2024

Bayesian Variable Selection with Shrinkage Priors and Generative Adversarial Networks for Fraud Detection

Amina Issoufou Anaroua
University of Central Florida

BAYESIAN VARIABLE SELECTION WITH SHRINKAGE PRIORS AND GENERATIVE ADVERSARIAL NETWORKS FOR FRAUD DETECTION

by

AMINA ISSOUFOU ANAROUA
B.S Embry Riddle Aeronautical University, 2019
M.S Embry Riddle Aeronautical University, 2022

A thesis submitted in partial fulfilment of the requirements
for the degree of Master of Science
in the Department of Statistics and Data Science
in the College of Science
at the University of Central Florida
Orlando, Florida

Spring Term
2024

Major Professor: Hsin-Hsiung Huang

ABSTRACT

This research paper focuses on fraud detection in the financial industry using Generative Adversarial Networks (GANs) in conjunction with Uni and Multi Variate Bayesian Model with Shrinkage Priors (BMSP). The problem addressed is the need for accurate and advanced fraud detection techniques due to the increasing sophistication of fraudulent activities. The methodology involves the implementation of GANs and the application of BMSP for variable selection to generate synthetic fraud samples for fraud detection using the augmented dataset. Experimental results demonstrate the effectiveness of the BMSP GAN approach in detecting fraud with improved performance compared to other methods. The conclusions drawn highlight the potential of GANs and BMSP for enhancing fraud detection capabilities and suggest future research directions for further improvements in the field.

To my family, for their endless love and unwavering support throughout this journey.

ACKNOWLEDGMENTS

I would like to extend my deepest gratitude to Dr. Hsin-Hsiung Huang, under whose expert guidance I have had the privilege of working for the past one year. His unwavering support and insightful direction have been instrumental and fundamental to the completion of this study.

Special appreciation goes to Dr. Richard Ajayi for his invaluable suggestions and meticulous analysis. Their contributions have significantly shaped my research journey.

I am also deeply indebted to Dr. Edgard Maboudou for his consistent assistance and thoughtful advice throughout this work. His encouragement and support have been a guiding light in my academic endeavors.

My sincere thanks are extended to the staff and professors in the Department of Statistics and Data Science. The education and mentorship I received there have been pivotal in my academic growth.

A special mention of my family is warranted here, especially my mother, whose unwavering support has been my backbone. Her sacrifices and encouragement made it possible for me to pursue my studies abroad. My journey would not have been the same without the strength and support of my entire family.

Lastly, I want to acknowledge and thank everyone who has provided support, encouragement, and inspiration. This includes all those who have contributed, in ways big and small, to making this endeavor a reality. Your collective support has been invaluable, and I am truly grateful.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Fraud detection plays a crucial role in the financial industry, where the consequences of fraudulent activities can be severe for both individuals and businesses. With the rise of digital transactions and the increasing sophistication of fraudsters, accurately detecting and preventing fraudulent transactions has become more challenging than ever before. The impact of fraud extends beyond financial losses, encompassing reputational damage, compromised personal information, and a loss of trust in financial systems.

Statistics reveal the alarming prevalence of fraud in today's society. According to a recent report by the Federal Trade Commission (FTC), there were over 4.7 million reports of fraud in the United States alone in 2022, resulting in approximately 3.3 billion dollars in losses for individuals. These figures highlight the urgent need for effective fraud detection mechanisms that can protect individuals and businesses from falling victim to fraudulent activities.

However, detecting fraud poses several significant challenges. One of the major obstacles is the inherent imbalance in fraud data. Legitimate transactions far outnumber fraudulent ones, making it difficult for traditional fraud detection algorithms to accurately identify fraudulent patterns. This data imbalance often leads to a skewed learning process, where the algorithm predominantly focuses on the majority class, resulting in poor fraud detection performance.

To address this issue, machine learning techniques, particularly Generative Adversarial Networks (GANs), have emerged as a promising approach. GANs have the ability to generate synthetic fraud samples by capturing the complex data distribution of fraudulent activities. By augmenting the training data with these synthetic samples, the imbalance problem can be mitigated, enabling fraud detection models to learn from a more balanced dataset and improve their performance in identifying fraudulent transactions.

This research integrates Generative Adversarial Networks (GANs) and BMSP (Uni and Multi Variate Bayesian Model with Shrinkage Priors) for advanced fraud detection in the financial sector. We address the challenge of imbalanced datasets in fraud data, where legitimate transactions significantly outnumber fraudulent ones, hindering traditional fraud detection algorithms. The study utilizes GANs to generate synthetic fraud samples, thereby enhancing the balance of datasets for improved model performance. Additionally, BMSP, adapted for variable selection, aids in refining the GANs' synthetic data generation by selecting the most relevant variables from a pool of predictors. These selected variables are then integrated into the GANs' generator model to generate synthetic fraud samples. This careful selection ensures that the generated data contains the most informative features for fraud detection, leading to more accurate predictions. This research also explores the BMSP model's classification performance on the RNA-Seq (HiSeq) PANCAN dataset, involving gene expression data for various tumor types, further emphasizing its versatility and potential in diverse data-intensive domains. This approach aims not only to enhance fraud detection in the financial sector but also demonstrates BMSP's versatility in classifying complex biological data.

The objective of this study is twofold. Firstly, to investigate the effectiveness of GANs in generating synthetic fraud samples. Secondly, to explore the integration of Bayesian models to further enhance the accuracy and reliability of fraud detection systems. To evaluate the proposed approach, real-world financial datasets containing both genuine and fraudulent transactions will be utilized.

By conducting this research, we aim to provide valuable insights into the challenges, limitations, and potential future research directions in fraud detection using GANs and Bayesian models. The findings of this study will contribute to the existing body of knowledge by offering a comprehensive analysis of the effectiveness and potential of these techniques in detecting fraudulent activities. Additionally, it will underscore the importance of addressing data imbalance and selecting relevant variables when generating synthetic data for fraud detection applications.

Fraud detection in the financial industry has been a topic of significant interest, resulting in the development of various methods and techniques to identify and prevent fraudulent activities. Traditional statistical approaches, such as rule-based methods and anomaly detection, have long been utilized for fraud detection. These methods rely on predefined rules or statistical thresholds to flag transactions that deviate from normal patterns [10]. While these techniques can be effective in some cases, they often struggle to adapt to evolving and sophisticated fraud schemes, leading to high false positive rates and missed detections.

In recent years, machine learning techniques have gained prominence for fraud detection due to their ability to learn from data and detect complex patterns. One popular approach is the use of supervised machine learning algorithms, such as logistic regression and support vector machines, which learn from labeled data to classify transactions as fraudulent or legitimate [3]. While these methods can achieve good accuracy, they also rely heavily on having a balanced dataset, which is often not the case in fraud detection, leading to sub-optimal performance.

The imbalance in fraud data has led to the exploration of more advanced machine-learning techniques for fraud detection, such as ensemble methods, random forests [1], and gradient boosting. These techniques attempt to mitigate the impact of data imbalance by combining multiple models or leveraging sampling techniques like SMOTE [2]. However, while they show promise, they may still struggle to effectively handle highly imbalanced data and capture the complex patterns of fraud.

As the field of machine learning advances, Generative Adversarial Networks (GANs) have emerged as a powerful tool for various tasks, including generating synthetic data [6]. GANs consist of two neural networks, the generator, and the discriminator, engaged in a game-like scenario, where the generator attempts to produce realistic synthetic data, and the discriminator tries to differentiate between genuine and synthetic data. In the context of fraud detection, GANs have shown promise

in generating synthetic fraud samples, thus addressing the data imbalance problem and enhancing the performance of fraud detection models.

CHAPTER 2: LITERATURE REVIEW

In recent years, the application of Generative Adversarial Networks (GANs) in fraud detection has gained significant attention. GANs, introduced by [6], represent a novel method in machine learning for generating synthetic data, which is particularly useful in scenarios where data is imbalanced or scarce. The effectiveness of GANs in generating synthetic financial transaction data was demonstrated by [4], underscoring their potential in addressing privacy concerns and data scarcity in the fraud detection domain.

The challenge of data imbalance in fraud detection has been a critical issue. Traditional fraud detection methods often suffer from skewed learning processes due to the overwhelming majority of legitimate transactions over fraudulent ones. [12] highlighted the capability of GANs in generating realistic synthetic samples of the minority class, thus contributing to a more balanced and effective training of predictive models.

In parallel, Bayesian models have been increasingly recognized for their efficacy in statistical analysis and variable selection, especially in high-dimensional data settings typical in finance. [5] provided insights into Bayesian variable selection techniques, which are crucial in refining the performance of machine learning models in fraud detection .

The integration of GANs with Bayesian models, such as the Uni and Multi Variate Bayesian Model with Shrinkage Priors (BMSP), offers a promising approach to further enhance fraud detection systems. This combination aims to address the challenges of data imbalance and variable selection, providing a more robust framework for detecting fraudulent activities.

Building on this integration, the ”KnockoffGAN” method introduced by [9] presents a significant advancement in the field of variable selection using GANs. By employing a modified GAN frame-

work, ”KnockoffGAN” enables the generation of knockoff features without any assumptions on the distribution of the original features, thereby offering a flexible solution for feature selection in high-dimensional datasets. This approach is particularly notable for its ability to generate valid knockoffs that satisfy key statistical properties, ensuring the integrity of the variable selection process. The innovative use of a multi-output cross-entropy loss function in the discriminator design allows ”KnockoffGAN” to effectively distinguish between original and knockoff features, thereby enhancing the reliability of feature selection in fraud detection models.

The integration of GANs with advanced statistical methods like Bayesian models and the incorporation of ”KnockoffGAN” hold significant potential for improving fraud detection. However, challenges related to model complexity, interpretability, and ethical considerations in synthetic data generation remain areas for future research.

CHAPTER 3: METHODOLOGY

BMSP (Uni and Multi Variate Bayesian Model with Shrinkage Priors) is a novel algorithm that has been primarily used for variable selection in time series analysis [8]. The algorithm identifies the most relevant variables in a dataset by conducting binary segmentation based on P-values. While its primary application has been in time series analysis, it can also be adapted to select the best predictors for generating synthetic data in fraud detection.

The potential advantages of combining GANs and BMSP for fraud detection are multifold. By utilizing GANs to generate synthetic fraud samples, the imbalance in the fraud dataset can be addressed, providing a more balanced and representative training set. Moreover, BMSP can play a crucial role in selecting the most informative variables from the pool of predictors, ensuring that the generated synthetic data contains the most relevant features for accurate fraud detection. This combination is expected to improve the overall performance of fraud detection models by enhancing their ability to detect fraudulent activities, reducing false positives, and minimizing missed detections. The integration of GANs and BMSP leads to more robust and effective fraud detection systems that are better equipped to tackle the challenges posed by evolving and complex fraud schemes in the financial industry. To the best of our knowledge, there is limited literature exploring the combination of GANs and BMSP for fraud detection in the financial industry.

3.1 GAN in Fraud Detection

Generative Adversarial Networks (GANs) are a class of deep learning models consisting of two neural networks: the generator and the discriminator. GANs are designed to generate realistic synthetic data by learning the underlying data distribution from the training data. In the con-

text of fraud detection, GANs can be used to generate synthetic fraud samples that resemble real fraudulent transactions. By augmenting the training data with these synthetic samples, the GAN addresses the data imbalance problem, enabling fraud detection models to learn from a more balanced dataset.

The training process of GANs involves a game-like scenario where the generator generates synthetic data, and the discriminator tries to differentiate between genuine and synthetic data. The generator learns to improve its ability to generate realistic data by fooling the discriminator, while the discriminator improves its ability to distinguish between genuine and synthetic data. This adversarial training process leads to the generation of high-quality synthetic data that closely resembles the real data.

3.1.1 GAN Model Architecture

Generative Adversarial Networks (GANs) are a class of deep learning models consisting of two neural networks: the generator and the discriminator. GANs are designed to generate realistic synthetic data by learning the underlying data distribution from the training data. In the context of fraud detection, GANs can be used to generate synthetic fraud samples that resemble real fraudulent transactions. By augmenting the training data with these synthetic samples, the GAN addresses the data imbalance problem, enabling fraud detection models to learn from a more balanced dataset.

The training process of GANs involves a game-like scenario where the generator generates synthetic data, and the discriminator tries to differentiate between genuine and synthetic data. The generator learns to improve its ability to generate realistic data by fooling the discriminator, while the discriminator improves its ability to distinguish between genuine and synthetic data. This adversarial training process leads to the generation of high-quality synthetic data that closely re-

sembles the real data.

3.1.1.1 Network of Generator (G)

For this paper, the Generator G is designed as a sequential network with an initial dense layer comprising 128 units, followed by batch normalization in table 3.1. We placed the layer batch normalization after the first dense layer and before the second dense layer. This means that the outputs from the first dense layer will be normalized before they are passed to the second dense layer. This is then succeeded by another dense layer with 64 units, and finally, a dense output layer with a single unit having a linear activation function.

Table 3.1: Configuration of Generator (G) Network.

Layer Type	Units	Input Shape	Activation
Dense	128	10	relu
Batch Normalization	-	-	-
Dense	64	-	relu
Dense	1	-	linear

3.1.1.2 Network of Discriminator (D)

In table 3.2, the Discriminator D is constructed as a sequential network. It starts with a dense layer of 128 units, followed by a dropout layer with a rate of 0.3. Subsequently, another dense layer with 64 units is integrated, followed again by a dropout layer at the same rate. The network concludes with a dense output layer with a single unit and a sigmoid activation function. For compilation, the Adam optimizer with a learning rate of 0.0001 is employed. The loss function utilized is binary crossentropy, and accuracy is chosen as the metric for evaluation.

Table 3.2: Configuration of Discriminator (D) Network.

Layer Type	Units	Input Shape	Activation / Rate
Dense	128	1	relu
Dropout	-	-	0.3
Dense	64	-	relu
Dropout	-	-	0.3
Dense	1	-	sigmoid

This simplified diagram shows the basic flow of GAN:

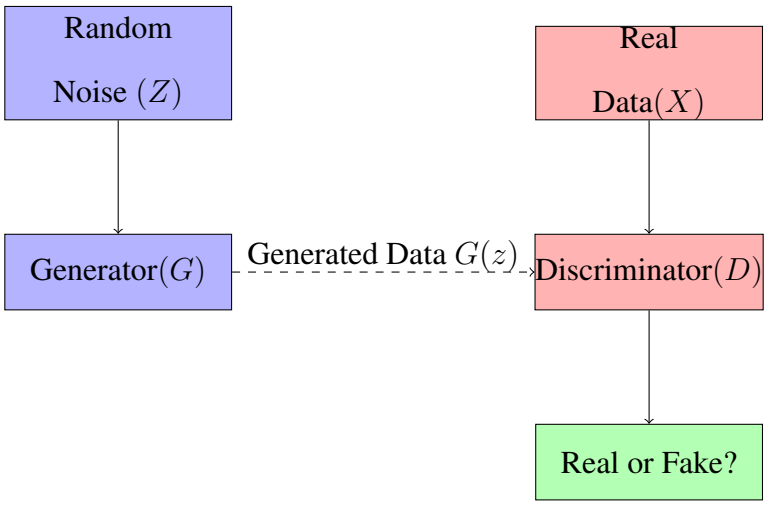

Figure 3.1: Architecture of Generative Adversarial Network (GAN)

- The Generator takes random noise as input and tries to produce data.

- The Discriminator takes both the real data and the data produced by the Generator and tries to differentiate between the two.

c. GAN Training Loop

The training of Generative Adversarial Networks (GANs) is conceptualized as a minimax game between two networks: the generator (G) and the discriminator (D), as described by [4]. The generator aims to produce synthetic data indistinguishable from real data, while the discriminator strives to accurately differentiate between the real and generated data. This dynamic can be expressed through the GAN objective function:

$$\min_{\theta_G} \max_{\theta_D} \left(\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p_Z} [\log(1 - D(G(z)))] \right) \tag{3.1}$$

where:

- p_{data} represents the distribution of real data x
- p_Z denotes the distribution of input noise z to the generator
- θ_G and θ_D are the parameters of the generator and discriminator

The discriminator's objective is to maximize its ability to correctly label real and generated data, expressed as:

$$\max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_Z(z)} [\log(1 - D(G(z)))] \tag{3.2}$$

Conversely, the generator's objective is to minimize the discriminator's ability to distinguish generated data from real data:

$$\min_G \mathbb{E}_{z \sim p_Z(z)}[\log(1 - D(G(z)))] \tag{3.3}$$

Algorithm 1 Training Generative Adversarial Network (GAN)

Input: Real data distribution $p_{\text{data}}(x)$, noise prior distribution $p_z(z)$, batch size m, learning rate α.
Initialize: Initialize generator $G(z; \theta_g)$ and discriminator $D(x; \theta_d)$ parameters.
while "not converged" **do**
    for k steps **do**
        Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior $p_z(z)$.
        Sample minibatch of m examples $\{x^{(1)}, ..., x^{(m)}\}$ from real data $p_{\text{data}}(x)$.
        Update the discriminator by ascending its stochastic gradient:

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

    end for
    Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior $p_z(z)$.
    Update the generator by descending its stochastic gradient:

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

end while
Output: Trained generator $G(z; \theta_g)$ capable of generating data resembling the real data distribution $p_{\text{data}}(x)$.

- $p_{\text{data}}(x)$: probability distribution of the real data.
- $p_z(z)$: prior probability distribution of the input noise to the generator.
- $G(z; \theta_g)$: generator network parameterized by θ_g, which maps the noise input z to the data space.
- $D(x; \theta_d)$: discriminator network parameterized by θ_d, which outputs a scalar representing the probability that x came from the real data rather than the generator.

- m: size of the minibatch, the number of samples used in each iteration of the training.
- α: learning rate, which controls the step size in the gradient-based optimization.
- $z^{(i)}$: i-th noise sample drawn from the noise prior $p_z(z)$.
- $x^{(i)}$: i-th real data sample drawn from the real data distribution $p_{\text{data}}(x)$.
- ∇_{θ_d}: gradient with respect to the discriminator parameters.
- ∇_{θ_g}: gradient with respect to the generator parameters.
- k: number of steps to apply to the discriminator's optimization for every generator's optimization step. This is used to ensure that the discriminator is sufficiently powerful at distinguishing real data from fake data generated by the generator.

The GAN algorithm involves two networks, the (G) and the (D), which are trained simultaneously through a min-max game. The generator tries to produce synthetic data that is indistinguishable from real data, while the discriminator aims to distinguish between real and synthetic data accurately.

- **1. Initialization:** both the generator and discriminator networks are initialized with random weights.
- **2. Training Loop:**
  - The discriminator is trained for k steps in each iteration of the loop. In each step, a minibatch of real data and a minibatch of fake data generated by the generator are used. The discriminator's parameters are updated by ascending the gradient of the discriminator's loss, which encourages it to correctly classify real data as real and generated data as fake.

  - The generator is then trained by generating a new minibatch of fake data and updating its parameters by descending the gradient of a loss function that measures how well the discriminator is fooled into thinking the generated data is real. The generator's goal is to maximize the probability of the discriminator making a mistake.

- **3. Convergence:** the process is repeated until the generator produces realistic data, and the discriminator can no longer easily distinguish real data from fake data.

- **4. Output:** the final output is the trained generator model, which can then be used to generate data samples that mimic the distribution of the real data.

This adversarial training process leads to the generator learning to produce data that is increasingly similar to the real data, as the discriminator becomes better at distinguishing real from fake, forcing the generator to improve.

3.1.2 BMSP Integration

The function operates within a Bayesian framework where prior distributions, with parameters like 'u', 'a', 'tau', 'd1', 'd2', 'c1', and 'c2', are assigned to the model parameters. These priors enforce shrinkage, which effectively allows the model to penalize less relevant variables and favor more relevant ones.

The response variable y consists of a mixture of 1s and 0s and the variable selection function effectively identifies the most relevant predictors in Z. The function is designed to discern variables in 'z' that are most predictive of the binary response 'y', regardless of the distribution of 1s and 0s. It is particularly adept at recognizing variables that have a strong relationship with the occurrence of the event represented by $y = 1$ and $y = 0$.

The function selects active columns (variables) based on their contribution to explaining the response ‘y‘. It utilizes a Gibbs sampling method (as indicated by the reference to ‘Mt_MBSP_Gibbs‘ within the function) to estimate the posterior distributions of the model parameters, including the regression coefficients (‘B_est‘). The ’active’ variables are those with non-zero coefficients after considering the shrinkage effect imposed by the priors.

The Bayesian framework, with its incorporation of prior information and the robust exploration of the parameter space provided by Gibbs sampling, enables the model to infer the relevance of predictor variables effectively. It identifies the most informative patterns and relationships within the data that differentiate active variables from noise.

Algorithm 2 Uni and Multi Variate Bayesian Model Selection and Shrinkage Priors (BMSP)

Input: Data $Z \in \mathbb{R}^{n \times p}$, Response vector $Y \in \mathbb{R}^n$, Response types, Hyperparameters: $u, a, \tau, d_1, d_2, c_1, c_2$, Algorithm options: 'algorithm', 'step1_iter', 'bound_error', 'max_iter', 'burnin'

Output: Estimated coefficients B_{est}, Active variables, Model parameters

Validate input dimensions and types; initialize $B_{\text{init}} \leftarrow 0_{p \times q}$, $\Sigma_{\text{init}} \leftarrow I_q$

For non-count response types, adjust B_{init} and Σ_{init}:

$B_{\text{init}}[i, j] \leftarrow (Z^T Z + \lambda I_p)^{-1} Z^T Y[:, j]$ for j in non-count types

$\Sigma_{\text{init}}[i, j] \leftarrow \frac{1}{n-1} \sum_{k=1}^{n} (y_{k,j} - X_k B_{\text{init}}[:, j])^2$

if 'algorithm' is '1step' **then**

    Perform Gibbs sampling with initialized parameters for 'max_iter' iterations

else if 'algorithm' is '2step' **then**

    Stage 1: Perform Gibbs sampling for 'step1_iter', identifying candidate set J

    $P(\beta_{ij} \neq 0 | Z, Y) \leftarrow$ Posterior inclusion probability for each β_{ij}

    $J \leftarrow \{i : \max_j P(\beta_{ij} \neq 0 | Z, Y) >$ threshold$\}$

    Stage 2: Refine Gibbs sampling on set J for remaining iterations

end if

Update and sample B_{est} and Σ_{est} from posterior distributions:

$B_{\text{est}}[i, j] | Z, Y, \Sigma \sim \mathcal{N}(\mu_{\beta_{ij}}, \Sigma_{\beta_{ij}})$

$\Sigma_{\text{est}} \leftarrow \frac{1}{n-1} \sum_{k=1}^{n} (Y_k - Z_k B_{\text{est}})^T (Y_k - Z_k B_{\text{est}})$

Identify active variables A based on posterior probabilities:

$A \leftarrow \{i : \max_j P(\beta_{ij} \neq 0 | Z, Y) >$ threshold$\}$

return Active variables A, Estimated coefficients B_{est}, Model parameters Σ_{est}

- $Z \in \mathbb{R}^{n \times p}$**:** Data matrix with n samples and p predictors or variables.
- $Y \in \mathbb{R}^n$**:** response vector for univariate responses with n samples.

- **Response types:** Vector indicating the type of each response variable (e.g., binary, count, continuous).
- u, a, τ**:** Hyperparameters related to the prior distributions, controlling aspects like the scale and shape of the distributions.
- d_1, d_2**:** Degrees of freedom and scale parameter for the Inverse-Wishart distribution used in the prior of the covariance matrix.
- c_1, c_2**:** Shape and rate parameters for the Gamma distribution used in the prior of variance components.
- **Algorithm options:** Settings like 'algorithm' choice (e.g., "1step" or "2step"), 'step1_iter' for the number of iterations in the first step of the two-step approach, 'bound_error' for the error tolerance, 'max_iter' for the maximum number of iterations, and 'burnin' for the number of initial iterations to discard.
- $B_{\textbf{init}} \leftarrow 0_{p \times q}$**:** Initial coefficient matrix, initialized to zeros, with dimensions corresponding to the number of predictors p and the number of responses q.
- $\Sigma_{\textbf{init}} \leftarrow I_q$**:** Initial covariance matrix, typically initialized as the identity matrix I_q with dimensions $q \times q$.
- $B_{\textbf{est}}$**:** Estimated coefficient matrix resulting from the algorithm, reflecting the strength and direction of relationships between predictors and responses.
- **Active variables:** Set of predictor variables identified by the algorithm as having a significant association with the response variables.
- **Model parameters:** Parameters like the covariance matrix Σ_{est} that are estimated during the algorithm's execution and are essential for understanding the model's structure and variability.

The Uni and Multi Variate Bayesian Model Selection and Shrinkage Priors (BMSP) algorithm is a sophisticated statistical technique designed for both univariate and multivariate response variable selection within a Bayesian framework. It starts by validating input data and response types, adjusting initial estimates based on the response characteristics. The algorithm offers flexibility in its approach, allowing users to choose between a single-step or a two-step Gibbs sampling process based on their specific needs.

In the one-step approach, the algorithm performs Gibbs sampling directly on the initialized parameters, iterating until convergence criteria are met. Alternatively, the two-step approach first identifies a candidate set of variables through initial Gibbs sampling, then refines this set in a subsequent focused sampling stage. This method aims to enhance the efficiency and accuracy of variable selection.

Throughout the process, the algorithm applies Bayesian shrinkage priors to effectively manage model complexity and prevent overfitting. It continuously updates the estimated coefficients and model parameters based on the posterior distributions derived from the sampling process. Active variables are identified based on their posterior inclusion probabilities, highlighting those with significant contributions to the model.

This BMSP algorithm stands out for its capacity to handle both uni and multivariate data, providing a versatile and powerful tool for researchers and analysts in various fields. Its rigorous statistical foundation and adaptability make it a valuable asset for complex data analysis tasks, enabling informed decision-making based on robust Bayesian principles.

Comparing the ‘BMSP‘ function with the ‘Mt_MBSP‘ function, here are the steps present in ‘BMSP‘ that are not explicitly detailed in ‘Mt_MBSP‘:

- **1. Lambda Calculation for Regularized Regression:** In ‘BMSP‘, there’s a specific men-

tion of using $\lambda = 0.01$ for adjusting initial coefficient estimates (B_{init}) for non-count response types using a form of regularized regression. This step involves calculating B_{init} as $(Z^T Z + \lambda I_p)^{-1} Z^T Y$ for each non-count response type.

- **2. Residual Calculation for Covariance Matrix Adjustment:** 'MBSP' explicitly calculates residuals for non-count response types to update the initial covariance matrix (Σ_{init}). The residuals are calculated as $Y - ZB_{\text{init}}$, and Σ_{init} is updated based on these residuals.
- **3. Bound Error Consideration in Two-Step Algorithm:** The 'BMSP' function considers a 'bound_error' parameter specifically in the two-step algorithm. This parameter is used to determine the threshold for identifying the candidate set of variables in the first stage of the two-step procedure.

These steps are tailored towards the initial setup and adjustment of model parameters before the main Gibbs sampling procedure, focusing on handling different response types and ensuring that the initial model parameters are reasonable given the data.

3.1.3 Proposed BMSP-GAN Algorithm in Fraud Detection

This algorithm demonstrates the integration of Bayesian variable selection, GANs, and deep learning techniques to identify relevant features for fraudulent transactions and generate synthetic data. It showcases the power of combining probabilistic modeling, neural networks, and advanced statistical techniques to gain insights, improve model performance, and generate realistic data for various applications.

The BMSP-GAN training loop integrates the BMSP process into the GAN framework by refining the input noise Z through BMSP before it is used by the generator G. This refinement process tailors Z to emphasize features that are important for the specific task, such as fraud detection.

The BMSP-GAN Objective Function is :

$$\min_{\theta_G} \max_{\theta_D} \left(\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p_Z} [\log(1 - D(G(Z_{\text{refined}})))] \right)$$

Where:

- $Z_{\text{refined}} = \text{BMSP}(z; \text{params})$ represents the noise vector z refined by the BMSP process based on the BMSP parameters (params).

The Discriminator's Objective is:

$$\max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_Z(z)} [\log(1 - D(G(Z_{\text{refined}})))]$$

The Generator's Objective is:

$$\min_G \mathbb{E}_{z \sim p_Z(z)} [\log(1 - D(G(Z_{\text{refined}})))]$$

By making these adjustments, the BMSP-GAN framework leverages the strengths of both BMSP for feature selection and GANs for synthetic data generation, resulting in a more targeted approach to generating synthetic data that is well-suited for tasks such as fraud detection.

The algorithm delineates the training of GAN augmented with Bayesian Model Selection and Averaging via BMSP, specifically tailored for generating synthetic fraud data. It begins by initializing the (G) and (D) networks, with G structured to transform noise into synthetic data through a series of dense and batch normalization layers, and D designed to differentiate real from synthetic data,

Algorithm 3 GAN Training with BMSP

Inputs:
d_{noise}: Dimension of the noise vector
d_{output}: Dimension of the output data
m: Batch size
BMSP parameters: Parameters for BMSP noise refinement
n_{epochs}: Number of training epochs
Adam learning rate: Learning rate for Adam optimizer
Initialize Generator G:
$G : \mathbb{R}^{d_{\text{noise}}} \to \mathbb{R}^{d_{\text{output}}}$
$G \leftarrow \text{Dense}(128, \text{ReLU}) \to \text{BatchNorm} \to \text{Dense}(64, \text{ReLU}) \to \text{Dense}(1, \text{linear})$
Initialize Discriminator D:
$D : \mathbb{R}^{d_{\text{output}}} \to [0, 1]$
$D \leftarrow \text{Dense}(128, \text{ReLU}) \to \text{Dropout}(0.3) \to \text{Dense}(64, \text{ReLU}) \to \text{Dropout}(0.3) \to \text{Dense}(1, \text{sigmoid})$
Compile D **with Loss Function** L_D:
$L_D = -\mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] - \mathbb{E}_{z \sim p_z}[\log(1 - D(G(Z_{\text{refined}})))]$
Compile D using Adam optimizer and binary crossentropy loss function
Generator Loss Function L'_G:
$L'_G = -\mathbb{E}_{z \sim p_z}[\log(1 - D(G(Z_{\text{refined}})))]$
function REFINENOISE(d_{noise}, m)
    Generate $Z \sim \log \mathcal{N}(\text{meanlog}, \text{sdlog}) \in \mathbb{R}^{m \times d_{\text{noise}}}$
    $Z_{\text{refined}} \leftarrow \text{BMSP}(Z, \text{params})$
    return Z_{refined}
end function
for each epoch **do**
    for each batch i **do**
        Sample $\{x_{\text{real}}\} \sim p_{\text{data}}$
        $Z_{\text{refined}} \leftarrow$ REFINENOISE(d_{noise}, m)
        Generate $\{x_{\text{fake}}\} = G(Z_{\text{refined}})$
        Update D using $\{x_{\text{real}}\}$ and $\{x_{\text{fake}}\}$ with loss L_D
    end for
    $Z_{\text{refined}} \leftarrow$ REFINENOISE(d_{noise}, m)
    Update G to minimize L'_G using Z_{refined}
    Apply early stopping based on L_D and L'_G
end for
Generate synthetic data $x_{\text{synthetic}} \leftarrow G(Z_{\text{final}})$
Return trained G, $x_{\text{synthetic}}$

incorporating dropout for regularization. The discriminator is compiled using the Adam optimizer and binary crossentropy loss, focusing on accurate classification.

A pivotal function, 'RefineNoise', generates initial noise and refines it using BMSP, a process that involves Bayesian statistical techniques to enhance the quality of the noise fed into G, by identifying and emphasizing informative patterns within the noise.

Training proceeds over a set number of epochs and batch sizes, where for each batch, real data samples are fetched, and refined noise is generated and passed through G to produce synthetic data. D is then updated to improve its distinguishing capabilities. Subsequently, G is updated based on D's responses to further refine its synthetic outputs. An early stopping mechanism is employed to halt training when improvements fall below a specified threshold, optimizing computational efficiency and preventing overfitting.

Upon completion, refined noise is once again generated to produce a final batch of synthetic data through G, which, along with the trained generator itself, is outputted. This sophisticated approach, blending GANs with Bayesian refinement via BMSP, is aimed at enhancing the generation of realistic synthetic fraud data, improving fraud detection models by providing rich, diverse training samples.

CHAPTER 4: FINDINGS

We utilized a dataset aimed at analyzing customer default payments in Taiwan, consisting of 30,000 individual credit card client instances. This inherently multivariate dataset encompasses a wide array of features relevant to credit defaults, including both integer and real number types. These features encompass various dimensions of a client's financial background and credit history, such as credit limit, payment history, demographic information, and bill statements. A critical aspect of this dataset is its distribution in the target variable - the likelihood of a client defaulting on the next payment.

The dataset reveals that 85.4% of clients did not default (represented as '0s') and 14.6% did default (represented as '1s'), indicating a significant imbalance that is crucial for predictive modeling and analysis. This classification aim aligns with the broader context of risk management in the financial sector, making the dataset a valuable tool for understanding patterns in customer financial behavior. The rich blend of features in this dataset provides a robust foundation for applying advanced data mining and machine learning techniques, including innovative approaches like GANs. These techniques are particularly relevant given the challenges posed by the imbalanced class distribution, underscoring the dataset's potential for exploring various predictive models and strategies.

Initial examination of the dataset revealed a pronounced class imbalance in the context of customer default payments. The instances labeled as non-defaulting (non-fraudulent), form the majority class. In contrast, the defaulting transactions (fraudulent), which are of critical interest in our study, represent the minority class. Such class imbalances are common in financial datasets and pose a significant analytical challenge. Machine learning models trained on imbalanced datasets often develop a bias towards the majority class, leading to suboptimal performance when identifying instances from the minority class. This imbalance necessitates the use of specialized techniques

and methodologies, such as oversampling the minority class or employing complex models like GANs, to enhance the model's ability to accurately identify and predict default payments.

To address this challenge, we employed a Generative GAN and BMSP-GAN to synthesize samples for the minority (fraudulent) class. In our results analysis, a pivotal aspect will be comparing the performance of models trained with data synthesized using the traditional GAN approach against those trained with data from the GAN-BMSP methodology. Additionally, two-sample tests, visualizations of distributions, feature wise statististics, and diversity tests are employed to ascertain the similarity between the distributions of the real and synthetic data.

In subsequent sections, our results analysis focuses on comparing the performance of various classification models trained on the RNA-Seq (HiSeq) PANCAN dataset featuring gene expression data for various tumor types including BRCA, KIRC, COAD, LUAD, and PRAD using BMSP function. This intricate and multivariate dataset, pivotal in the fields of biology and genomics, comprises 801 instances with a staggering 20,531 features per instance. It is noteworthy that BRCA emerges as the most prevalent tumor type with 300 instances, followed by KIRC, LUAD, PRAD, and COAD with 146, 141, 136, and 78 instances, respectively. This uneven distribution among tumor types poses a critical challenge in machine learning model performance, necessitating strategies like oversampling, downsampling or class weighting to mitigate the effects of this imbalance.

4.1 GAN vs BMSP GAN

The provided figures 4.1 and 4.2 depict a GAN and BMSP GAN models that have undergone effective training, marked by the evident convergence between the generator and discriminator. This convergence is a promising indicator of the GAN's capability to synthesize high-quality data

samples that closely mirror the distribution and characteristics of the original dataset.

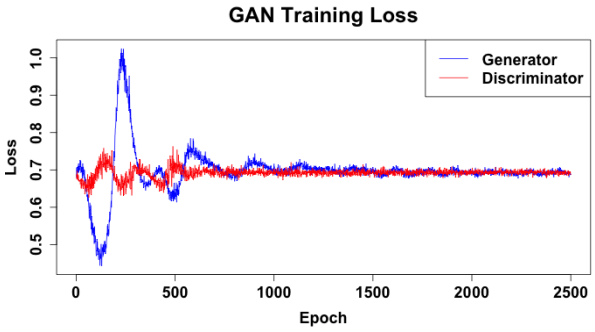

Figure 4.1: GAN.

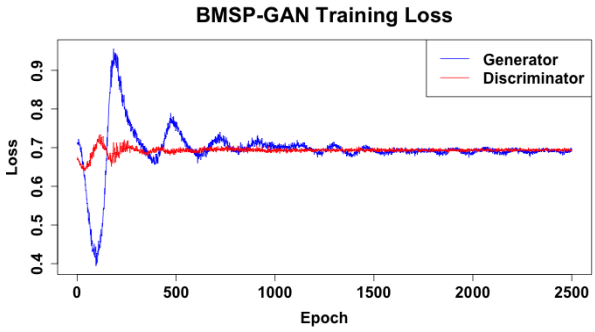

Figure 4.2: BMSP GAN.

The training loss graphs for BMSP-GAN and GAN models provide valuable insights into the models' learning dynamics over the course of 2500 epochs in figure 4.1 and 4.2. In the BMSP-GAN graph, the losses for both the generator and discriminator exhibit an initial period of volatility, which quickly stabilizes. The discriminator loss demonstrates a downward trend, stabilizing around a value that suggests effective learning without overpowering the generator. Conversely, the GAN graph presents a more erratic convergence pattern, with wider fluctuations in the discriminator's loss, indicating a less stable training process. The generator's loss in both graphs stabilizes to a similar degree, implying that the quality of the synthetic data generated might be comparable. However, the more stable convergence and tighter loss margins in the BMSP-GAN model suggest a more synchronized training process, leading to synthetic data that better captures the complexity of the underlying real data distribution.

4.1.1 Visualization of Density Distributions

We provide visual comparisons through density plots to offer intuitive insights into how well the synthetic minority data captures the essence and distribution of the real-world data it aims to mimic.

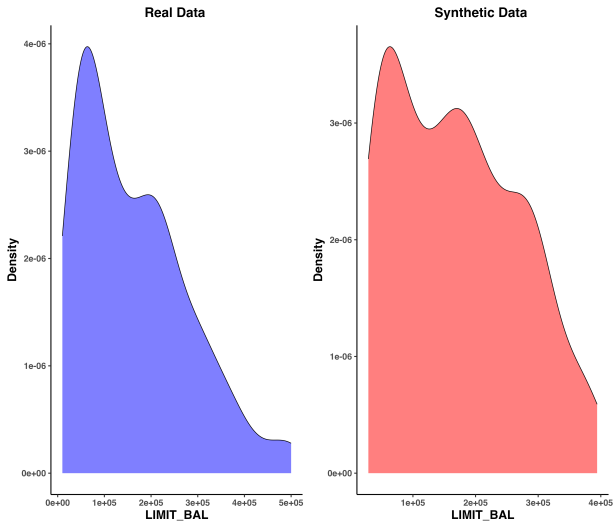

Figure 4.3: GAN.

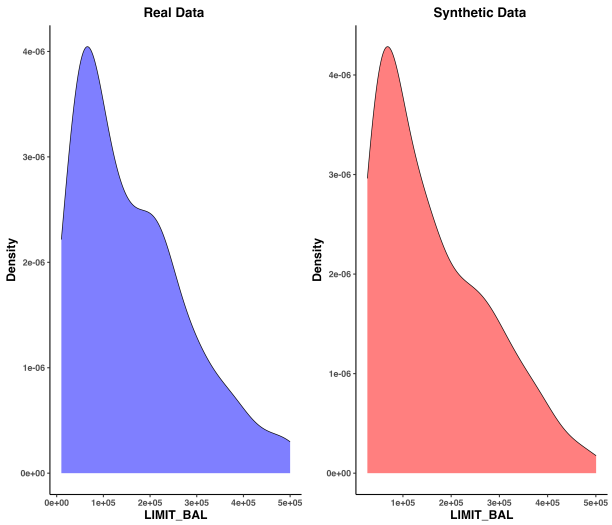

Figure 4.4: BMSP GAN.

- **Side-by-Side Density Plots Observation:**

From the side-by-side density plots (Figure 4.3 and 4.4), we observe a notable distinction in the quality of synthetic data generated by the two models. The density distribution of the synthetic data from BMSP GAN aligns almost seamlessly with the original data. On the other hand, the GAN's output, although commendable, shows noticeable disparities in certain density regions when juxtaposed with the original data. This suggests that BMSP GAN captures the nuances of the minority class with a higher degree of precision compared to the traditional GAN.

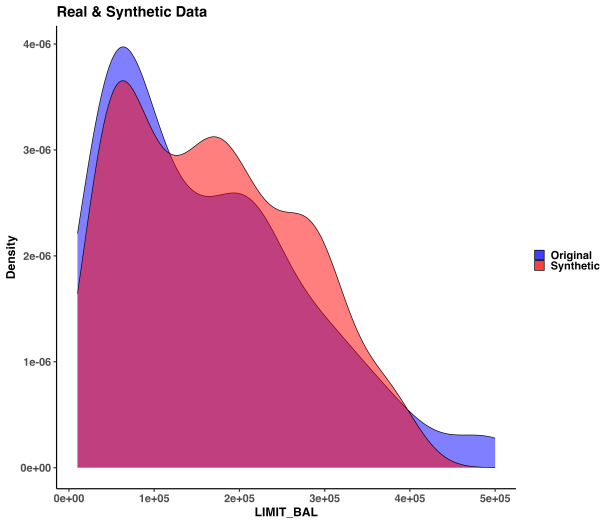

Figure 4.5: GAN.

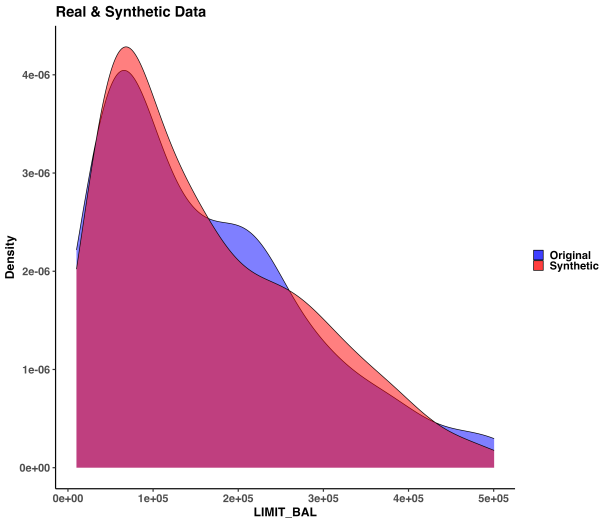

Figure 4.6: BMSP GAN.

- **Combined Density Plot Analysis:**

The combined density plot (Figure 4.5 and 4.6) provides a more holistic view of the models' performance. It's evident that the synthetic data overlay for BMSP GAN is nearly indistinguishable from the real data, showcasing its robust data generation capabilities. In contrast, the GAN's synthetic data exhibits areas of overlap, indicating potential regions where it hasn't perfectly emulated the original data distribution. This further consolidates the superiority of BMSP GAN in mimicking the real data's intricate patterns.

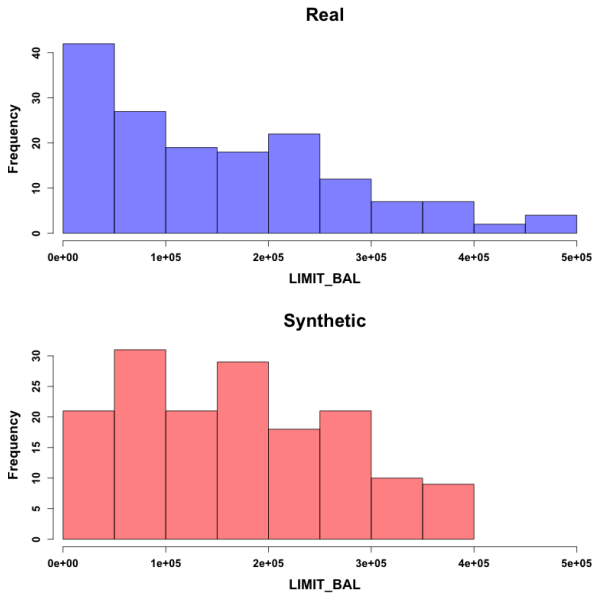

Figure 4.7: GAN.

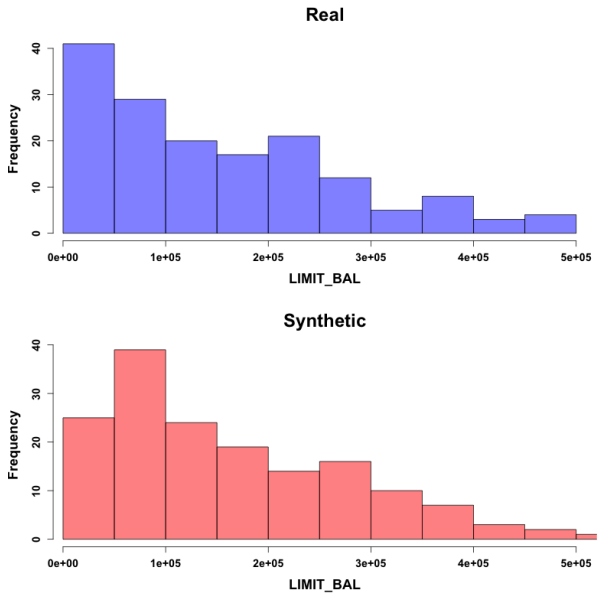

Figure 4.8: BMSP GAN.

- **Histograms:**

The histograms in figure 4.7 and 4.8 provide a bar-by-bar comparison between the real and synthetic data distributions for both the GAN and BMSP GAN models. The BMSP GAN model's histogram exhibits a closer approximation to the real data histogram when contrasted with the standard GAN model. This is evidenced by the BMSP GAN's more accurate reflection of the distribution's shape, spread, and frequencies of occurrences. Specifically, the synthetic data generated by the BMSP GAN aligns more closely with the real data across various ranges, suggesting a superior capability in capturing the intricate patterns and inherent variability of the dataset. This fidelity in emulation is indicative of the BMSP GAN's robustness and its potential for producing more realistic synthetic datasets, which is critical for training machine learning models where true data representation is paramount. The BMSP GAN's refined performance in matching the real data's histogram underscores its effectiveness as a generative model in our study.

4.1.2 Two-sample tests

The Kolmogorov-Smirnov (KS) test is employed to determine if two samples come from the same distribution. In this context, it's used to compare the distribution of the original data with that of the synthetic data generated by the models. A smaller D-value and a higher p-value suggest that the synthetic data closely resembles the original data distribution.

Table 4.1: Asymptotic two-sample Kolmogorov-Smirnov test results.

Asymptotic two-sample KS test	**D statistic**	**p-value**
Real and Synthetic data (GAN)	0.13125	0.127
Real and Synthetic data (BMSP GAN)	0.1	0.4005

For the GAN model, the KS test yields a D statistic of 0.13125 with a corresponding p-value of 0.127 in table 4.1. This indicates a moderate level of similarity between the synthetic data generated by the GAN and the original data distribution, as a p-value greater than 0.05 suggests that we cannot reject the null hypothesis of the distributions being the same at a 95% confidence level.

On the other hand, the BMSP GAN model produces a D statistic of 0.1 and a higher p-value of 0.4005, implying an even closer alignment between the synthetic and original data distributions. The increased p-value here suggests a stronger similarity, reinforcing the likelihood that the synthetic data from the BMSP GAN model comes from the same distribution as the original data.

4.1.3 Feature-wise statistics

The provided tables 4.2 and 4.3 detail the summary statistics of the balance variable for both real and synthetic data as generated by the GAN and BMSP GAN models. For almost all the summary

statistics, the BMSP GAN model produces synthetic data that is more representative of the real data than the traditional GAN model.

Table 4.2: Summary for GAN.

Real_Var1	**Real_Var2**	**Real_Freq**	**Synthetic_Var1**	**Synthetic_Freq**
A	Min.	10000.0	A	28832.00
A	1st Qu.	50000.0	A	71362.87
A	Median	140000.0	A	163730.20
A	Mean	162750.0	A	169116.92
A	3rd Qu.	230000.0	A	248781.30
A	Max.	500000.0	A	394117.12

Table 4.3: Summary for BMSP_GAN.

Real_Var1	**Real_Var2**	**Real_Freq**	**Synthetic_Var1**	**Synthetic_Freq**
A	Min.	10000.0	A	26237.62
A	1st Qu.	50000.0	A	62909.96
A	Median	140000.0	A	132938.90
A	Mean	162250.0	A	162690.47
A	3rd Qu.	230000.0	A	239116.94
A	Max.	500000.0	A	500554.47

The summary statistics compare the balance variable characteristics between the real dataset and the synthetic datasets generated by the GAN and BMSP GAN models. When analyzing the synthetic data's adherence to the real data's distribution, the BMSP GAN model's output exhibits an enhanced alignment, particularly in capturing central tendency measures. The synthetic median (132,938.90) and mean (162,690.47) generated by BMSP GAN closely approximate the real data's median (140,000) and mean (162,250), demonstrating the model's precision in reflecting the real dataset's core distributional features.

Contrastingly, the synthetic data from the GAN model shows a larger variance from the real data,

especially at the distribution's extremities. The minimum (28,832) and maximum (394,117.12) synthetic values manifest a broader range from the real data's minimum (10,000) and maximum (500,000), indicating a less accurate emulation of the real data's full range.

Furthermore, the interquartile range produced by BMSP GAN, with the first quartile (62,909.96) and the third quartile (239,116.94), more accurately mirrors the real data's quartile distribution than the traditional GAN. This is indicative of the BMSP GAN's superior capability in modeling the variability and dispersion inherent in the real data.

In summary, the BMSP GAN model proves to be more adept at generating synthetic data that maintains the integrity of the real data's statistical properties, particularly in terms of median and quartile values. This fidelity is crucial for applications requiring synthetic datasets that are representative of actual distributions, thereby reinforcing the BMSP GAN model's validity as a tool for generating realistic synthetic data for analytical purposes.

4.1.4 Diversity Test

The importance of capturing variance when generating synthetic data is critical, as also discussed in the broader context of statistical learning by [7]. When considering the generation of synthetic data, particularly using Generative Adversarial Networks, [6] provide a seminal framework that has been instrumental in advancing the field.

A diversity test in the context of synthetic data generation is a method used to compare the variance in the synthetic data to the variance in the real data. A higher variance means more diversity within the dataset. In generating synthetic data, one aims to replicate the statistical properties of the real data, including the mean, distribution shape, and variance.

The diversity ratio, calculated by dividing the variance of the synthetic data by the variance of the

real data, serves as an indicator of how well the synthetic data captures the diversity of the real data. A ratio close to 1 suggests that the synthetic data has a variance (and thus, diversity) similar to that of the real data. Ratios significantly lower than 1 indicate less diversity in the synthetic data compared to the real data, while ratios above 1 suggest more diversity than the real data.

Table 4.4: Diversity Test Results.

Method	LIMIT_BAL
GAN	0.7456542
BMSP GAN	0.9313663

The diversity ratio of 0.7456542 for GAN from table 4.4 suggests that the synthetic data generated by the GAN has less variance than the real data. The synthetic data is not capturing the full diversity of the real dataset, but it's reasonably close, indicating that the GAN is somewhat effective at replicating the real data's variability.

On the other hand , the diversity ratio of 0.9313663 for BMSP GAN is closer to 1, indicating that the BMSP GAN synthetic data has a variance very similar to that of the real data. This implies that the BMSP GAN model is highly effective at capturing the real data's diversity, performing slightly better than the traditional GAN.

In summary, both models are relatively successful in capturing the diversity of the real dataset, with the BMSP GAN showing a marginally better performance. This subtle difference may have significant implications when using the synthetic data for tasks that rely on the data's variability.

4.2 BMSP Classification Performance

In this study, we analyzed the RNA-Seq (HiSeq) PANCAN dataset from the UCI Machine Learning Repository, which includes gene expression data for various tumor types: BRCA, KIRC, COAD, LUAD, and PRAD. This multivariate dataset is crucial for classification and clustering tasks in biology. Our initial step involved loading and merging gene expression data with their respective tumor labels. Given the disparity in class sizes, notably the BRCA group with 300 samples, we employed oversampling techniques to balance the classes, targeting 300 samples for each. The dataset initially comprised 20,531 features, posing a challenge for model selection. To address this, we focused on addressing correlated variables and employing other preprocessing techniques. This approach effectively reduced the feature set from 20,531 to 12701. Subsequently, we applied BMSP for feature selection, successfully refining the feature set further to 61 significant features. These 61 features were then used to train multiple classification models, allowing us to compare the performance of BMSP with other model selection techniques. This comparative analysis aimed to evaluate the efficacy of different model selection strategies in the context of high-dimensional biological data, thereby providing a more robust understanding of the predictive capabilities and characteristics of the selected gene expression profiles in tumor classification.

4.2.1 Random Forest

The Random Forest model is trained using all features, and feature importance is assessed to identify the most influential ones. The top features are then selected based on the MeanDecreaseAccuracy criterion. In contrast, BMSP is a versatile approach capable of handling different types of response variables, including binary, count, and continuous. For this specific study, it was employed to directly select a subset of features from the oversampled dataset, focusing on binary

responses representative of the presence or absence of tumor types.

Table 4.5: Random Forest Selection and Results.

Class	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced Accuracy**	**Test Error Rate**
BRCA	0.9933333	0.9677419	1.0000000	0.9836066	0.9958333	0.006666667
COAD	0.9933333	1.0000000	0.9666667	0.9830508	0.9833333	0.006666667
KIRC	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667
LUAD	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667
PRAD	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667

Table 4.6: BMSP.

Class	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced Accuracy**	**Test Error Rate**
BRCA	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667
COAD	0.9933333	1.0000000	0.9666667	0.9830508	0.9833333	0.006666667
KIRC	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667
LUAD	0.9933333	0.9677419	1.0000000	0.9836066	0.9958333	0.006666667
PRAD	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667

The analysis of the classification performance of models trained on features selected by Random Forest and BMSP techniques in table 4.6 and 4.7 demonstrates their effectiveness in tumor type classification. Despite employing different methodologies for feature selection, both approaches achieve exemplary precision, recall, and F1 scores, with accuracies exceeding 99% for all con-

sidered classes. The Random Forest model exhibits minor variations in precision across classes, which may be reflective of the random selection processes intrinsic to the algorithm. Conversely, the BMSP method showcases remarkable precision consistency, suggesting a potential advantage in identifying the most discriminative features for classification tasks.

The Random Forest method shows slight variations in performance across different classes, particularly in precision for BRCA and COAD. This variation could be attributed to the inherently stochastic nature of Random Forests, where the randomness in feature selection and bootstrapping samples might lead to these small discrepancies. BMSP displays a consistent precision of 1.00 across all classes except for LUAD, which suggests that the features selected by BMSP may be highly discriminative for most classes.

Furthermore, the balanced accuracy rates confirm that both methods are well-tuned to account for class imbalances, ensuring that the models are equally adept at identifying each tumor type. However, the test error rates, although low for both methods, indicate a marginal distinction in generalization performance. This suggests that the specific features selected by each method could have implications for model robustness in practical applications. The slight differences observed warrant a deeper investigation into the nature and biological relevance of the features selected by each approach, which could provide insights into their predictive power and potential utility in clinical settings

4.2.2 XGBoost

Table 4.7: XGboost Selection and results.

Class	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced Accuracy**	**Test Error Rate**
BRCA	0.9933333	0.9677419	1.0000000	0.9836066	0.9958333	0.006666667
COAD	0.9933333	1.0000000	0.9666667	0.9830508	0.9833333	0.006666667
KIRC	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667
LUAD	0.9933333	0.9677419	1.0000000	0.9836066	0.9958333	0.006666667
PRAD	0.9933333	1.0000000	1.0000000	1.0000000	1.0000000	0.006666667

Table 4.8: BMSP.

Class	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Balanced Accuracy**	**Test Error Rate**
BRCA	0.9916667	1.00	0.9916667	0.9958159	0.9958333	0.008333333
COAD	0.9916667	1.00	0.9666667	0.9830508	0.9983333	0.008333333
KIRC	0.9916667	1.00	1.0000000	1.0000000	1.0000000	0.008333333
LUAD	0.9916667	0.96	1.0000000	0.9795918	0.9947917	0.008333333
PRAD	0.9916667	1.00	1.0000000	1.0000000	1.0000000	0.008333333

From table 4.8 and 4.9, for the BRCA, KIRC, and PRAD classes, both XGBoost and BMSP achieved flawless precision and recall, resulting in an F1 score of 1.0000000, which suggests that for these specific tumor types, the models performed without any false positives or false negatives. This level of performance is also reflected in the balanced accuracy, which adjusts for any

imbalance in the classes, thus reinforcing the robustness of the models.

However, there are slight differences between the models for COAD and LUAD classes. For COAD, BMSP demonstrates a marginally lower precision than XGBoost, which may suggest a higher occurrence of false positives. Similarly, for LUAD, BMSP has a slightly lower precision than XGBoost, which may indicate a few instances where non-LUAD cases were incorrectly classified as LUAD.

The test error rates are very low for both models across all classes, with BMSP showing a slightly higher error rate for BRCA, COAD, and LUAD compared to XGBoost. Despite these minor differences, both XGBoost and BMSP show excellent performance, with BMSP having the advantage of being able to handle various types of response variables, which is particularly beneficial in multi-class classification scenarios.

Across all two approaches, Random Forest and XGboost classification exhibited remarkable consistency with an Accuracy exceeding 99 % and minimal Test Error Rates for all tumor types. This consistency emphasizes the robustness of the Random Forest and XGboost algorithms in handling the feature subsets generated by different selection methods.

CHAPTER 5: CONCLUSION

The BMSP GAN model emerges as a significant advancement in synthetic data generation for fraud detection. It provides a robust approach to managing class imbalances by producing high-fidelity synthetic samples. This research contributes to the field of data science by offering a novel method that enhances machine learning model training, which could be applied across various domains facing similar challenges with imbalanced datasets. Future research should focus on enhancing the BMSP model's variable selection capabilities for faster speed convergence. Such improvements could lead to a more streamlined and efficient process, further augmenting the model's applicability and effectiveness in various data-intensive domains, including but not limited to fraud detection, potentially broadening its utility in the broader landscape of data analytics.

APPENDIX A: MODE COLLAPSE AND VANISHING GENERATOR GRADIENTS OF GAN

The problem of mode collapse and vanishing generator gradients has also been mentioned by [11] in the context of GAN training.

A.1 Mode Collapse

At its core, mode collapse in GANs happens when the generator produces a limited variety of samples even if the real data distribution has multiple modes (distinct types of data). For instance, if the real data has modes A and B, mode collapse might result in the generator predominantly producing samples that resemble just A or B, but not both. This problem can be illustrated when different latent vectors $z \sim p_z$ start mapping to very similar outputs $G(z)$. Given the objective of the generator in GANs:

$$\Delta_{\theta_g} \frac{1}{m} \sum_{i=0}^{m} \left[\log \left(1 - D \left(G \left(z^{(i)} \right) \right) \right) \right]$$

In the presence of mode collapse, for different latent vectors z_1 and z_2, the generated samples I_1 and I_2 tend to be similar. This is not desired, as we want diverse outputs for diverse inputs. The BMSP function, by selecting the most pertinent variables, it can guide the generator to better represent the underlying data distribution. By focusing on the most influential variables, the generator can gain better insights into the underlying multi-modal nature of the real data distribution to produce a more comprehensive and diverse set of samples, potentially mitigating the mode collapse issue.

A.2 Diminishing generator gradients

[11] highlighted the challenge of diminishing gradients, especially in high-dimensional data scenarios. This phenomenon occurs when the generator's outputs are uniformly rejected with closely

similar loss values. In essence, when the real data distribution and the generator's data distribution are nearly disjoint, a near-perfect discriminator can emerge, leading to a scenario where generator gradients vanish. Consider two distributions P_g and P_r. If these distributions lie on low-dimensional manifolds that are mostly disjoint, a discriminator D can easily differentiate between them. This can be represented by the conditions:

$$P_r[D(x) = 1] = 1 \quad \text{and} \quad P_g[D(x) = 0] = 1$$

[11] discuss the behavior of data when considering the distance between two independent variables U, V that follow a uniform distribution on $[0, 1]^d$. The mean square distance $\|U - V\|^2$ is given by:

$$\mathbb{E}\left[\|U - V\|^2\right] = \frac{d}{6} \text{ and } \sigma\left[\|U - V\|^2\right] \simeq 0.2\sqrt{d}$$

With the growth of dimensionality d, the concept of 'nearest neighbors' starts to lose its significance. This is because distances in these high-dimensional spaces begin to concentrate within a narrow range. At first glance, it might appear that given the near emptiness of these spaces, class separation using a hyperplane should be straightforward. However, they further illustrate that data points tend to congregate at the boundaries of these spaces, complicating the task of prediction. In essence, while traditional GANs might grapple with the vastness and complexity of such spaces, BMSP provides a focused lens, emphasizing only the most critical variables. This selective emphasis can potentially counteract the problems arising from data concentration at the boundaries, enabling the generator to produce high-quality synthetic samples that more genuinely reflect the underlying data distribution.

APPENDIX B: FUNCTIONS

B.1 MtMBSP Function

In recent developments within the realm of Bayesian frameworks for mixed-type multivariate regression, Wang et al. (2023) introduced a pioneering approach known as the Mixed-Typed Multivariate Bayesian Model with Shrinkage Priors (MtMBSP) which emphasizes the utilization of continuous shrinkage priors. The MtMBSP method accommodates the joint analysis of mixed continuous and discrete outcomes through a model that can be described as:

$$g(\mathbf{Y}_i) = \mathbf{Z}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \Sigma), \quad i = 1, ..., n$$

where:

- $\mathbf{Y}_i$ represents the multivariate response vector for the i-th observation, which may contain both continuous and discrete outcomes.
- $g(\cdot)$ denotes a suitable link function that is applied element-wise to the components of $\mathbf{Y}_i$, transforming each response to a scale where linear modeling is appropriate. For continuous responses, an identity link function may be implied, while for discrete responses (e.g., binary, count), a logit, probit, or log link function may be employed.
- $\mathbf{Z}_i$ is the design matrix for the i-th observation, incorporating the covariates.
- $\boldsymbol{\beta}$ is the coefficient matrix, representing the effects of the covariates on the transformed responses.
- Σ is the covariance matrix, which captures the correlations between the multivariate responses.

The continuous shrinkage priors are imposed on the elements of $\boldsymbol{\beta}$ to induce sparsity and facilitate variable selection, allowing for the identification of significant covariates influencing the responses.

Their innovative methodology facilitated the joint analysis of outcomes, both discrete and continuous in nature, while simultaneously performing variable selection from a myriad of covariates. Notably, the theoretical exploration of such Bayesian models for mixed-type multivariate responses is a daunting challenge, primarily due to the intricate correlations between varying responses. What distinguishes their work is their dive into the asymptotic regime where the number of covariates p can grow exponentially in terms of the sample size n, a territory that is rare in the existing literature. Their method's robustness, especially under such exponential growth of p, is commendably showcased through a series of simulations and real-world dataset applications.

B.2 BMSP Function

Building on the foundational work of Wang et al. (2023), we adapted the MtMBSP function to Univariate and Multivariate Bayesian Model with Shrinkage Priors (BMSP). Our modified version not only retains the capability to handle mixed-type multivariate responses but is also adept at managing single response variables. The proposed work broadens the scope and application of the Bayesian framework, catering to a wider spectrum of regression scenarios.

$$g(\mathbf{Y}_i) = \mathbf{Z}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \Sigma_i), \quad i = 1, ..., n$$

where:

- $\mathbf{Y}_i$ represents the response variable for the i-th observation, which can be univariate or multivariate, encompassing continuous and/or discrete outcomes.

- $g(\cdot)$ denotes a suitable link function applied element-wise to $\mathbf{Y}_i$. This function transforms the response variable to a scale where linear modeling is appropriate. For continuous responses, this might be an identity link, while for discrete responses, appropriate link functions such as logit (for binary data), probit, or log (for count data) may be used.
- $\mathbf{Z}_i$ is the design matrix for the i-th observation, which includes the covariates.
- $\boldsymbol{\beta}$ is the coefficient matrix (or vector in the univariate case), which describes the effects of the covariates on the transformed responses.
- Σ_i is the covariance matrix for the i-th observation's errors. In the univariate case, this reduces to a scalar variance σ^2, while in the multivariate case, it remains a covariance matrix capturing the correlations among the multiple responses.

The BMSP model, similar to its predecessor, employs continuous shrinkage priors on the coefficients $\boldsymbol{\beta}$ to induce sparsity and facilitate the selection of significant covariates. This adaptation ensures that the Bayesian framework remains versatile and applicable across a broad array of regression contexts, from univariate to multivariate and from continuous to mixed-type data scenarios.

B.2.1 Algorithm Steps in BMSP

The BMSP function, like MtMBSP, offers the flexibility of using both "1step" and "2step" algorithms. However, BMSP introduces additional complexity in the "2step" algorithm, particularly in handling subsets of variables. This is evident in the way it deals with the sets set_J and set_Jc. In the '2step' algorithm of BMSP, after the first stage, it identifies a subset of predictors (set_J) that are active (i.e., having non-zero coefficients). This subset is then used in the second stage of the algorithm, where the model is refitted only using these selected predictors. Additionally, it takes a

unique approach by also considering the complementary set (set_Jc), which consists of predictors not included in set_J. This bifurcation allows for a focused analysis on significant predictors, enhancing the model's accuracy and efficiency. It's a form of variable selection and dimensionality reduction, which is especially useful in datasets with a large number of predictors.

B.2.2 BMSP from MtMBSP

In BMSP, there's a notable difference in how the outputs, particularly the $\mathbf{B}$ estimates (regression coefficients) and $\mathbf{\Sigma}$ estimates (covariance matrix), are computed and adjusted compared to MtMBSP. In BMSP, after running the Gibbs sampling process, it compiles the final estimates by combining results from the subsets "set_J" and "set_Jc". This method seems to be more tailored, potentially leading to more accurate estimates, especially in cases where the dataset contains a mix of significant and non-significant predictors. Moreover, this approach might also contribute to the computational efficiency of the algorithm. By isolating and focusing computational resources on the more significant predictors ("set_J"), the algorithm achieves faster convergence and reduce the overall computational load, which is particularly beneficial for large datasets or when computational resources are limited.

In summary, BMSP's incorporates an advanced variable selection process, and its unique approach to compiling output estimates, not only add to its versatility but also enhance its computational efficiency. These adjustments make BMSP a more robust tool for statistical modeling, especially in scenarios with complex datasets.

LIST OF REFERENCES

[1] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[3] Andrea Dal Pozzolo, Olivier Caelen, Yann-Ael Le Borgne, Serge Waterschoot, and Gianluca Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*, 41(10):4915–4928, 2014.

[4] Ugo Fiore, Alfredo De Santis, Francesca Perla, Paolo Zanetti, and Francesco Palmieri. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, 479:448–455, 2019.

[5] Edward I George and Robert E McCulloch. Approaches for bayesian variable selection. *Statistica sinica*, pages 339–373, 1997.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[7] Trevor Hastie, Robert Tibshirani, Jerome Friedman, Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Boosting and additive trees. *The elements of statistical learning: data mining, inference, and prediction*, pages 337–387, 2009.

[8] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[9] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. Knockoffgan: Generating knockoffs for feature selection using generative adversarial networks. In *International conference on learning representations*, 2018.

[10] Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. Survey of fraud detection techniques. In *IEEE International Conference on Networking, Sensing and Control, 2004*, volume 2, pages 749–754. IEEE, 2004.

[11] Hadi Mansourifar, Lin Chen, and Weidong Shi. Virtual big data for gan based data augmentation. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1478–1487. IEEE, 2019.

[12] Chuanjun Zhao, Xuzhuang Sun, Meiling Wu, and Lu Kang. Advancing financial fraud detection: Self-attention generative adversarial networks for precise and effective identification. *Finance Research Letters*, page 104843, 2023.