Balanced GAN (BAGAN) to restore imbalance on image data sets. Mullick, Datta, and Das [124] propose Generative Adversarial Minority Over-sampling (GAMO) for imbalanced image data sets, compared against SMOTE and common GAN architectures. GAMO outperformed the other methods on image data sets.

Armanious et al. [8] propose the learning of multi-label discrete variables for electronic health records using medical GAN (medGAN). This study compared medGAN, VAE, stacked RBMs and vanilla GANs and noted a significant improvement on samples generated. Mottini, Lheritier, and Acuna-Agost [122] generate new airline passenger name records using GANs. Aviñó, Ruffini, and Gavaldà [9], Camino, Hammerschmidt, and State [25] and Che et al. [28] generate multi-label discrete data for healthcare records in various settings.

However, these studies were not compared with other SMOTE density estimation algorithms nor with other common ML classifiers. Even though there is improvement over SMOTE in these studies, it is not clear if GAN can outperform SMOTE architectures which focus on density estimation. Whilst there is promising work for GANs in handling class imbalance in other domains, this work has not been thoroughly explored with other SMOTE density-based approaches nor with other generative models. This research determines whether GANs may provide a superior alternative compared to SMOTE density-based approaches.

2.5.3.2 GMMNs

GMMNs minimize the maximum mean discrepancy (MMD) between the moments of p_{data} and p_{model} and are known to be simpler than other generative models [98]. Moment matching evaluates whether the moments of the true distribution $p_{true}(x)$ match those of the data $p_{data}(x)$ through MMD. This approach is similar to GANs in terms of training except using a different loss function which leads to faster sampling. However, GMMNs have received less attention than GANs and VAEs, limiting their sample generative scheme [7, 67, 79].

2.5.4 Summary

There are a number of deep generative models for synthetic sample generation. Some of the models are explicit with an intractable likelihood and inference. Some models are only approximate and generate blurry samples. Other methods do

not sample in parallel, are complex and rely on Markov chains which are time-consuming. GANs are attractive as they do not make any explicit density estimation and they remedy most of these issues. GANs have generated extremely good examples in many domains such as images [21, 90, 118, 184], videos [168], music [40, 173], medicine [8], texts [145], anime creation [88] and imputation [97, 176].

2.6 Other Solutions

Cost-sensitive learning incorporates mis-classification costs in the evaluation metric [72]. This approach is more computationally efficient than data-level solutions [58]. However, mis-classification costs are often unknown and difficult to set, making this method less popular than sampling techniques [72]. A different perspective is phrasing the problem as a one-class classifier and treating it as an anomaly detection approach [58, 72, 105]. However, SMOTE and its variants remain the most studied and widely used solutions [49].

2.7 Synthesis of Literature Reviews

Data-level solutions such as over-sampling are generally better than under-sampling techniques. Specifically, SMOTE is the pioneering and popular method in imbalanced learning. However, this approach has its limitations such over-lapping classes. SMOTE variants have been designed to address some of its issues. However, most of these techniques still neglect the true probability distribution, could lead to information loss and altering the probability distribution [38, 41, 179]. Other approaches are also subjective such as distribution-based, clustering-based and kernel-based functions. GANs may be useful in augmenting the minority class by implicitly generating new synthetic samples from the true underlying probability distribution, unlike other SMOTE explicit density-based approaches.

Whilst each of the SMOTE variants have been shown to outperform the original SMOTE algorithm, there is no consensus on which approach is optimal or best in a given setting. Moreover, there have been few studies which systematically compare SMOTE taxonomies, particularly those relying on the PDF approach [49]. This work has also been less explored for tabular studies [41]. Since GANs are entirely based on deep learning, a theoretical literature on ANNs is covered in Chapter 3.

3 Neural Networks

There are different types of ANNs for various tasks such as tabular data, sequences, text and images. This section deals with fully connected layers, termed Multi-Layer Perceptron (MLP), for tabular data. Both MLPs and ANNs can be used interchangeably. This chapter describes ANNs as it forms the foundation work for GANs.

3.1 Definition

From a statistical viewpoint, an ANN represents a nested combination of several functions stacked sequentially to yield a desired output [67]. An example of an ANN is shown in Figure 3.1. Below we describe each key operation in detail.

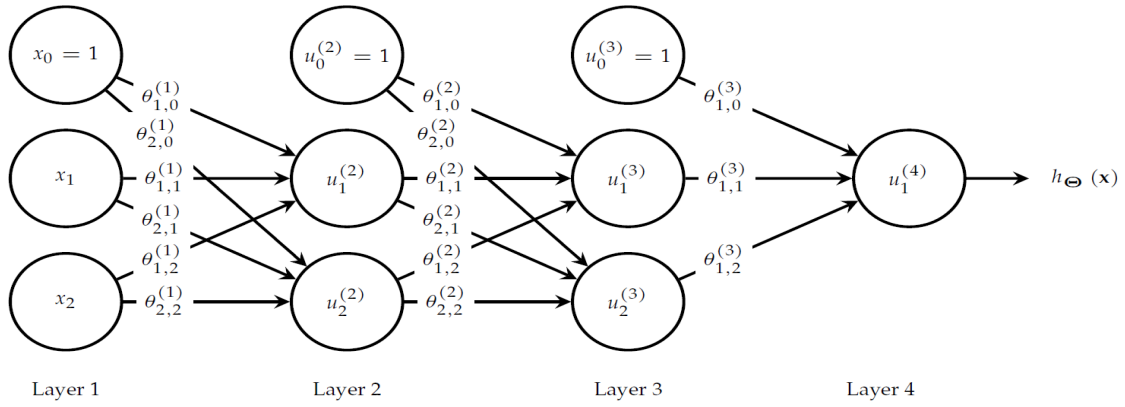


FIGURE 3.1: A fully connected MLP example with three layers

3.2 Layers

An ANN has input features, hidden layers and the target which gives the resulting output [67]. Deep Learning is concerned with many complex layers of the ANN.

Each input feature is assigned a weight θ which represents its importance. Hidden layers receive inputs from prior nodes beneath them and propagate the output to other hidden layers above them [67, 120].

As an illustration, Figure 3.1 shows an ANN with three layers, with two-dimensional input, two layers with three units and one output layer with one unit, where $u_i^l = g_l(\Theta_l + b_l)$, x_i is the input, x_0 is the bias term, $\Theta_{l,i}^l$ is a weight parameter for each layer and $h_\theta(x^n)$ is the predicted target. This ANN can be a regression or a classifier, depending on the activation function in the output.

3.3 Activation Functions

An input X is multiplied by a weight θ, results added together and the resulting sum flows through an activation function. Activation functions introduce non-linearities and transmit the resulting output into the target output [67, 120], restricting the output to a certain finite value [54]. A key characteristic of activation functions is that they must be continuously differentiable. Table 3.1 lists common activation functions where z is the result of the weight matrix Θ multiplied by the feature vector X and $g(.)$ is the activation function.

Activation Function	**Formula**
Linear	$g(z) = z$
Sigmoid	$g(z) = \frac{1}{1+\exp^{-z}}$
Hyperbolic tangent (tanh)	$g(z) = \frac{\exp^{z} - \exp^{-z}}{\exp^{z} + \exp^{-z}}$
Rectified Linear Unit (ReLU) [65]	$g(z) = max\,(0, z)$
Leaky ReLU [107]	$g(z) = max\,(0.01z, z)$
Parametric ReLU (PReLU) [74]	$g(z) = max\,(\alpha z, z)$
Softplus [125]	$g(z) = \log\left(1 + \exp^{z}\right)$
Swish [142]	$g(z) = z.sigmoid\left(\beta z\right)$

TABLE 3.1: Taxonomy of activation functions for ANNs

The linear activation function is not as useful for complex and non-linear classification problems. The sigmoid outputs a vector where each element is a probability, bounded between 0 and 1 and is typically adopted in the final layer for binary classification problems. ReLU ranges between zero and infinity and is known for being robust against vanishing gradients [34, 65, 107]. Leaky ReLU [107] solves the dying

ReLU problem. The tanh function is bounded between -1 and 1 and has been a default activation function in the hidden layers until ReLU was proposed [65]. The tanh function gives values of different signs which makes it easier to decide which scores to consider in the next layer and which to ignore. However, it shares the unfortunate weakness of vanishing gradients with the sigmoid activation function [34, 65, 107].

Parametric ReLU (PReLU) [74] is of the same form as Leaky ReLU except that it has a scalable and learnable parameter α. Softplus [125] is a smoother version of ReLU [142]. Ramachandran, Zoph, and Le [142] show that the Swish activation function behaves in a similar manner as the ReLUs and worked better on many challenging data sets. It remains to be seen if recent activation functions such as Swish [142] and Mish [119] will replace ReLU and Leaky ReLU in the future.

3.4 Gradient Descent

The weights Θ are optimised to minimise a loss function [67]. This means that training an ANN means to show it many examples, make predictions through feed-forward computations and then compare them with the actual labels to compute the resulting loss. Finally, the ANN adjusts these weights from all nodes until it gets a desired minimum loss value and thus optimal weights. Mathematically, for a binary problem, the loss function $J(\theta)$ to be minimised is:

$$J(\Theta) = \frac{-1}{N}\left[\sum_{i=1}^{N}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_\theta(x^{(i)})_k\right) + \left(1 - y_k^{(i)}\right) \log\left(1 - h_\theta(x^{(i)})_k\right)\right] \tag{3.1}$$

where N is the size of the data set, $h_\theta(x^{(i)})_k$ is the predicted target, θ_k's are the unknown coefficients, X is the feature vector and $y_k^{(i)}$ is the actual target. Gradient Descent (GD) optimises the above loss function [148]. GD finds the most optimal weights Θ iteratively [75] using the following process:

- Initialise weights $\theta_k^{(0)}$ randomly using He or Xavier initialisation;
- Loop until convergence i.e. until sufficient number of epochs t are reached:
- Compute the gradient $\frac{\partial J(\Theta)}{\partial \theta_k}$;
- Update weights θ_k using the learning rate η to move towards the minimum loss, $\theta_k^{(t)} = \theta_k^{(t-1)} - \eta \frac{\partial J(\Theta)}{\partial \theta_k}$;

- Return weights Θ.

Other optimisation approaches can be used such as second order approximations i.e. Newton's method. However, these methods tend to be infeasible for high dimensions and large training data sets [148]. Thus GD is the most popular and common approach for solving ANN weights.

3.4.1 Gradient Descent Variants

The above process is called Batch GD (BGD) as the weights are updated using the entire data set [148]. This can be very slow, intractable and does not allow to update weights online [44, 91, 177]. Stochastic GD (SGD) updates the weights one sample at a time [148].

However, SGD usually performs frequent updates and this leads to volatility as there might be fluctuations and over-shooting [101]. A compromise between BGD and SGD is called mini-batch GD and this updates weights using a batch of m training samples. Typically, mini-batch training samples can be anything from 50 to 256 but this could vary with different domains [148].

3.4.2 Learning Rate Scheme

However, even though mini-batch GD tends to be better than BGD or SGD, it may still be slow in convergence due to η [44, 91]. The learning rate η specifies how fast an ANN updates its weights. If η is too small, the model may not converge or descend slowly and this can be computationally expensive [101]. If η is too large, the model may take gigantic descents and miss the global minimum [43, 91, 128]. Adaptive learning rates have been proposed to improve η (shown in Table 3.2).

Basically, the algorithms incorporate a term to adapt η or use exponential moving average of current and/or past gradients [91, 119, 181]. Adaptive Moment estimation (Adam) [91] is the most popular and recommended algorithm for solving weights of an ANN [101, 144, 148].

Table 3.2 shows a taxonomy of GD optimisers, differing on two ways on either modifying η or modifying the gradient component or both. We describe three popular GD optimisation variants as these are typically used in most deep ANNs: Momentum [139], Root Mean Square propagation (RMSprop) [78] and Adam [91].

GD optimiser	Year	Learning rate	Gradient
Momentum [139]	1964		√
Adaptive gradient (AdaGrad) [44]	2011	√	
Root Mean Square propagation (RMSprop) [78]	2012	√	
Adaptive delta (Adadelta) [177]	2012	√	
Nesterov Accelerated Gradient (NAG) [128]	2013		√
Adaptive Moment estimation (Adam) [91]	2014	√	√
AdaMax [91]	2015	√	√
Nesterov Adam (NAdam) [43]	2015	√	√
AMSGrad [144]	2018	√	√
Rectified Adam (RAdam) [101]	2019	√	√
LookAhead, Ranger [181]	2019	√	√

TABLE 3.2: Gradient descent optimisation variants

3.4.2.1 Momentum

Momentum helps in accelerating SGD in the correct direction and dampening oscillations [139] using the following equation:

$$\theta_{t+1,i} = \theta_{t,i} - V_{t,i}$$
$$V_{t,i} = \gamma V_{t-1,i} + \eta \frac{\partial J(\Theta)}{\partial \theta_{t,i}}$$

where V_t is the velocity representing the exponential moving average of past gradients and γ is the momentum (typically 0.9) [148]. Values for V_t are typically initialised close to zero.

3.4.2.2 RMSprop

RMSprop [78] chooses a different η for each weight Θ, formulated as follows:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{E\left[g_{t,i}^2\right] + \varepsilon}}$$
$$E[g_{t,i}^2] = \gamma E\left[g_{t-1,i}^2\right] + (1-\gamma) g_{t,i}^2$$
$$g_{t,i} = \frac{\partial J(\Theta)}{\partial \theta_{t,i}}$$

where typically $\gamma = 0.9$, $\eta = 0.001$, ε avoids null division and $E[g^2]_{t,i}$ represents the running average of past gradients at time step t.

3.4.2.3 Adam

Adam is a combination of Momentum and RMSprop. Kingma and Ba [91] show a superior performance of Adam over other optimisers. Adam is defined below:

$$\begin{aligned}
\theta_{t+1,i} &= \theta_{t,i} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon}\hat{m}_t \\
\hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\
\hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\
m_t &= \beta_1 m_{t-1} + \left(1 - \beta_1\right) g_t \\
v_t &= \beta_2 v_{t-1} + \left(1 - \beta_2\right) g_t^2
\end{aligned}$$

where $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected first m_t and second moment estimates v_t of the gradients respectively, typically initialised to 0's. The parameters can be estimated via cross-validation approach or using default values proposed by the authors as per Keras documentation [52, 91, 148].

In general, Adam has been empirically shown to work well in practise and compares fairly well with other optimisers [52, 91, 101, 140, 144, 148]. However, it remains to be seen if recent optimisers such as Rectified Adam (RAdam) [101], LookAhead and Ranger [181] will consistently outperform Adam in the future.

3.5 Weight Initialisation

To conduct GD, weights Θ needs to be initialised. The weights can affect how quickly or if at all the local minimum is found by the network training algorithm [66], known as the exploding gradient problem.

Two popular approaches are the He (if using ReLU/Leaky ReLU in the hidden layers) [74] and Xavier (if using tanh in the hidden layers) [64] initialisation. He initialisation initialises weights from a standard Gaussian distribution and then multiplied by the square root of $(2/n_i)$ where n_i is number of input units for that layer. Xavier initialisation works by replacing 2 with a 1 instead.

3.6 Regularisation

The specification of many of the hyper-parameters in ANNs could often cause over-fitting or under-fitting [67]. Regularisation is a practice in ML that is used to curb over-fitting. Typically, **batch normalisation** [84], **drop-out** [160], **early stopping**, **L1** and **L2** regularisation [54, 164] can be used.

3.6.1 Batch normalisation

Batch normalisation standardizes hidden layers such that they have a mean 0 and unit standard deviation for each training mini-batch as units flow through each layer [84]. In practise, this results in faster, more stable training and a regularization effect [84, 140].

3.6.2 Drop-out

In drop-out, some units in the layer are temporarily excluded at random from the training [160]. The drop-out parameter is typically in the range $[0, 1]$ with 0.5 the most popular value for retaining the output of each layer [66, 140, 160]. This can be implemented per layer in the network. This forces the training process to be more noisy, allowing each layer to take flexible responsibility for the inputs.

3.6.3 Early Stop

GD proceeds in epochs which consist of using the training set entirely to update each parameter [148]. Initially, weights Θ are initialised. Then at each epoch, the weights are updated using partial derivatives using any GD optimiser until the process the weights do not change much i.e. until convergence [64, 74]. Typically, we require many epochs until this convergence and then we stop. Early stop is a practice where training is stopped when the cost starts increasing steadily instead of decreasing. One can then stop training the model at that epoch.

3.6.4 L1 and L2 regularisation

Regularisation enforces the ANN to learn a less complex model by adding a penalising term Equation 3.1 [164].

L1 regularisation performs sparse modelling by adding $\lambda \sum_{k=1}^{d} \theta_k$ to Equation 3.1 where λ is the importance parameter. This shrinks some coefficients to zero, yielding to implicit variable selection. This method is preferred for model explainability. L2 regularisation or ridge regression adds $\lambda \sum_{k=1}^{d} \theta_k^2$ to Equation 3.1. This is typically preferred for maximising model performance [54]. Elastic net combines both L1 and L2 regularisations. Hyper-parameterisation can be done in order to chose which approach is desirable.

3.7 Summary

There are a number of parameters to tune in ANNs. Typically, weight initialisation, activation function, η, the number of layers, regularisation approach, the number of neurons, GD optimiser and the number of epochs are required before training an ANN. There is no best approach but some heuristics and best practise are typically followed. In this work, ANNs provide theoretical foundations for GANs.

4 GAN Methodology

This chapter describes in detail the theoretical operation of GANs, their challenges and tricks to improve their training. Throughout this paper, it is assumed that both GAN networks are implemented with ANNs.

4.1 Vanilla GAN

This section describes the original GAN formulation, called MiniMax GAN (MM-GAN). This is the baseline model over which all other variants are based.

4.1.1 The Discriminator

The discriminator (D) receives generated samples from a generator G and the true data examples from $p_{data}(x)$, and must distinguish between the two for authenticity through a deep ANN [68]. The resulting output $D_{\theta_d}(x)$ for an input x is the probability of x being sampled from $p_{data}(x)$ instead of p_g, where p_g is the implicit distribution defined by G. The vector Θ_d represents learned parameters from D.

The discriminator's goal is to yield $D(x)$ near 1 for $x \sim p_{data}$ and $D(G(z))$ closer to 0 for $p_z(z)$ using the sigmoid function in the output layer. This is achieved by maximising D's loss over θ_d:

$$J_D^{MM-GAN} = \mathbb{E}_{X\sim p_{data}(x)}\left[\log D_{\theta d}\left(x\right)\right] + \mathbb{E}_{Z\sim p_z(z)}\left[\log(1 - D_{\theta d}(G_{\theta g}(z))\right] \tag{4.1}$$

4.1.2 The Generator

The generator (G) randomly picks a sample z from the prior latent space defined by $p(z)$ and then generates samples from this distribution using an ANN. This deep ANN must learn the parameters Θ_g given an input $z \sim p_z(z)$, that will give the output $G_{\theta g}(z)$. G is trained to fool D i.e. to make D's output for fake/generated

sample $D(G(z))$ closer to 1. The parameters of G are learned by minimising G' loss over Θ_g:

$$J_G^{MM-GAN} = \mathbb{E}_{Z\sim p_z(z)}\Big[\log(1 - D_{\theta d}(G_{\theta g}(z))\Big] \tag{4.2}$$

4.1.3 GAN Loss

Combining the losses for D and G, GANs solve the following minimax game in alternate steps through GD:

$$\min_{\theta_g}\max_{\theta_d} \mathbb{E}_{X\sim p_{data}(x)}\Big[\log D_{\theta_d}(x)\Big] + \mathbb{E}_{Z\sim p_z(z)}\Big[\log(1 - D_{\theta_d}(G_{\theta g}(z))\Big] \tag{4.3}$$

The above losses for D and G are the original formulation proposed by Goodfellow in 2014, called minimax GAN (MM-GAN). Since we are minimising over θ_g and maximising over θ_d, training of GANs alternate between GD on G and gradient ascent on D [67]. Typically, for every training of G, D is trained k times although an optimal choice is debatable among researchers. This is shown in Algorithm 1.

Remark 1. *Gradient based updates on the ANN can be accomplished using any of the GD optimisers reviewed in Chapter 3. Typically, SGD with Momentum for D, RMSProp or Adam for G tend to work well in practise [68, 140].*

4.1.4 Non-Saturating GAN

While the above loss function is useful for theoretical results, unfortunately it does not work well in practise and there are challenges getting the GAN to convergence, stabilise its training and getting diverse samples [7, 68, 118, 140, 153]. In practice, rather than training the above loss function for G, to provide better gradients in earlier training, Goodfellow et al. [68] suggest to maximise the following objective function for G instead:

$$J_G^{LS-GAN} = \mathbb{E}_{Z\sim p_z(z)} \log\Big(D_{\theta d}(G_{\theta g}(z))\Big) \tag{4.4}$$

This version of GAN is called non-saturating GAN (NS-GAN) and is typically used as the benchmark in most studies and in practise. This leads to the following

NS-GAN loss function:

$$\max_{\theta_g} \max_{\theta_d} \mathbb{E}_{X \sim p_{data}(x)} \left[\log D_{\theta_d}(x) \right] + \mathbb{E}_{Z \sim p_z(z)} \log \left(D_{\theta d}(G_{\theta g}(z)) \right) \tag{4.5}$$

With this new loss function, we alternate between gradient ascent on D and gradient ascent on G. The algorithm presented below is based on the original MM-GAN formulation, however, it can easily be tweaked to represent NS-GAN.

Algorithm 1 Mini-batch SG ascent of GANs with the original objective for MM-GAN [68]. The number of steps to apply to D, k, is a hyper-parameter. For every training of G, we train D k times. Goodfellow et al. [68] used $k = 1$.

1: **for** *number of epochs* **do**
2: **update the discriminator**
3: **for** k *steps* **do**
4:
- Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
- Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
- Update the discriminator D by ascending its stochastic gradient on these mini-batches:

$$\Delta_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^i\right) + \log \left(1 - D\left(G(z^i)\right) \right) \right].$$

5: **end for**
6: **update the generator**
7:
- Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient computed on this mini-batch:

$$\Delta_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D\left(G(z^i)\right) \right).$$

8: **end for**

4.1.5 Optimal Solution

Theoretically, it can be shown that for $p_g = p_{data}$, the GAN zero-sum game in Equation 4.3 has a global optima. Given enough capacity for both networks and D is trained to optimality for a fixed G, convergence of the GAN algorithm is guaranteed [68, 111, 118, 129, 140]. The optimal discriminator $D^*_G(x)$ for a fixed G is:

$$D^*_G(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \tag{4.6}$$

Assuming that D is perfectly trained and if we substitute $D^*_G(x)$ into Equation 4.3 for G's loss, this gives rise to the Jensen-Shannon (JS) divergence [100]. The JS divergence can be written as a function of the Kullback-Leibler (KL) divergence [93, 94].

Definition 1. *The KL divergence between two probability distributions p_{data} and p_g is defined as*

$$KL(p_{data}, p_g) = D_{KL}\left(p_{data}||p_g\right) = \int p_{data}(x) \log\left(\frac{p_{data}(x)}{p_g(x)}\right) dx$$

Definition 2. *The JS divergence between two probability distributions p_{data} and p_g is defined as*

$$JS(p_{data}, p_g) = D_{JS}\left(p_{data}||p_g\right) = \frac{1}{2}KL\left(p_{data}, \frac{p_{data} + p_g}{2}\right) + \frac{1}{2}KL\left(p_g, \frac{p_{data} + p_g}{2}\right)$$

If we substitute $D^*_G(x)$ into Equation 4.3, the minimum loss for G is reached if and only if $p_g = p_{data}$, thus one can show that:

$$J_G = -\log 4 + 2JS\left(p_{data}, p_g\right) \tag{4.7}$$

This equation tells us that when D has no capacity limitation and is optimal, the GAN loss function measures the similarity between p_{data} and p_g using JS divergence. However, although the above results provide a nice theoretical result, in practise, D is rarely ever fully optimal when optimising G [68]. Thus alternative

GAN architectures have been proposed to fix this issue and to get closer to optimality. Below we describe what causes this failure to convergence and how to fix it.

4.2 Challenges with GANs

GANs are notoriously difficult to train properly, difficult to evaluate, the likelihood cannot be easily be computed, suffer from the vanishing gradient problem, mode collapse, boundary distortion and over-fitting [7, 36, 67, 79, 82, 99, 153]. This section describes key challenges on GAN training.

4.2.1 Mode collapse

Mode collapse is when many latent noise values z are mapped to the same data point x, leading to a lack of diversity in the samples that are created i.e. underfitting. This is regarded as the most significant problem with GANs [99, 111]. Many studies have spent lots of time in varied contexts to fix this.

4.2.2 Vanishing gradient

This occurs when D becomes perfect in its training without giving G the chance to improve. As a result, GANs may fail to converge and thereby leading to poor generated samples [7].

4.3 Improved GAN Training

There are many GAN architectures which avoid the problems that come with the vanilla GAN. We briefly describe some of the most common and popular GAN solutions. Given the vast number of taxonomies, we are not able to cover all of them but only discuss the most popular and those subsequently used in this work.

4.3.1 Conditional GANs

The first extension of GAN was the Conditional GAN (cGAN) which gave the generator the label Y in the latent space, making them class conditional [30, 118, 130].

Most of the GAN variants can be modified to include cGAN. cGAN allows to create diversified samples and forcing G to create specific samples and thereby fixing mode collapse problem.

4.3.2 Loss Variants

There are a number of GAN architectures which change the loss function to improve GAN training and stability. The loss function for GAN measures the similarity between p_{data} and p_g using JS. Unfortunately, JS tends not to be smooth enough to ensure a stable training [82, 111]. Broadly, there are two loss function groups with better properties i.e. f-divergence [129] and IPM [123]. Among these loss groups, WGAN is arguably the most popular and well-studied [79, 82, 170]. WGAN is considered a general unified framework under the recently proposed Relativistic GAN (RGAN) [89]. Thus we adopt WGAN in this work.

4.4 WGAN

This section describes WGAN and its improved training using WGAN-GP.

4.4.1 Wasserstein distance

IPM generalises a critic function f belonging to an arbitrary function class where IPM measures the maximal distance between two distributions under some functional frame f [79]. Among the IPMs, the Wasserstein distance is the most common and widely used metric [111]. Informally, the Earth mover (EM) [147] distance $W(p_{data}, p_g)$ measures the minimal changes needed to transform p_g into p_{data}. More formally, EM between two probability distributions p_{data} and p_g is:

$$W\left(p_{data}, p_g\right) = \inf_{\gamma \sim \Pi(p_{data}, p_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\| x - y \|\right] \tag{4.8}$$

where $\Pi_{(p_{data}, p_g)}$ represents a set of all joint probability distributions whose marginal distributions are respectively $p_{data}(x)$ and $p_g(x)$. Precisely, $\gamma(x, y)$ is a transport plan i.e. percentage of mass that should be moved from x to y to transform p_g into p_{data}. The infimum in Equation 4.8 is intractable as it is tricky to exhaust all the

elements of $\Pi_{(p_{data}, p_g)}$ [7]. This is solved using the following functional format:

$$W\left(p_{data}, p_g\right) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim pdata}\left[f(x)\right] - \mathbb{E}_{x \sim pg}\left[f(x)\right] \tag{4.9}$$

where the supremum is taken over a 1-Lipschitz function f. A function f is 1-Lipschitz if for all $x_1, x_2 : |f(x_1) - f(x_2)| \leq |x_1 - x_2|$.

4.4.2 The Critic

In WGAN, D's output is not a probability anymore but can instead be any number and for this reason, D is typically called the *critic*. The WGAN critic tries to maximise the difference between its predictions for real samples and generated samples, with real samples scoring higher. Arjovsky, Chintala, and Bottou [7] force the critic to be *1-Lipschitz continuous* for the loss function to work well:

$$J_{WGAN} = \max_{w \in W} \mathbb{E}_{X \sim p_{data}(x)}\left[D(x)\right] + \mathbb{E}_{Z \sim p_z(z)}\left[1 - D(G(z))\right] \tag{4.10}$$

where W is the set of 1-Lipschitz continuous functions. Typically, to enforce the Lipschitz constraint, the critic weights w are clipped to lie within a small range, usually $[-0.01, 0.01]$ after each training batch [7, 69].

The critic is trained to convergence so that the gradients of G are accurate, thus removing the need to balance the training of G and D by simply training D several times between G's updates, to ensure it is close to convergence. Typically, 5 critic updates to 1 generator update is used [7]. The WGAN training algorithm is shown in Algorithm 2 as per the original paper [7]. WGAN used the RMSProp version of gradient GD with a small learning rate and no momentum [7]. However, Adam may also be used as it is a combination of RMSProp with Momentum.

4.5 Improved WGAN Training

Even though WGAN has been shown to stabilise GAN training, it is not generalized for deeper training due to weight clipping which tends to localise most parameters at -0.01 and 0.01 [69, 111]. This effect dramatically reduces the modelling capacity

Algorithm 2 Wasserstein GAN [7]. Default experiments used $\eta = 0.00005$, $c = 0.01$, $m = 64$ and $n_{critic} = 5$.

Require: η, the clipping parameter c, the batch size m, the number of iterations of the critic per generator iteration n_d.
Require: initial critic parameters w_0, initial parameters of the generator Θ_0
while θ *has not converged* **do**
for $t = 0, \cdots, n_{critic}$ **do**
Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
$g_w \leftarrow \Delta_w \left[\frac{1}{m} \sum_{i=1}^{m} f_w \left(x^{(i)} \right) - \frac{1}{m} \sum_{i=1}^{m} f_w \left(g_\theta \left(z^{(i)} \right) \right) \right]$
$w \leftarrow w + \eta.RMSProp\left(w, g_w\right)$
$w \leftarrow clip\left(w, -c, c\right)$
end for
Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
$g_\theta \leftarrow -\Delta_\theta \frac{1}{m} \sum_{i=1}^{m} f_w \left(g_\theta(z^{(i)}) \right)$
$\theta \leftarrow \theta + \eta.RMSProp\left(\Theta, g_\theta\right)$
end while

for D. Gulrajani et al. [69] further amend WGAN through an addition of a gradient-penalty (GP) to the loss function, coming with WGAN-GP. In total, three changes are made to WGAN critic to convert it to WGAN-GP: include a GP to the loss function; do not clip critic weights; and do not use batch normalisation layers in the critic. WGAN-GP defined using the following loss function:

$$\mathbb{E}_{X \sim p_{data}(x)} \left[D(x) \right] + \mathbb{E}_{Z \sim p_z(z)} \left[1 - D(G(z)) \right] + \lambda \mathbb{E}_{\tilde{x} \sim p_{data}} \left[\left(\| \Delta D(\tilde{x}) \|_2 - 1 \right)^2 \right] \tag{4.11}$$

where $\tilde{x}$ samples uniformly along the straight line between points sampled from p_{data} and p_g and λ is the GP term. Gulrajani et al. [69] show a better distribution of learned parameters compared to WGAN and this method has been the default method in most GAN loss variants.

We adopt the conditional version of WGAN-GP, called WCGAN-GP, as an alternative to SMOTE density estimation approaches. Once WGAN-GP is trained to convergence, G can be used to create new samples by feeding it the latent space Z.

5 SMOTE Methodologies

This chapter describes the theoretical operation of each SMOTE density estimation approach chosen in this study, i.e. SMOTE, PDFOS and RWO. We did not consider RACOG due to its relatively high computation requirements.

5.1 SMOTE

Considering a random minority instance x, a new instance s is generated by considering its k-NNs. These k-NNs are found by using the Euclidean distance metric [27]. Initially, an instance y is generated at random from the k-NNs. Then a new synthetic minority instance s is generated as follows:

$$s = x + \alpha \left(y - x\right) \tag{5.1}$$

where α is randomly generated from the Uniform distribution $[0, 1]$. SMOTE parameters are the value of k and the number of minority cases to generate. The number of k-NNs can be varied such that an optimal metric is found, whilst restricting the number of generated instances to ensure a balanced class distribution.

5.2 PDFOS

PDFOS [59] uses the Parzen-window (PW) method [132], a widely used non-parametric statistical method to estimate a kernel density estimate (KDE) for a specific point from a sample. This is done by a mixture of continuous distributions K, called kernels, that are centered at each data point and have bandwidth of h.

5.2.1 Kernel Function

Denoting the unknown PDF generating the minority class set by $p(x)$, then a general KDE for $p(x)$ is formulated as:

$$\hat{p}\left(\mathbf{x}\right) = \frac{1}{mh^d} \sum_{i=1}^{m} K\left(\frac{\left(\mathbf{x} - \mathbf{x}_i\right)}{mh}\right) \tag{5.2}$$

where K is a differentiable non-negative kernel; x is the feature vector; d is the number of features; m is the number of minority class examples and h is the bandwidth. Due to its convenient mathematical properties and popularity, a multivariate d-Gaussian kernel function is typically adopted. The PDF of a d-Gaussian function with a mean of 0 and co-variance matrix Ψ is:

$$\Phi^{\Psi}(x) = \frac{1}{\sqrt{(2\pi.det(\Psi))^d}} \exp\left(-\frac{1}{2} x \Psi^{-1} x^T\right) \tag{5.3}$$

Define $S^+ = x_i = (w^{(i)}_1, \cdots, w^{(i)}_d)_{i=1}^m$ as minority class instances, then an unbiased co-variance estimate of S^+ is given by:

$$U = \frac{1}{m-1} \sum_{i=1}^{m} \left(x_i - \bar{x}_i\right)\left(x_i - \bar{x}_i\right)^T \tag{5.4}$$

where $\bar{x}_i = \frac{1}{m}\sum_{i=1}^{m} x_i$. PDFOS uses the kernel function $\Phi_h(x) = \Psi^U(\frac{x}{h})$ where h is a bandwidth that needs to be estimated.

5.2.2 Bandwidth

The bandwidth h is determined using the mean integrated squared error (MISE) [156], by minimising this function:

$$MISE(h) = \frac{1}{m^2 h^d} \sum_{i=1}^{m} \sum_{j=1}^{m} \Psi^*_h(x_i - x_j) + \frac{2}{mh^d} \Psi_h(0) \tag{5.5}$$

where $\Psi^*_h \approx \Psi_{h\sqrt{2-2\Psi_h}}$. Typically, using Silverman's approach [156], h is initialised as

$$h_{Silverman} = \left(\frac{4}{m(d+2)}\right)^{\frac{1}{d+4}} \tag{5.6}$$

where d is the number of variables and m is the size of the minority cases. SGD can be used to find an optimal h value.

5.2.3 Generating Synthetic samples

After a suitable h is found, synthetic sampling for a new instance s is accomplished using:

$$s = x_i + hRr \tag{5.7}$$

where $x_i \in S^+$, $r \sim N^d(0,1)$; $N^d(0,1)$ is a multivariate d-Gaussian distribution with mean 0 and variance 1; r is a sample from this distribution and R is an upper-triangular Cholesky decomposition of the positive-definite unbiased co-variance matrix (U) of the minority class defined as $U = R.R^T$. For the PDFOS algorithm to work, U should be a strict positive-definite matrix [35, 59].

Algorithm 3 PDFOS [59]

Require: $S^+ = \{x_i = (w^{(i)}_1, \cdots, w^{(i)}_d\}_{i=1}^m$, minority class instances.
Require: T, the required number of synthetic minority instances to over-sample.
  Initialise $S' = \emptyset$.
  Find for h which minimizes $MISE(h)$.
  Solve for U the unbiased positive definite co-variance matrix of S^+.
  Compute $U = R.R^T$ with Choleski decomposition.
  for $i = 1, \ldots, T$ **do**
    Randomly choose $x \in S^+$.
    Randomly pick r from $N^d(0,1)$.
    Generate a synthetic sample using $S' = S' \cup \{x + h.R.r\}$.
  end for
  return S', the synthetic minority cases.

5.3 RWO Sampling

Random Walk Over-sampling (RWO) [179] generates new minority class instances with approximately the same mean and variance as the original minority class data.

5.3.1 Central Limit Theorem

Given a collection of minority class variables, $W_1, \cdots, W_m$, with mean $E(W_i) = \mu$ and variance $Var(W_i) = \sigma^2 < \infty$, central limit theorem (CLT) states that:

$$\lim_m P\left[\frac{\sqrt{m}}{\sigma}(\bar{W} - \mu) \leq z\right] = \Psi(z) \tag{5.8}$$

where $\bar{W} = \frac{1}{m}\sum_{i=1}^{m} W_i$, $\Psi \sim N(0,1)$ and thus $\frac{\bar{W}-\mu}{\sigma/\sqrt{m}} \rightarrow N(0,1)$. This idea generates synthetic samples that conform to the sample mean and variance of the original data.

Algorithm 4 RWO [179]

Require: $S^+ = \{x_i = (w^{(i)}_1, \cdots, w^{(i)}_d\}_{i=1}^m$, minority class instances.
Require: T, the required number of synthetic minority cases to over-sample.
  Initialise $S' = \emptyset$.
  for each $j = 1, \ldots, d$ **do**
    if $j - i$th variables is continous **then**
    $\sigma'_j = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(w_j^{(i)} - \frac{\sum_{i=1}^{m} w_j^{(i)}}{m}\right)^2}$
    end if
  end for
  Assign $M = \frac{T}{m}$
  for $t = 1, \ldots, M$ **do**
    for $i = 1, \ldots, m$ **do**
      for $j = 1, \ldots, d$ **do**
        if $j - i$th variables is continuous **then**
          Pick $r \sim N(0,1)$
          $\hat{w}_i(j) = w_i(j) - r_j \times \frac{\sigma_i}{\sqrt{m}}$
        else
          Pick w_j uniformly over $\{(w^{(1)}_j, \cdots, w^{(m)}_j\}$
        end if
      end for
      $S' = S' \cup \{(w_1, \cdots, w_d)\}$
    end for
  end for
  S' = Pick T random fraud cases from S'
  return S', the synthetic minority cases

5.3.2 Generating Synthetic samples

Let $S^+ = \{x_i = (w^{(i)}_1, \cdots, w^{(i)}_d\}_{i=1}^m$ be the minority class instances. Suppose that we fix some $j \in \{i, \cdots, d\}$, and assume that the $j - ith$ column is continuous with mean u_j and variance σ_j^2. Then synthetic cases are created as follows where $\hat{w}_i$ is a new minority class instance:

$$\hat{w}_i(j) = w_i(j) - r_j \times \frac{\sigma_i}{\sqrt{m}},\ i \in \{1, 2, 3, \ldots, T\},\ j \in \{1, 2, 3, \ldots, m\} \tag{5.9}$$

where T is the number of synthetic cases to create; m is the original number of minority examples; σ_i is the standard deviation for the ith attribute of w and $r \sim N(0, 1)$. It can be shown that the generated instances have the same mean and variance as the original minority class data [179].

6 Experiments

This chapter outlines the experiments of comparing SMOTE density-based approaches and WCGAN-GP for synthetic data generation.

6.1 Data

This section describes the data sources, data sets and any pre-processing applied before the LR model is trained on the original and over-sampled data sets.

6.1.1 Data Sets

We considered 5 publicly available imbalanced data sets from the Machine Learning Repository UCI. The data sets are described below and shown in Table 6.1.

Imbalanced Data Set	Majority Cases	Minority Cases	Number of Features	Numeric Features	Ordinal Features
Credit Card Fraud	284,807	492	31	31	0
Pima Indians Diabetes	500	268	8	8	0
Glass Identification	144	70	9	9	0
German Credit Scoring	700	300	20	14	6
Breast Cancer Wisconsin	357	212	28	28	0

TABLE 6.1: Imbalanced data sets used in the experiments

6.1.1.1 Credit Card Fraud

European public credit card fraud transactions made in 2013 are utilised [37]. This data is highly imbalanced, with 492 fraudulent transactions out of a total of 284, 807 transactions, representing a mere 0.172% of fraud cases. This data set contains 31 anonymised numeric features *(Time, Amount, V0,V1,...V28)* and the Class indicator

showing 1 for frauds and 0 for non-fraudulent cases. The goal is to predict whether a transaction is fraudulent or not.

6.1.1.2 Pima Indians Diabetes

This data set contains the prediction of the onset of diabetes within 5 years in Pima Indians given some medical details, representing 34.90% of diabetic cases out of a total of 768 women [157]. There are 8 medical variables: plasma glucose concentration, diastolic blood pressure, triceps skin fold thickness, serum insulin, body mass index, number of times pregnant, diabetes pedigree function and age in years. All the variables are numeric.

6.1.1.3 Glass Identification

This data set is used to determine whether the glass type is float or not in terms of their oxide content [47]. There are 32.71% of float glass types out of a total of 214 cases. The original data was a multi-class classification problem. In this report, we considered the binary version of the data by choosing the smallest class as the minority class and collapsing the rest of the classes into one as was done in the KEEL [3] imbalanced data set repository. There are 9 real variables.

6.1.1.4 German Credit Scoring

This data set is a German Credit Scoring problem from the UCL Machine Learning Repository. There are 1000 observations with 20 variables. The dependent variable is the evaluation of customer's current credit status which indicates whether a borrower's risk is good or bad. There are 14 numeric variables and 6 categorical variables.

6.1.1.5 Breast Cancer Wisconsin

This data set represents the characteristics of a cell nuclei that is present in the digitised image of a breast mass [161]. The data is used to predict the presence of benign or malignant cancer, with 37.25% being malignant samples from a total of 569 nucleus cases. There are 28 real-valued variables for each cell nucleus.

6.1.2 Data Pre-processing

This section describes how the data was pre-processed before the LR model was trained, particularly how variables were treated and converted to the same basis so that there is avoidance of the dominance of certain variables. This ensures consistency and stability of the LR model [67, 120].

6.1.2.1 Continuous Variables

Many ML methods expect data to be of the same scale to avoid the dominance of certain numeric variables and this can affect the accuracy of specific models [84, 120]. Normalisation re-scales the data to the range between 0 and 1. Standardisation centers the data distribution to $N(0,1)$. We adopt normalisation as it does not assume any specific distribution. This potentially speeds up convergence [67, 120].

6.1.2.2 Categorical Encoding

Categorical encoding is a process of converting categories to numbers, using One-Hot Encoding or Label Encoding [137]. One-Hot Encoding takes a categorical column and then splits the column into multiple columns. The numbers are replaced by 1s and 0s depending on which column has what value. It creates additional features based on the number of unique values in the categorical feature. Every unique value in the category is added as a feature. Label Encoding converts all the categorical variables into numeric numbers based on their alphabetical order and may give false sense of the impact of that feature category. In this report, the variables that are categorical were converted using One-Hot Encoding before the LR model was trained.

6.1.3 Train-Test Split

ML models are usually trained and tested on unseen data. Two approaches to split the data are cross-validation (CV) and train-test split [54]. CV divides the data into K subsets that can lack sufficient credibility and can result in higher variability of predictions, if the data size is too small [54]. Train-test split, however, can allow a larger subset of the data to be used for estimating model coefficients and results in more reasonable results [120].

Existing literature typically uses a 70%-30% train-test split, especially if the data is large. This technique is simple, easy to understand and widely used, despite giving noisy estimates sometimes [54, 67, 120]. CV is typically used to optimise parameters of a classifier. This work adopts 75%-25% train-test split.

6.2 SMOTE Implementations

Over-sampling is performed on the 75% training data using the R *imbalance* library [35]. The R imbalance library contains functions for performing SMOTE, RWO and PDFOS. Over-sampling is performed to ensure a balanced class distribution in each data set i.e. over-sample the minority class to the size of the majority class.

6.2.1 SMOTE

The two parameters to tune are the number of neighbors (K-NN) and the over-sampling rate. We kept the over-sampling rate the same to ensure balanced class distributions within each data set. We varied the number of K-NNs for each data set to ensure optimal parameters are chosen through a 10-fold CV.

This was done through a grid search scheme, with values of K-NN ranging from 1 to 15, optimised using the Area under the Precision-Recall Curve (AUPRC) defined in Section 6.5. The best parameter values for each data set are shown in Table 6.2 below.

Data Set	**Value of K-NN**
Credit Card Fraud	6
Pima Indians Diabetes	9
Glass Identification	10
German Credit Scoring	12
Breast Cancer Wisconsin	10

TABLE 6.2: Optimal parameter values for K-NN for each data set

6.2.2 PDFOS

Due its popularity and wide use, a Gaussian kernel function is chosen [59]. The bandwidth h is automatically obtained through MISE as defined in Equation 5.5.

The co-variance matrix of S^+ was found to be semi positive-definite, thus PDFOS could be used for over-sampling.

6.2.3 RWO

There are no parameters to tune in RWO as it uses minority class data to capture the first and second moments of each numeric feature. If the feature is categorical, RWO picks the minority instances uniformly over the distribution of each feature.

6.3 GAN Implementation

Given its popularity and wide use, WGAN is adopted for an alternative synthetic sample generation. Specifically, we adopt the conditional version of WGAN with GP, thus we use WCGAN-GP [69, 118]. Below we describe how parameters are chosen and results generated.

6.3.1 Software

GANs can be implemented in a number of open-source neural-network libraries in Python [51]. Due to its simplicity and faster computations, the high-level Keras library [52] with Tensorflow [163] back-end is chosen to implement WCGAN-GP. This is trained using all minority cases of each data set.

6.3.2 The Generator

This section describes how the parameters for G are chosen.

6.3.2.1 Latent Noise

The random noise for z is generated from $N(0,1)$ with 100 dimensions. This is based from GAN hacks which suggest to sample from a spherical distribution [153].

6.3.2.2 Activation Function

ReLU is adopted in the hidden layers [140, 153]. For G's output later, tanh is adopted. No drop out or batch normalisation is applied following advise from Gulrajani et al. [69] for WGAN-GP.

6.3.2.3 Layers

The layers are chosen such that they are ordered in an ascending manner for G. For simplicity, after a number of iterations, 3 layers were chosen for each data set. In the first layer, there were 128 units, in the second layer 256 units and in the third layer 512 units. These layers worked well in the experiments conducted. The output layer had the data dimension of the data as the number of units.

Weights are initialised using the He initialisation method and ReLU is adopted [74]. Adam is used to optimise the weights of G [140, 153]. We used default values with $\beta_1 = 0.5$ and $\beta_2 = 0.9$ for G [91].

We used a batch size of 128 when optimising the gradients for faster training [84]. Initial η for G was fixed at 0.00004. The number of epochs were found to be $5,000$ where the GAN training was found to be stable.

6.3.3 The Critic

Leaky ReLu is adopted with a negative slope of 0.2 [107, 140]. As per the generator, 3 layers were used in the hidden layers. The layers were arranged in a descending manner, with 512 units in the first layer, 256 units in the second layer and 128 units for the last layer. The critic gives the output a single value using a linear function [7]. Adam was used with the following default parameters in Keras [52]: $\eta = 0.00001$, $\beta_1 = 0.5$, $\beta_2 = 0.90$ and $\varepsilon = 10^{-8}$.

Critic weights were also initialised using the He method and a similar batch size as in the generator was used. We pre-trained the critic 100 times at each adversarial training step [7]. This ensures faster convergence at each step before G is updated. We used WGAN with a GP with the default values as per the original paper [69]. The GP value was left unchanged at 10. We call this model WGAN-GP. We found that after 5000 epochs, the losses plateaued and did not change much.

6.3.4 Labels

Typically, to boost faster training and fix mode collapse, additional information can be incorporated in both G and D using cGAN [118]. We used the conditional version of WGAN-GP where class labels were added to the minority cases. To accomplish this, clustering was done on the minority cases in order to induce class labels

on the training data.

We explored a number of common mechanisms considering k-means, AHC [169], Hierarchical DBSCAN [46] and t-SNE [108]. The details of these algorithms are beyond the scope of this report. Due to its wide use and simplicity, we adopted k-means clustering with 2 clusters for each data set. This yielded labels that could be fed into G and D to induce generated samples. We call the final model WCGAN-GP after incorporating these class labels into the training.

6.3.5 Training WGAN-GP

Figure 6.1 shows the critic loss for each epoch, where after 1000 epochs, the loss starts to plateau. Thus we decided to stop the training after 5000 epochs. We repeated this experiment for each data set and adopted WCGAN with GP after $5,000$ epochs as the model to use for synthetic sample generation.

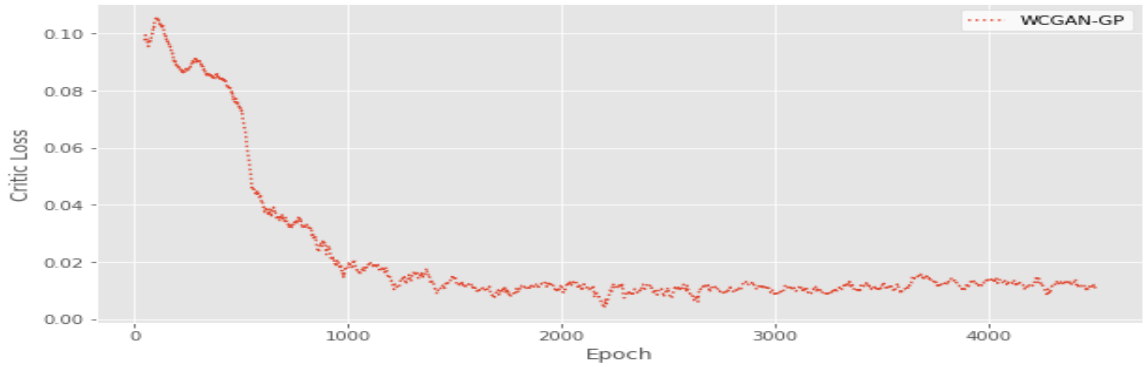


FIGURE 6.1: Difference between generated and real data critic loss

Figure 6.2 presents the experiments of training WCGAN with GP. For comparative purposes, using similar parameters, we show the quality of samples generated for WCGAN with GP, WGAN, cGAN and non-saturating GAN on the credit card fraud data. The version of the WCGAN was incorporated with an improved WGAN training using the GP term as per the paper by Gulrajani et al. [69]. We consider this for two combinations of the features for illustrative purposes up to 5000 epochs.

The results show the superiority of samples generated by WCGAN with GP. There is a clear mode collapse problem on the vanilla GAN and cGAN. WGAN and WCGAN with GP show better samples. There is also clear damped oscillations and

unstable losses for GAN and cGAN where Wasserstein GANs exhibit stable training and losses, especially after 1000 iterations where it seems to settle and stabilise.

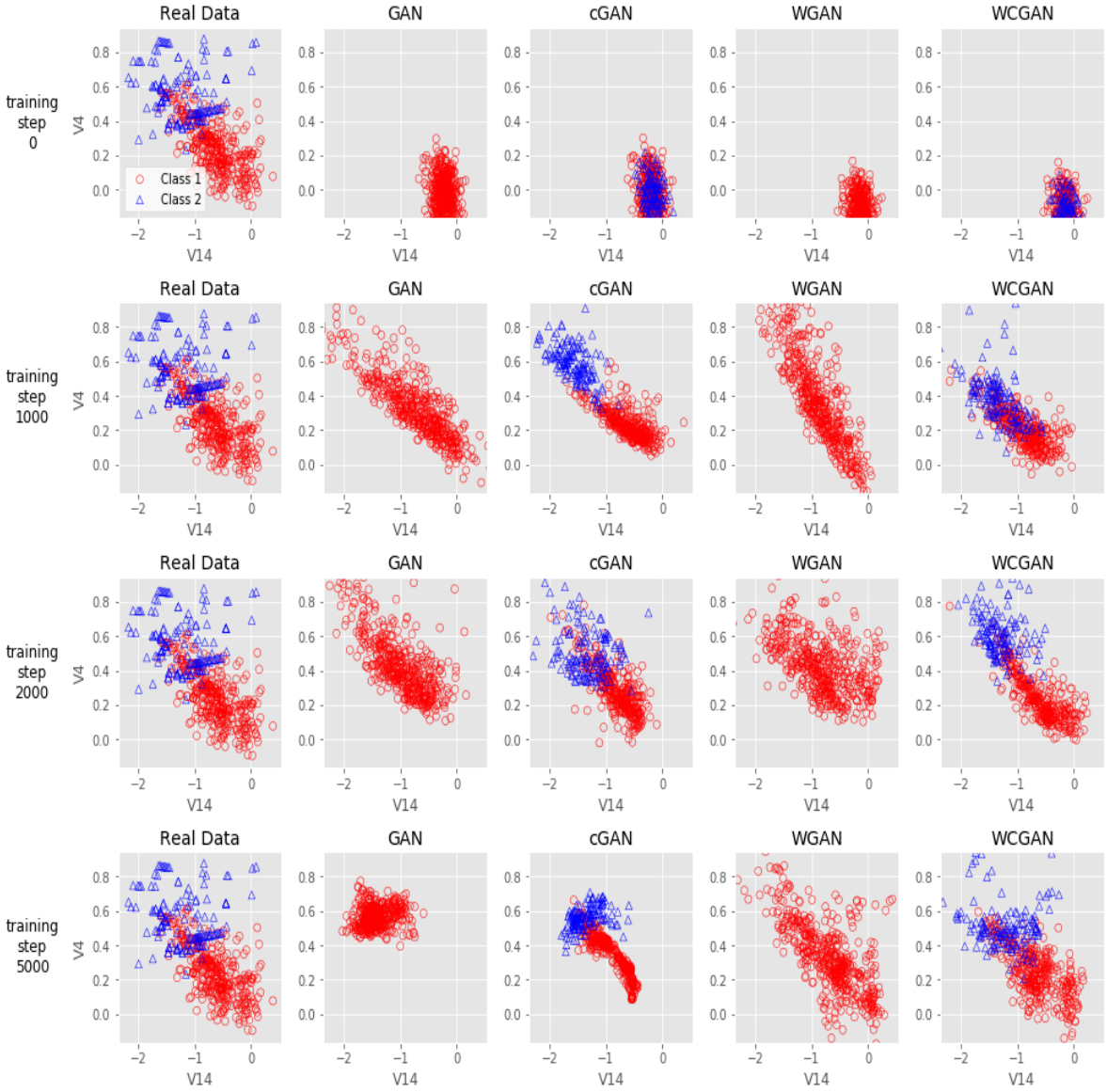


FIGURE 6.2: Comparison of GAN experiments ran on fraud data cases

6.3.6 Generating Synthetic samples

Once the WCGAN with GP is trained to 5000 epochs, the learned generator distribution is used to create more synthetic samples by feeding it the number of samples to output.

6.4 Logistic Regression

LR is trained using Python 3.7 [51] on both the imbalanced training data and over-sampled data sets to predict the likelihood of each minority case using this equation:

$$\log\left(\frac{h_\theta(x^{(d)})}{1-h_\theta(x^{(d)})}\right) = \theta_0 + \sum_{i=1}^{d}\theta_i X_i,\ 0 \ < \ h_\theta(x^{(d)}) \ < \ 1 \tag{6.1}$$

where $h_\theta(x^{(d)})$ is the probability of the given minority case, θ_i's are the estimated coefficients using SGD, X_i is the feature vector for sample i and d is the number of features to include in the LR model. The coefficients are estimated by minimising a loss function through SGD in Equation 3.1. Typically, classification is such that when $h_\theta(x^{(d)}) \geq 50\%$ for each instance, assign the minority case, otherwise the majority case. We varied the regularisation parameter λ considering both L1 and L2 regularisation using the values $[0.001, 0.01, 0.1, 1, 10, 100, 1000]$ through a 10-fold CV on the training data sets. The LR model was optimised for each data set.

6.5 Evaluation

This section describes evaluation metrics adopted to compare the different over-sampling methods.

6.5.1 Confusion Matrix

The confusion matrix returns a report showing how predicted classes on unseen test data using the LR model compare to actual observed classes, as depicted in Table 6.3.

Confusion Matrix	**Predicted: Minority**	**Predicted: Majority**
Actual: Minority	True Positive (TP)	False Negative (FN)
Actual: Majority	False Positive (FP)	True Negative (TN)

TABLE 6.3: The confusion matrix

TN is the number of majority cases that were correctly classified as such. **FP** is the number of majority cases that were incorrectly classified as minority. **TP** is

the number of minority cases that were correctly classified as minority. **FN** is the number of minority cases that were incorrectly classified as majority. Using these definitions, Table 6.4 presents the most well known evaluation metrics for binary classification problems. Accuracy, Precision, Recall and F1-Score should be close to 100% for a LR model to do well on the testing data. Accuracy can be misleading and inappropriate when there are imbalanced classes and thus may be biased towards majority cases [27, 58, 72]. Thus we do not use rely on it in this work.

Metric	**Formula**
Accuracy	$\left(\frac{TP+TN}{TP+TN+FP+FN}\right)$
Precision	$\left(\frac{TP}{TP+FP}\right)$
Recall	$\left(\frac{TP}{TP+FN}\right)$
F1-Score	$2 * \left(\frac{Precision*Recall}{Precision+Recall}\right)$

TABLE 6.4: Evaluation metrics for binary classification problems

Precision is the ability of the LR model not to label a minority case that is actually majority. Recall is the ability of the LR model to find all minority cases. F1-Score is a harmonic mean between Precision and Recall [72]. F1-Score puts equal weight to both Precision and Recall. The higher the scores are towards 100%, the better is the LR model. However, these scores are influenced by what threshold is used to decide between the two binary classes.

6.5.2 Precision-Recall and ROC curve

The ROC curve [20, 71] measures a classifier's performance on a test set over different decision thresholds by varying the Precision and the FP rate. AUC measures the performance of the LR model trained on both imbalanced and over-sampled data sets and tested on unseen data with values close to 100% considered excellent performance [14, 71]. We also compute the Precision-Recall curve and compute AUPRC to get a weighted score. Table 6.5 shows the AUC/AUPRC scale for interpreting performance of classifiers [14].

Metric Value	Performance
50% <AUC/AUPRC <= 60%	Poor
60%<AUC/AUPRC <= 70%	Fair
70% <AUC/AUPRC <= 80%	Good
80% <AUC/AUPRC <= 90%	Very Good
90% <AUC/AUPRC <= 100%	Excellent

TABLE 6.5: Interpretation of AUC/AUPRC performance

6.6 Statistical Hypothesis Testing

Friedman test [55] followed by a post-hoc Nemenyi test [127] are performed to verify the statistical significant differences between WCGAN-GP, SMOTE, PDFOS and RWO.

6.6.1 Friedman test

The Friedman test is a non-parametric ranking test to determine whether all over-sampling methods perform similarly in mean performance rankings based on the measures above, when normality does not hold [55] on 30 experiments.

6.6.2 Post-hoc Nemenyi test

If the null hypothesis is rejected, a post-hoc test can be applied where WCGAN-GP is considered as the control method. The post-hoc Nemenyi test evaluates pairwise comparisons between the over-sampling methods if the Friedman test suggests that there is a difference in performance [127, 136]. We adopt WCGAN-GP as the control method.

6.6.3 Implementation

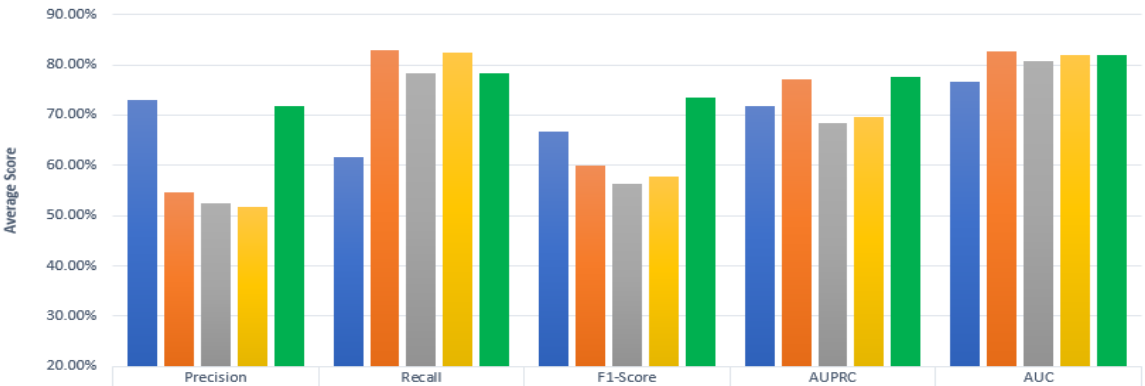
Both tests are conducted using the Pairwise Multiple Comparison Ranks Package (PMCMR) [136] available in R. We assume statistical significance of the alternative hypothesis at p-values < 0.05. In other words, we fail to reject the null hypothesis when the resulting p-value is higher than 0.05, suggesting that there is no difference between the over-sampling methods.

7 Results and Discussion

This chapter presents the results of all the LR models applied on the baseline and over-sampled data sets, with metrics on Precision, Recall, F1-Score, AUC and AUPRC computed on the same unseen test data.

7.1 Comparisons

Table 7.1 presents the evaluation metrics (based on the testing set) of the LR model applied on the baseline and over-sampled data sets for a default threshold of 50% across 30 experiments. Figure 7.1 shows the average performance across all data sets from each evaluation metric. Bold shows an algorithm that performs the best for that data set i.e. a higher score for that metric.



	Precision	Recall	F1-Score	AUPRC	AUC
Baseline	72.97%	61.68%	66.62%	71.87%	76.66%
SMOTE	54.60%	83.00%	59.88%	77.03%	82.55%
PDFOS	52.56%	78.20%	56.43%	68.39%	80.73%
RWO	51.84%	82.54%	57.85%	69.56%	81.83%
WCGAN-GP	71.90%	78.40%	73.55%	77.71%	81.88%

FIGURE 7.1: Average performance across all data sets

Figure 7.1 depicts that on the average, over-sampling increases Recall despite slightly reducing the Precision. On average, the F1-Score is highest for WCGAN-GP, followed by Baseline, then SMOTE, with PDFOS and RWO performing the worst.

Data set	Precision	Recall	F1-Score	AUPRC	AUC
Credit Card Fraud					
Baseline	85.710%	63.410%	72.893%	74.600%	81.700%
SMOTE	5.110%	**93.333%**	9.689%	72.284%	**98.358%**
PDFOS	6.290%	90.240%	11.760%	48.270%	93.960%
RWO	6.670%	90.240%	12.422%	48.460%	94.030%
WCGAN-GP	**86.240%**	76.420%	**81.034%**	**81.350%**	88.200%
Pima Indians Diabetes					
Baseline	74.470%	56.450%	64.220%	72.490%	73.610%
SMOTE	53.535%	**80.303%**	64.242%	68.183%	75.480%
PDFOS	61.430%	69.350%	65.150%	70.340%	74.290%
RWO	64.290%	72.580%	**68.184%**	72.860%	**76.670%**
WCGAN-GP	**75.510%**	59.680%	66.668%	**74.100%**	75.220%
German Credit Scoring					
Baseline	**60.320%**	51.350%	55.475%	63.030%	68.570%
SMOTE	47.826%	70.513%	56.995%	58.838%	69.610%
PDFOS	59.380%	51.350%	55.074%	62.560%	68.290%
RWO	54.880%	60.810%	57.693%	63.640%	69.890%
WCGAN-GP	46.510%	**81.080%**	**59.112%**	**66.600%**	**70.940%**
Glass Identification					
Baseline	50.000%	42.860%	46.156%	53.840%	63.930%
SMOTE	**73.913%**	70.833%	**72.340%**	**87.286%**	72.860%
PDFOS	41.380%	85.710%	55.814%	65.400%	71.610%
RWO	40.620%	**92.860%**	56.517%	67.670%	72.680%
WCGAN-GP	55.000%	78.570%	64.705%	69.560%	**78.040%**
Breast Cancer Wisconsin					
Baseline	94.340%	94.340%	94.340%	95.390%	95.500%
SMOTE	92.593%	**100.000%**	96.154%	**98.556%**	96.450%
PDFOS	94.340%	94.340%	94.340%	95.390%	95.500%
RWO	92.730%	96.230%	94.448%	95.180%	95.890%
WCGAN-GP	**96.230%**	96.230%	**96.230%**	96.930%	**97.000%**

TABLE 7.1: Evaluation metrics based on a default threshold of 50%

While the univariate results on Precision, Recall and F1-Score are useful, they do not give the entire picture over different thresholds [14]. Since AUC and AUPRC are based on varied thresholds, these metrics are typically preferred over one dimension measurements such as Precision, Recall and F1-Score [14, 58, 105]. Since we are also comparing the above results with the Baseline model, these metrics are impacted by class imbalance [72]. Thus we rely more on the AUC and AUPRC.

7.1.1 Performance between over-sampling techniques

This section compares the performance of each over-sampling technique against the Baseline and other techniques using the AUPRC and AUC.

7.1.1.1 SMOTE

SMOTE appears worse than the Baseline on 3 of the 5 data sets used on the AUPRC. On the contrary, SMOTE shows a superior AUC score than the Baseline for all data sets. SMOTE appears better than both PDFOS and RWO on both AUC and AUPRC scores, showing a relatively good performance. The poor performance of SMOTE compared to the Baseline on the Credit Card Fraud, Pima Indians and German Credit Scoring data sets may be partially due to the underlying minority class distribution which may have been altered by SMOTE.

For example, there are clusters and small disjuncts in the Credit Card Fraud data set, which may create noisy examples and over-lapping of classes when SMOTE is applied, resulting in the degradation of performance. The German Credit Scoring data set has ordinal and categorical features and SMOTE does not do well with such features [49]. There is a possible dimensional impact of the data set which may be affecting SMOTE, especially for those data sets with many variables such as the Credit Card Fraud data set [11].

7.1.1.2 PDFOS

PDFOS appears worse than the Baseline on 4 of the 5 data sets for the AUPRC. Looking at the AUC, PDFOS is at least better on 4 of the data sets used. PDFOS performs worse than SMOTE on 3 data sets. For the German Credit Scoring data set, PDFOS is better than SMOTE whilst showing a poorer performance than the

Baseline. The German Credit Scoring data contains a number of categorical variables which both PDFOS and SMOTE may not be dealing with appropriately. In general, PDFOS appears as the worst over-sampling approach than SMOTE. This can be attributed to its rather strong Gaussian KDE which may not appropriate for the given variables.

7.1.1.3 RWO

RWO is worse than the Baseline and SMOTE even though it appears to be slightly better than PDFOS. RWO uses the CLT to approximate the minority class distribution, based on the mean and variance of each variable. PDFOS assumes a KDE of the entire data set using the Gaussian distribution. While the two approaches seem fundamentally similar, they are based on rather different distributional approaches. PDFOS can be said to be a non-parametric approach which requires a parameter estimation for the bandwidth used in the KDE [59].

RWO assumes that, theoretically, the distribution of each variable is Gaussian as the sample size increases. These are strong data assumptions which may not be entirely met, especially with skewed data sets such as the Credit Card Fraud, German Credit Scoring and Pima Indians Diabetes. As a result of possible inadequate distributions, overall, both RWO and PDFOS are showing poorer results than both SMOTE and the Baseline.

7.1.1.4 WCGAN-GP

Overall, WCGAN-GP has the highest AUPRC value and second highest AUC value. WCGAN-GP outperforms the Baseline in both the AUPRC and AUC scores. Using the AUPRC, WCGAN-GP outperforms SMOTE on 3 data sets which include Credit Card Fraud, Pima Indians Diabetes and German Credit Scoring. Overall, WCGAN-GP outperforms SMOTE.

The poor performance of WCGAN-GP compared to SMOTE on the Glass Identification and Breast Cancer Wisconsin data sets may be attributed to their slightly lower sample sizes compared to other data sets which have large sample sizes. GAN training requires a substantial training size to ensure a stable training and better sample quality [68, 66, 153, 7].

Compared to PDFOS and RWO, WCGAN-GP is better for all the other data sets when using the AUPRC. WCGAN-GP does not make any probability distribution assumptions but instead learns the true minority class distribution in order to create synthetic samples from it. This means that WCGAN-GP captures the probability distribution better than SMOTE, PDFOS and RWO, leading to a better predictive performance when the LR model is applied on the resulting synthesized data sets.

When using the AUC, WCGAN-GP is better than all the other methods on the German Credit Scoring, Glass Identification and Breast Cancer Wisconsin data sets. However, for the other 2 data sets, WCGAN-GP appears to be worse than SMOTE and its density-based variants.

Overall, using the AUC, SMOTE appears better than WCGAN-GP and all the density-based variants. When using the AUPRC, WCGAN-GP is better, followed by SMOTE. Below we provide insights as to which metric is appropriate to use for imbalanced learning.

7.1.2 AUC

The ROC curve represents the trade-off between Precision and the FP rate while the AUC is the area under the ROC curve [14]. Overall, SMOTE techniques report higher AUC values than the Baseline, suggesting that over-sampling improves the LR model. PDFOS appears as the worst method on the AUC score compared to other SMOTE density-based approaches. RWO is worse than SMOTE on 3 of the 5 data sets except for Pima Indians Diabetes and German Credit Scoring data sets.

In general, WCGAN-GP is better on 3 of the 5 data sets except on Credit card fraud and Diabetes data sets. Overall, the average AUC value is not too different between WCGAN-GP, SMOTE and RWO, while PDFOS is the lowest. This result conflicts the AUPRC scores where WCGAN-GP shows a clear dominant superiority over all the methods.

Whilst AUC may be useful, it does not consider Recall, which may be the most important metric for minority cases. AUC may be affected by skewed data sets and the data distribution [72]. ROC curves are appropriate when the data is balanced, whereas Precision-Recall curves are appropriate for imbalanced data sets [14, 72]. AUC may tend to provide an overly optimistic view than AUPRC [72], as can be seen by the results shown above, which seem to suggest that SMOTE is better.

In general, an algorithm that dominates in AUC may not necessarily dominate the AUPRC space [72]. Saito and Rehmsmeier [151] suggest that the Precision-Recall curve and AUPRC is more informative than the ROC curve and AUC. Since we are also comparing with the Baseline which is imbalanced, ROC and AUC may be inappropriate, thus AUPRC provides a sensible measure for all methods.

7.1.3 AUPRC

AUPRC has all the characteristics of the AUC and thus for the purposes of this work, we rely more on AUPRC than AUC [72, 151]. Overall, WCGAN-GP shows better improvements over all other SMOTE techniques. WCGAN-GP is highest on AUPRC, suggesting this algorithm performs the best across many thresholds and all the data sets used.

On the average, PDFOS and RWO do not provide a superior predictive performance than the Baseline. PDFOS appears as the worst approach for over-sampling. Below we further provide conclusive evidence on the statistical significance of the above results on the AUPRC.

7.2 Statistical Hypothesis Testing

Table 7.2 shows the results of the Friedman test applied on AUPRC to verify the statistical significance of WCGAN-GP compared to the other over-sampling methods. There is enough evidence at 5% significance level (since all p-values are less than 5%) to reject the null hypothesis on all the data sets, suggesting that over-sampling methods are not performing similarly and are different.

Data set	**P-value**	**Significance**
Credit Card Fraud	2.595112e-18	Yes
Pima Indians Diabetes	3.384100e-12	Yes
German Credit Scoring	2.021626e-05	Yes
Glass Identification	3.253634e-12	Yes
Breast Cancer Wisconsin	4.828195e-13	Yes

TABLE 7.2: Results for Friedman's test

Since the null hypothesis was rejected on all the data sets, a Post-hoc test was applied to further determine pairwise comparisons using the Nemenyi test where WCGAN-GP is the control method. These results confirm the significant superiority of WCGAN-GP over SMOTE as all the p-values are less than 0.05 for the data sets where Friedman suggested a difference.

There is a statistically significant superiority of WCGAN-GP over PDFOS except on the Glass Identification data set. WCGAN-GP is also superior and statistically different compared to RWO, except on the Glass Identification and Pima Indians Diabetes data sets.

Comparison Test	**Credit Card Fraud**	**German Credit Scoring**	**Breast Cancer Wisconsin**	**Glass Identification**	**Pima Indians Diabetes**
WCGAN-GP vs. SMOTE	**0.001000**	**0.001000**	**0.007510**	**0.001000**	**0.001000**
WCGAN-GP vs. PDFOS	**0.001000**	**0.001000**	**0.003000**	0.014361	**0.001000**
WCGAN-GP vs. RWO	**0.001000**	**0.001000**	**0.003000**	0.779980	0.22811
SMOTE vs. PDFOS	**0.002623**	0.059946	0.900000	**0.001000**	0.779980
SMOTE vs. RWO	**0.001000**	0.014361	0.836106	**0.001000**	**0.001000**
PDFOS vs. RWO	**0.001000**	0.187904	0.823993	0.153112	**0.001236**

TABLE 7.3: Results for the Post-hoc Nemenyi test

We observe a statistically significant difference between SMOTE techniques on the Credit Card Fraud data set. We observe no statistically significant differences between RWO and PDFOS on 3 data sets. We also note that RWO and PDFOS are not fundamentally too different in terms of the assumptions made as they both assume that the minority class data has a Gaussian distribution somehow.

Thus for the data sets that do not seem to exhibit numeric features that conform to this assumption, RWO and PDFOS do not seem to provide significantly too different results.

These results confirm the findings shown in Figure 7.1 and Table 7.1 where the average performance seen on both the AUC and AUPRC was lower for SMOTE techniques compared to WCGAN-GP. In general, WCGAN-GP provides statistically superior significant performance.

7.3 Discussion

Overall, SMOTE was worse than the Baseline on 3 of the 5 data sets when using the AUPRC. SMOTE samples synthetic points along line segments joining minority instances using the Euclidean distance. This approach may end up using majority instances and thus creating noisy examples and over-lapping cases [70, 135]. SMOTE is not based on the true distribution of the minority class data [38]. The poor performance of SMOTE on the AUPRC may be attributed to these effects, especially since the German Credit data set contains categorical variables which SMOTE may not be dealing with appropriately. Overall, SMOTE, RWO and PDFOS alter the data distribution as was observed by the significant compromise on Precision and generally lower F1-Score, AUPRC and AUC values. As a result, there was no statistically significant differences between the SMOTE techniques.

PDFOS and RWO are meant to improve the above SMOTE weaknesses. However, they both make strict assumptions about the structure and distribution of the minority class data. PDFOS assumes a KDE of the minority class using a multivariate Gaussian probability distribution [59]. PDFOS may not deal well with other non-continuous and multiple data structures such as the presence of categorical or ordinal variables [16, 59]. As a result, PDFOS was the worst over-sampling technique.

RWO was worse than both the Baseline and SMOTE although it appeared better than PDFOS. RWO uses CLT i.e. a Gaussian distribution and this approach is very similar to PDFOS. Thus the performance between PDFOS and RWO were not too dissimilar. RWO is only based on the minority class data and makes no assumptions and no pre-training is needed [179]. Thus run-times are way shorter than PDFOS. The lower RWO performance on the AUPRC than both SMOTE and Baseline may be attributed to the assumption made on the PDF of the minority class data which may not be appropriate. Furthermore, RWO requires the data to be continuous and this assumption may be invalid for categorical variables [35, 179].

Run-times for PDFOS appeared to be longer than SMOTE and RWO. PDFOS requires a rapid pre-training and computation of the co-variance matrix of the minority class data and determination of the bandwidth before over-sampling. The determination of h is performed via CV and this takes a while. SMOTE was the quickest to over-sample, followed by RWO and PDFOS and then WCGAN-GP.

WCGAN-GP requires a significant pre-training of both the critic and the generator. GANs are well-known for their training and computing powers [36, 106]. Thus they have expensive run-times. However, current GANs such as WGAN and WGAN-GP remedy this impact with stable training. The quality of generated samples may be worth it compared to the training times. GANs do not make explicit assumptions about the probability distribution of the minority class data. This idea has been used to create new samples for images [140], music [40], arts [88] and videos [60, 168, 175].

There is a significant potential to create new samples using GANs and augment imbalanced data sets. Recent work on this [41, 50] report a GAN superior performance over SMOTE but no mention is referenced on other SMOTE density-estimation approaches. This work is comprehensive and offering a distinct comparative study on density-estimation approaches. While GANs are notoriously difficult to train and optimise, in this study, using a simple architecture provided stable superior significant results after 5000 epochs.

Given the current surge in interest for GANs, optimising and training GANs is becoming straightforward as there are many implementations in Keras [52], Pytorch [133] and Tensorflow [163]. Given their impressive results and advancement in deep learning techniques, we expect a wider extensive use of GANs. The training instability of GANs will soon be done without any problems as the maturity of the training process improves with new techniques being invented at a rapid speed. Thus running times for GANs might not necessarily be an issue, forcing GANs to provide a superior over-sampling approach to supplement imbalanced data sets. Because GANs have become so popular, their limitations have been improved tremendously.

However, there are still open challenges for GANs. GANs rely on the generated examples being completely differentiable with respect to the generative parameters. As a result, GANs cannot product discrete data directly [79, 170]. Another key challenge is the evaluation of GANs after training even though there are measures to compute the quality of results generated [36]. Research for GANs grows each year. Practitioners may need to add GANs to their toolkit as this will significantly improve their models and aid on decision-making as GANs will be characterised by advancements in deep learning, training process maturity, open acceptance and their wide use in commercial applications.

8 Conclusions and Future Research

This section concludes this work, gives limitations and provides scope for future research.

8.1 Conclusions

This work detailed a class imbalance study on imbalanced data sets where SMOTE density estimation approaches and WCGAN-GP were used to over-sample the minority cases on 5 imbalanced data sets. A LR model was trained on the baseline and over-sampled data sets and the results were compared using Precision, Recall, F1-Score, AUPRC and AUC. SMOTE improved the classification performance. However, SMOTE is not based on the true underlying minority class distribution. SMOTE density estimation approaches remedy this issue, however, these techniques make assumptions around the minority class distribution. As a result, both PDFOS and RWO performed poorly than SMOTE and the Baseline. WCGAN-GP was statistically better than all the SMOTE techniques on the majority of the data sets.

PDFOS and RWO results were not significantly too dissimilar to SMOTE results on 3 of the data sets. Thus there were no statistically significant differences between SMOTE, RWO and PDFOS on 3 of the experimental data sets used. Using WCGAN-GP, it is possible to create synthetic cases implicitly and this turned out to offer a significantly better improvement over all SMOTE techniques, across various thresholds and on 3 of the data sets used. AUPRC appeared as a more sensible informative measure to compare the algorithms.

This work has demonstrated the potential for GANs for data augmentation and boosting predictive models. There are other useful areas where GANs are becoming more useful such as anomaly detection [2, 154], semi-supervised learning [31,

103, 121, 159], domain adaptation [80, 81, 165], time series generation [45, 57], privacy preservation [13], joint distribution learning [33, 102, 174, 184], reinforcement learning [158], missing data imputation [97, 155, 176] and many other computer vision areas [18, 90, 96, 180].

8.2 Limitations

This work considered binary cases whereas other data sets may have multiple classes. We repeated training and testing of each over-sampling method 30 times to minimise stochastic effects - this sample size can be increased for more robustness. Alternatively a bootstrapping approach can be applied to better understand the distributional attributes of the model errors.

There were mixed results when using AUC and AUPRC. Existing literature has no consensus on which metric to prefer, despite most studies using AUC. Other evaluation metrics exist and these can offer a different and comprehensive perspective. These include Partial AUC [115], Weighted AUC [171], Discriminative Power [5], Matthews correlation coefficient (MCC) [39], Gain and Lifts charts [24]. A further comprehensive study would be to utilise some of these metrics and statistically evaluate the performance of the over-sampling methods.

8.3 Future Research

Possible future research to improve this work includes a consideration on other data sets to apply the same methods, especially complex data sets that include small disjuncts, over-lapping, mixed data types and multiple classes. The results could be repeated by varying the imbalanced ratio to determine which technique performs the best depending on the extent of imbalance observed in the data set. We could consider other ML algorithms such as ANN and Gradient Boosting Machines [29]. An empirical comparison of these results with other tabular data sets where GAN was applied would be useful. New Adam variants were recently proposed called Rectified Adam (RAdam) [101], AMSGrad [144] and LookAhead or Ranger [181], which seem to show better results for GAN training. Other loss variants and advanced architectures such as BEGAN, EBGAN, DRAGAN, LSGAN and VAE-GAN could be explored for better GAN training.

Bibliography

[1] D.H. Ackley, G.E. Hinton, and T.J. Sejnowski. "A learning algorithm for Boltzmann machines". In: *Cognitive Science* 9.1 (1985), pp. 147–169.

[2] S. Akcay, A. Atapour-Abarghouei, and T.P. Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training". In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 622–637.

[3] J. Alcalá-Fdez et al. "Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework." In: *Journal of Multiple-Valued Logic & Soft Computing* 17 (2011).

[4] B.A. Almogahed and I.A. Kakadiaris. "NEATER: Filtering of Over-sampled Data Using Non-cooperative Game Theory". In: *2014 22nd International Conference on Pattern Recognition*. 2014, pp. 1371–1376.

[5] A. An, N. Cercone, and X. Huang. "A case study for learning from imbalanced data sets". In: *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer. 2001, pp. 1–15.

[6] G. Antipov, M. Baccouche, and J. Dugelay. "Face aging with conditional generative adversarial networks". In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, pp. 2089–2093.

[7] M. Arjovsky, S. Chintala, and L. Bottou. "Wasserstein GAN". In: *arXiv preprint arXiv:1701.07875* (2017).

[8] K. Armanious et al. "MedGAN: Medical image translation using GANs". In: *arXiv preprint arXiv:1806.06397* (2018).

[9] L. Aviñó, M. Ruffini, and R. Gavaldà. "Generating Synthetic but Plausible Healthcare Record Datasets". In: *arXiv preprint arXiv:1807.01514* (2018).

[10] S. Barua, M.M. Islam, and K. Murase. "A novel synthetic minority oversampling technique for imbalanced data set learning". In: *International Conference on Neural Information Processing*. Springer. 2011, pp. 735–744.

[11] S. Barua et al. “MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 26.2 (2014), pp. 405–425.

[12] G.E. Batista, R.C. Prati, and M.C. Monard. “A study of the behavior of several methods for balancing machine learning training data”. In: *ACM SIGKDD explorations newsletter* 6.1 (2004), pp. 20–29.

[13] B.K. Beaulieu-Jones et al. “Privacy-preserving generative deep neural networks support clinical data sharing”. In: *Circulation: Cardiovascular Quality and Outcomes* 12.7 (2019), e005122.

[14] M. Bekkar, H. Kheliouane Djemaa, and Taklit A. A. “Evaluation measures for models assessment over imbalanced data sets”. In: *Journal of Information Engineering and Applications* 3.10 (2013).

[15] C. Bellinger, C. Drummond, and N. Japkowicz. “Beyond the Boundaries of SMOTE”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2016, pp. 248–263.

[16] 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)*. IEEE. 2015, pp. 948–953.

[17] Y. Bengio et al. “Deep generative stochastic networks trainable by backprop”. In: *International Conference on Machine Learning*. 2014, pp. 226–234.

[18] U. Bergmann, N. Jetchev, and R. Vollgraf. “Learning texture manifolds with the periodic spatial GAN”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 469–477.

[19] D. Berthelot, T. Schumm, and L. Metz. “Began: Boundary equilibrium generative adversarial networks”. In: *arXiv preprint arXiv:1703.10717* (2017).

[20] A.P. Bradley. “The use of the area under the ROC curve in the evaluation of machine learning algorithms”. In: *Pattern Recognition* 30.7 (1997), pp. 1145–1159.

[21] A. Brock, J. Donahue, and K. Simonyan. “Large scale gan training for high fidelity natural image synthesis”. In: *arXiv preprint arXiv:1809.11096* (2018).

[22] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. "DBSMOTE: density-based synthetic minority over-sampling technique". In: *Applied Intelligence* 36.3 (2012), pp. 664–684.

[23] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem". In: *Pacific-Asia conference on Knowledge Discovery and Data Mining*. Springer. 2009, pp. 475–482.

[24] J. Burez and D. Van den Poel. "Handling class imbalance in customer churn prediction". In: *Expert Systems with Applications* 36.3 (2009), pp. 4626–4636.

[25] R. Camino, C. Hammerschmidt, and R. State. "Generating Multi-Categorical Samples with Generative Adversarial Networks". In: *arXiv preprint arXiv:1807.01202* (2018).

[26] N.V. Chawla, N. Japkowicz, and A. Kotcz. "Special issue on learning from imbalanced data sets". In: *ACM Sigkdd Explorations Newsletter* 6.1 (2004), pp. 1–6.

[27] N.V. Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[28] Z. Che et al. "Boosting deep learning risk prediction with generative adversarial networks for electronic health records". In: *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2017, pp. 787–792.

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

[30] X. Chen et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets". In: *Advances in Neural Information Processing systems*. 2016, pp. 2172–2180.

[31] L. Chongxuan et al. "Triple generative adversarial nets". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4088–4098.

[32] C. Chow and T. Wagner. "Consistency of an estimate of tree-dependent probability distributions (corresp.)" In: *IEEE Transactions on Information Theory* 19.3 (1973), pp. 369–371.

[33] C. Chu, A. Zhmoginov, and M. Sandler. "Cyclegan, a master of steganography". In: *arXiv preprint arXiv:1712.02950* (2017).

[34] D. Clevert, T. Unterthiner, and S. Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)". In: *arXiv preprint arXiv:1511.07289* (2015).

[35] I. Cordón et al. "Imbalance: oversampling algorithms for imbalanced classification in R". In: *Knowledge-Based Systems* 161 (2018), pp. 329–341.

[36] A. Creswell et al. "Generative adversarial networks: An overview". In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 53–65.

[37] A. Dal Pozzolo. "Adaptive machine learning for credit card fraud detection". In: (2015).

[38] B. Das, N.C. Krishnan, and D.J. Cook. "RACOG and wRACOG: Two probabilistic oversampling techniques". In: *IEEE transactions on knowledge and data engineering* 27.1 (2015), pp. 222–234.

[39] Z. Ding. "Diversified ensemble classifiers for highly imbalanced data learning and its application in bioinformatics". PhD thesis. Georgia State University, 2011.

[40] H. Dong et al. "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[41] G. Douzas and F. Bacao. "Effective data generation for imbalanced learning using conditional generative adversarial networks". In: *Expert Systems with applications* 91 (2018), pp. 464–471.

[42] G. Douzas, F. Bacao, and F. Last. "Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE". In: *Information Sciences* 465 (2018), pp. 1–20.

[43] T. Dozat. "Incorporating nesterov momentum into adam". In: *ICLR Workshop*. Vol. 1. 2013. 2016, p. 2016.

[44] J. Duchi, E. Hazan, and Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *Journal of Machine Learning Research* 12.Jul (2011), pp. 2121–2159.

[45] C. Esteban, S.L. Hyland, and G. Rätsch. "Real-valued (medical) time series generation with recurrent conditional gans". In: *arXiv preprint arXiv:1706.02633* (2017).

[46] M. Ester et al. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *Kdd*. Vol. 96. 34. 1996, pp. 226–231.

[47] I.W. Evett and E.J. Spiehler. "Rule induction in forensic science". In: *KBS in Goverment* (1987), pp. 107–118.

[48] A. Fernández, S. García, and F. Herrera. "Addressing the classification with imbalanced data: open problems and new challenges on class distribution". In: *International Conference on Hybrid Artificial Intelligence Systems*. Springer. 2011, pp. 1–10.

[49] A. Fernández et al. "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary". In: *Journal of Artificial Intelligence Research* 61 (2018), pp. 863–905.

[50] U. Fiore et al. "Using Generative Adversarial Networks for improving classification effectiveness in Credit card fraud detection". In: *Information Sciences* (2017).

[51] Python Software Foundation. "Python Language Reference (Version 3.6. 3.)" In: (2017).

[52] C. François. *keras*. https://github.com/fchollet/keras. 2015.

[53] B.J. Frey, G.E. Hinton, and P. Dayan. "Does the wake-sleep algorithm produce good density estimators?" In: *Advances in Neural Information Processing Systems*. 1996, pp. 661–667.

[54] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements Of Statistical Learning*. Vol. 1. Springer series in statistics New York, 2001.

[55] M. Friedman. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance". In: *Journal of the American Statistical Association* 32.200 (1937), pp. 675–701.

[56] N. Friedman, D. Geiger, and M. Goldszmidt. "Bayesian network classifiers". In: *Machine learning* 29.2-3 (1997), pp. 131–163.

[57] R. Fu et al. “Time Series Simulation by Conditional Generative Adversarial Net”. In: *arXiv preprint arXiv:1904.11419* (2019).

[58] V. Ganganwar. “An overview of classification algorithms for imbalanced datasets”. In: *International Journal of Emerging Technology and Advanced Engineering* 2.4 (2012), pp. 42–47.

[59] M. Gao et al. “PDFOS: PDF estimation based over-sampling for imbalanced two-class problems”. In: *Neurocomputing* 138 (2014), pp. 248–259.

[60] J. Gauthier. “Conditional generative adversarial nets for convolutional face generation”. In: *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester* 2014.5 (2014), p. 2.

[61] D. Georgios and B. Fernando. “Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning”. In: *Expert Systems with Applications* 82 (2017), pp. 40 –52. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2017.03.073. URL: http://www.sciencedirect.com/science/article/pii/S0957417417302324.

[62] C.J. Geyer. “Practical markov chain monte carlo”. In: *Statistical science* (1992), pp. 473–483.

[63] C. Gilles et al. “Learning from imbalanced data in surveillance of nosocomial infection”. In: *Artificial Intelligence in Medicine* 37.1 (2006), pp. 7 –18.

[64] X. Glorot and Y. Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

[65] X. Glorot, A. Bordes, and Y. Bengio. “Deep sparse rectifier neural networks”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011, pp. 315–323.

[66] I. Goodfellow. “NIPS 2016 tutorial: Generative adversarial networks”. In: *arXiv preprint arXiv:1701.00160* (2016).

[67] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Vol. 1. MIT press Cambridge, 2016.

[68] I. Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 2672–2680.

[69] I. Gulrajani et al. “Improved training of Wasserstein gans”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 5767–5777.

[70] H. Han, W. Wang, and B. Mao. “Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning”. In: *International Conference on Intelligent Computing*. Springer. 2005, pp. 878–887.

[71] J.A. Hanley and B.J. McNeil. “The meaning and use of the area under a receiver operating characteristic (ROC) curve”. In: *Radiology* 143.1 (1982), pp. 29–36.

[72] H. He and E.A. Garcia. “Learning from imbalanced data”. In: *IEEE Transactions on Knowledge & Data Engineering* 9 (2008), pp. 1263–1284.

[73] H. He et al. “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”. In: *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE. 2008, pp. 1322–1328.

[74] K. He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[75] G.E. Hinton. “Training products of experts by minimizing contrastive divergence”. In: *Neural computation* 14.8 (2002), pp. 1771–1800.

[76] G.E Hinton, S. Osindero, and Y. Teh. “A fast learning algorithm for deep belief nets”. In: *Neural Computation* 18.7 (2006), pp. 1527–1554.

[77] G.E. Hinton and R.R. Salakhutdinov. “Reducing the dimensionality of data with neural networks”. In: *science* 313.5786 (2006), pp. 504–507.

[78] G.E. Hinton and T. Tieleman. *Lecture 6.5 - RMSProp, COURSERA: Neural Networks for Machine Learning*. https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. 2012.

[79] S. Hitawala. “Comparative study on generative adversarial networks”. In: *arXiv preprint arXiv:1801.04271* (2018).

[80] J. Hoffman et al. “Cycada: Cycle-consistent adversarial domain adaptation”. In: *arXiv preprint arXiv:1711.03213* (2017).

[81] W. Hong et al. “Conditional generative adversarial network for structured domain adaptation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1335–1344.

[82] Y. Hong et al. “How Generative Adversarial Networks and Their Variants Work: An Overview”. In: *ACM Computing Surveys (CSUR)* 52.1 (2019), p. 10.

[83] H. Hotelling. “Analysis of a complex of statistical variables into principal components.” In: *Journal of Educational Psychology* 24.6 (1933), pp. 417–441.

[84] S. Ioffe and C. Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[85] N. Japkowicz. “The class imbalance problem: Significance and strategies”. In: *Proceedings of the International Conference on Artificial Intelligence*. 2000.

[86] N. Japkowicz and S. Stephen. “The class imbalance problem: A systematic study”. In: *Intelligent Data Analysis* 6.5 (2002), pp. 429–449.

[87] K. Jiang, J. Lu, and K. Xia. “A Novel Algorithm for Imbalance Data Classification Based on Genetic Algorithm Improved SMOTE”. In: *Arabian Journal for Science and Engineering* 41.8 (2016), pp. 3255–3266.

[88] Y. Jin et al. “Towards the automatic anime characters creation with generative adversarial networks”. In: *arXiv preprint arXiv:1708.05509* (2017).

[89] A. Jolicoeur-Martineau. “The relativistic discriminator: a key element missing from standard GAN”. In: *arXiv preprint arXiv:1807.00734* (2018).

[90] T. Karras et al. “Progressive growing of gans for improved quality, stability, and variation”. In: *arXiv preprint arXiv:1710.10196* (2017).

[91] D.P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[92] D.P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[93] S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[94] S. Kullback and R.A. Leibler. “On information and sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[95] A.B.L. Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”. In: *arXiv preprint arXiv:1512.09300* (2015).

[96] C. Ledig et al. “Photo-realistic single image super-resolution using a generative adversarial network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690.

[97] S.C. Li, B. Jiang, and B. Marlin. “Misgan: Learning from incomplete data with generative adversarial networks”. In: *arXiv preprint arXiv:1902.09599* (2019).

[98] Y. Li, K. Swersky, and R. Zemel. “Generative moment matching networks”. In: *International Conference on Machine Learning*. 2015, pp. 1718–1727.

[99] J.H. Lim and J.C. Ye. “Geometric gan”. In: *arXiv preprint arXiv:1705.02894* (2017).

[100] J. Lin. “Divergence measures based on the Shannon entropy”. In: *IEEE Transactions on Information theory* 37.1 (1991), pp. 145–151.

[101] L. Liu et al. “On the variance of the adaptive learning rate and beyond”. In: *arXiv preprint arXiv:1908.03265* (2019).

[102] M. Liu and O. Tuzel. “Coupled generative adversarial networks”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 469–477.

[103] Z. Liu, J. Wang, and Z. Liang. “CatGAN: Category-aware Generative Adversarial Networks with Hierarchical Evolutionary Learning for Category Text Generation”. In: *arXiv preprint arXiv:1911.06641* (2019).

[104] R. Longadge and S. Dongre. “Class imbalance problem in data mining review”. In: *arXiv preprint arXiv:1305.1707* (2013).

[105] V. López et al. “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics”. In: *Information sciences* 250 (2013), pp. 113–141.

[106] M. Lucic et al. “Are gans created equal? a large-scale study”. In: *Advances in Neural Information Processing systems*. 2018, pp. 700–709.

[107] A.L. Maas, A.Y. Hannun, and A.Y. Ng. “Rectifier nonlinearities improve neural network acoustic models”. In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[108] L. Maaten and G. Hinton. "Visualizing data using t-SNE". In: *Journal of Machine Learning Research* 9.Nov (2008), pp. 2579–2605.

[109] L. Van der Maaten and G. Hinton. "Visualizing non-metric similarities in multiple maps". In: *Journal of Machine Learning* 87.1 (2012), pp. 33–55.

[110] A. Makhzani et al. "Adversarial autoencoders". In: *arXiv preprint arXiv:1511.05644* (2015).

[111] P. Manisha and S. Gujar. "Generative Adversarial Networks (GANs): What it can generate and What it cannot?" In: *arXiv preprint arXiv:1804.00140* (2018).

[112] X. Mao et al. "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2794–2802.

[113] G. Mariani et al. "Bagan: Data Augmentation with Balancing GAN". In: *arXiv preprint arXiv:1803.09655* (2018).

[114] J. Mathew et al. "Kernel-based SMOTE for SVM classification of imbalanced datasets". In: *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*. IEEE. 2015, pp. 1127–1132.

[115] D.K. McClish. "Analyzing a portion of the ROC curve". In: *Medical Decision Making* 9.3 (1989), pp. 190–195.

[116] P. McCullagh. "Generalized linear models". In: *European Journal of Operational Research* 16.3 (1984), pp. 285–292.

[117] N. Metropolis et al. "Equation of state calculations by fast computing machines". In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[118] M. Mirza and S. Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[119] D. Misra. "Mish: A Self Regularized Non-Monotonic Neural Activation Function". In: *arXiv preprint arXiv:1908.08681* (2019).

[120] T.M. Mitchell. *The Discipline of Machine Learning*. Vol. 9. Carnegie Mellon University, School of Computer Science, Machine Learning Department, 2006.

[121] T. Miyato et al. "Virtual adversarial training: a regularization method for supervised and semi-supervised learning". In: *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018), pp. 1979–1993.

[122] A. Mottini, A. Lheritier, and R. Acuna-Agost. "Airline passenger name record generation using generative adversarial networks". In: *arXiv preprint arXiv:1807.06657* (2018).

[123] A. Müller. "Integral probability metrics and their generating classes of functions". In: *Advances in Applied Probability* 29.2 (1997), pp. 429–443.

[124] S.S. Mullick, S. Datta, and S. Das. "Generative Adversarial Minority Oversampling". In: *arXiv preprint arXiv:1903.09730* (2019).

[125] V. Nair and G.E. Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[126] I. Nekooeimehr and S.K. Lai-Yuen. "Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets". In: *Expert Systems with Applications* 46 (2016), pp. 405–416.

[127] P. Nemenyi. "Distribution-free multiple comparisons". In: *Biometrics*. Vol. 18. 2. Princeton University. 1962, p. 263.

[128] Y. Nesterov. "A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2)". In: *Doklady AN USSR*. Vol. 269. 1983, pp. 543–547.

[129] S. Nowozin, B. Cseke, and R. Tomioka. "f-gan: Training generative neural samplers using variational divergence minimization". In: *Advances in Neural Information Processing systems*. 2016, pp. 271–279.

[130] A. Odena, C. Olah, and J. Shlens. "Conditional image synthesis with auxiliary classifier gans". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 2642–2651.

[131] A. Van den Oord et al. "Conditional image generation with pixelcnn decoders". In: *Advances in neural information processing systems*. 2016, pp. 4790–4798.

[132] E. Parzen. "On estimation of a probability density function and mode". In: *The Annals of Mathematical Statistics* 33.3 (1962), pp. 1065–1076.

[133] A. Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[134] N. Patki, R. Wedge, and K. Veeramachaneni. "The synthetic data vault". In: *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE. 2016, pp. 399–410.

[135] M. Pérez-Ortiz, P.A. Gutiérrez, and C. Hervás-Martínez. "Borderline kernel based over-sampling". In: *International Conference on Hybrid Artificial Intelligence Systems*. Springer. 2013, pp. 472–481.

[136] T. Pohlert. "The pairwise multiple comparison of mean ranks package (PMCMR)". In: *R package* 27 (2014).

[137] K. Potdar, T.S. Pardawala, and C.D. Pai. "A comparative study of categorical variable encoding techniques for neural network classifiers". In: *International journal of computer applications* 175.4 (2017), pp. 7–9.

[138] G. Qi. "Loss-sensitive generative adversarial networks on lipschitz densities". In: *arXiv preprint arXiv:1701.06264* (2017).

[139] N. Qian. "On the momentum term in gradient descent learning algorithms". In: *Neural networks* 12.1 (1999), pp. 145–151.

[140] A. Radford, L. Metz, and S. Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[141] A.E. Raftery and S. Lewis. "How many iterations in the Gibbs sampler". In: *Bayesian Statistics* 4.2 (1992), pp. 763–773.

[142] P. Ramachandran, B. Zoph, and Q.V. Le. "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941* (2017).

[143] E. Ramentol et al. "SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory". In: *Knowledge and Information Systems* 33.2 (2012), pp. 245–265.

[144] S.J. Reddi, S. Kale, and S. Kumar. "On the convergence of adam and beyond". In: *arXiv preprint arXiv:1904.09237* (2019).

[145] S. Reed et al. "Generative adversarial text to image synthesis". In: *arXiv preprint arXiv:1605.05396* (2016).

[146] D.J. Rezende, S. Mohamed, and D. Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *arXiv preprint arXiv:1401.4082* (2014).

[147] Y. Rubner, C. Tomasi, and L.J. Guibas. "The earth mover's distance as a metric for image retrieval". In: *International Journal of Computer Vision* 40.2 (2000), pp. 99–121.

[148] S. Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[149] D.E Rumelhart, G.E. Hinton, and R.J. Williams. "Learning Representations by Back-propagating Errors". In: *Nature* 323.6088 (1986), p. 533.

[150] Y. Saatci and A.G. Wilson. "Bayesian gan". In: *Advances in Neural Information Processing systems*. 2017, pp. 3622–3631.

[151] T. Saito and M. Rehmsmeier. "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets". In: *PloS one* 10.3 (2015), e0118432.

[152] R. Salakhutdinov and G. Hinton. "Deep boltzmann machines". In: *Artificial Intelligence and Statistics*. 2009, pp. 448–455.

[153] T. Salimans et al. "Improved techniques for training GANs". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2234–2242.

[154] T. Schlegl et al. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery". In: *International Conference on Information Processing in Medical Imaging*. Springer. 2017, pp. 146–157.

[155] C. Shang et al. "VIGAN: Missing view imputation with generative adversarial networks". In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE. 2017, pp. 766–775.

[156] B.W. Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[157] J.W. Smith et al. “Using the ADAP learning algorithm to forecast the onset of diabetes mellitus”. In: *Proceedings of the Annual Symposium on Computer Application in Medical Care*. American Medical Informatics Association. 1988, pp. 261–265. URL: \url{https : www . kaggle . com / uciml / pima - indians - diabetes-database}.

[158] J. Song et al. “Multi-agent generative adversarial imitation learning”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 7461–7472.

[159] K. Sricharan et al. “Semi-supervised conditional gans”. In: *arXiv preprint arXiv:1708.05789* (2017).

[160] N. Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[161] W.N. Street, W.H. Wolberg, and O.L. Mangasarian. “Nuclear feature extraction for breast tumor diagnosis”. In: *Biomedical image processing and biomedical visualization*. Vol. 1905. International Society for Optics and Photonics. 1993, pp. 861–870.

[162] Y. Sun, A. Cuesta-Infante, and K. Veeramachaneni. “Learning Vine Copula Models for Synthetic Data Generation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 5049–5057.

[163] TensorFlow Team. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from http : //www . tensorflow . org. 2015. URL: http://www.tensorflow.org.

[164] R. Tibshirani. “Regression shrinkage and selection via the lasso”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[165] E. Tzeng et al. “Adversarial discriminative domain adaptation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7167–7176.

[166] P. Vincent et al. “Extracting and composing robust features with denoising autoencoders”. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 1096–1103.

[167] P. Vincent et al. “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion”. In: *Journal of Machine Learning Research* 11.Dec (2010), pp. 3371–3408.

[168] C. Vondrick, H. Pirsiavash, and A. Torralba. “Generating videos with scene dynamics”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 613–621.

[169] E.M. Voorhees. “Implementing agglomerative hierarchic clustering algorithms for use in document retrieval”. In: *Information Processing & Management* 22.6 (1986), pp. 465–476.

[170] K. Wang et al. “Generative adversarial networks: introduction and outlook”. In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), pp. 588–598.

[171] C.G. Weng and J. Poon. “A new evaluation measure for imbalanced datasets”. In: *Proceedings of the 7th Australasian Data Mining Conference-Volume 87*. Australian Computer Society, Inc. 2008, pp. 27–32.

[172] Z. Xie et al. “A synthetic minority oversampling method based on local densities in low-dimensional space for imbalanced learning”. In: *International Conference on Database Systems for Advanced Applications*. Springer. 2015, pp. 3–18.

[173] L. Yang, S. Chou, and Y. Yang. “MidiNet: A convolutional generative adversarial network for symbolic-domain music generation”. In: *arXiv preprint arXiv:1703.10847* (2017).

[174] Z. Yi et al. “Dualgan: Unsupervised dual learning for image-to-image translation”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2849–2857.

[175] D. Yoo et al. “Pixel-level domain transfer”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 517–532.

[176] J. Yoon, J. Jordon, and M. Van Der Schaar. “Gain: Missing data imputation using generative adversarial nets”. In: *arXiv preprint arXiv:1806.02920* (2018).

[177] M.D. Zeiler. “ADADELTA: an adaptive learning rate method”. In: *arXiv preprint arXiv:1212.5701* (2012).

[178] Z.Q. Zeng et al. “A classification method for imbalance data set based on kernel SMOTE”. In: *Acta Electronica Sinica* 37.11 (2009), pp. 2489–2495.

[179] H. Zhang and M. Li. “RWO-Sampling: A Random Walk Over-Sampling Approach to Imbalanced Data Classification”. In: 20 (Nov. 2014).

[180] H. Zhang et al. “Self-attention generative adversarial networks”. In: *arXiv preprint arXiv:1805.08318* (2018).

[181] M.R. Zhang et al. “Lookahead Optimizer: k steps forward, 1 step back”. In: *arXiv preprint arXiv:1907.08610* (2019).

[182] J. Zhao, M. Mathieu, and Y. LeCun. “Energy-based generative adversarial network”. In: *arXiv preprint arXiv:1609.03126* (2016).

[183] Y. Zheng et al. “Generative adversarial network based telecom fraud detection at the receiving bank”. In: *Neural Networks* 102 (2018), pp. 78–86.

[184] J. Zhu et al. “Unpaired image-to-image translation using cycle-consistent adversarial networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.