



KADIR HAS UNIVERSITY
SCHOOL OF GRADUATE STUDIES
PROGRAM OF MANAGEMENT INFORMATION SYSTEMS

**PRIVACY PRESERVATION FOR INDUSTRIAL DATA
USING GENERATIVE ADVERSARIAL NETWORKS AND
DIFFERENTIAL PRIVACY IN MACHINE LEARNING**

YAVUZ SELİM HİNDİSTAN

DOCTOR OF PHILOSOPHY THESIS

ISTANBUL, AUGUST, 2023

Yavuz Selim Hindistan

Ph.D. Thesis

2023

PRIVACY PRESERVATION FOR INDUSTRIAL DATA USING GENERATIVE ADVERSARIAL NETWORKS AND DIFFERENTIAL PRIVACY IN MACHINE LEARNING



YAVUZ SELİM HİNDİSTAN

A thesis submitted to
the School of Graduate Studies of Kadir Has University
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in
the Program of Management Information Systems

İstanbul, August, 2023

APPROVAL

This thesis titled PRIVACY PRESERVATION FOR INDUSTRIAL DATA USING GENERATIVE ADVERSARIAL NETWORKS AND DIFFERENTIAL PRIVACY IN MACHINE LEARNING submitted by YAVUZ SELİM HİNDİSTAN, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Program of Management Information Systems is approved by

Asst. Prof. E. Fatih YETKİN (Advisor)
Kadir Has University

Prof. Dr. Hasan DAĞ
Kadir Has University

Asst. Prof. Kerem ALTUN
Işık University

Assoc. Prof. Mehmet Nafiz AYDIN
Kadir Has University

Assoc. Prof. Oğuzhan CEYLAN
Marmara University

I confirm that the signatures above belong to the aforementioned faculty members.

.....

Prof. Dr. Mehmet Timur Aydemir
Dean of School of Graduate Studies

Date of Approval: 16.08.2023

DECLARATION ON RESEARCH ETHICS AND PUBLISHING METHODS

I, YAVUZ SELİM HİNDİSTAN; hereby declare

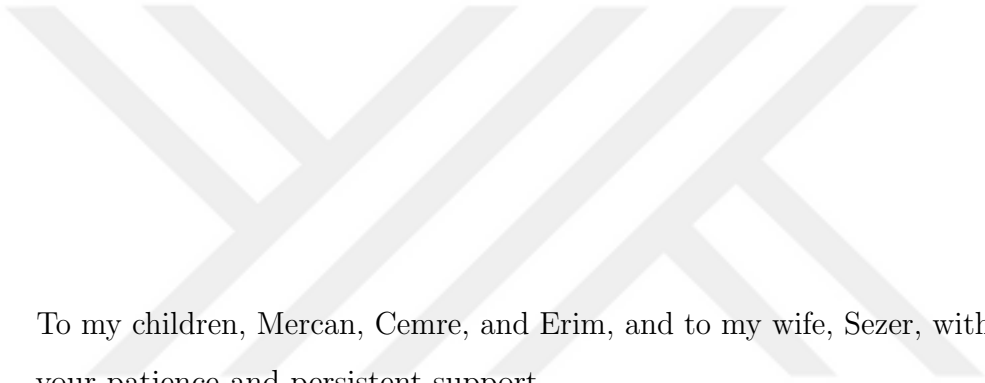
- that this Ph.D. Thesis that I have submitted is entirely my own work and I have cited and referenced all material and results that are not my own in accordance with the rules;
- that this Ph.D. Thesis does not contain any material from any research submitted or accepted to obtain a degree or diploma at another educational institution;
- and that I commit and undertake to follow the “Kadir Has University Academic Codes and Conduct” prepared in accordance with the “Higher Education Council Codes of Conduct”.

In addition, I acknowledge that any claim of irregularity that may arise in relation to this work will result in a disciplinary action in accordance with university legislation.

YAVUZ SELİM HİNDİSTAN

.....

16.08.2023



To my children, Mercan, Cemre, and Erim, and to my wife, Sezer, with gratitude for your patience and persistent support.

ACKNOWLEDGEMENT

When I made the decision to pursue a Ph.D. at the age of 44, the challenge of finding a graduate school that would support my research while I was actively engaged in the business world was substantial. I am profoundly grateful to Kadir Has University's MIS Department and Prof. Dr. Hasan Dağ for providing me the opportunity to commence my Ph.D. journey. I also wish to express my sincere gratitude to my research advisor, Asst. Prof. E. Fatih Yetkin, for his unwavering support and guidance. I am indebted to him for his patience and comprehensive supervision throughout this academic endeavor.

While I pursued this journey in part to set a benchmark for my children about the value of perseverance and dedication, I recognize that the time I devoted to this endeavor was borrowed from my family and parents. I sincerely thank them for their patience and steadfast support, as my success was made possible with their help.

In gratitude for all the support given to me, I aspire to continue my research and volunteer as a mentor for those just starting out, nurturing their enthusiasm for learning and discovery.

PRIVACY PRESERVATION FOR INDUSTRIAL DATA USING GENERATIVE ADVERSARIAL NETWORKS AND DIFFERENTIAL PRIVACY IN MACHINE LEARNING

ABSTRACT

The rapid expansion of Artificial Intelligence (AI) and Machine Learning (ML) has raised concerns about privacy preservation. These technologies rely on extensive datasets containing personal and sensitive information from sources like industrial IoT, social media, and online platforms, introducing privacy risks. Ensuring robust privacy protection measures is crucial to mitigate AI and ML application risks. This thesis investigates privacy preservation within AI and ML systems. Our research utilized publicly available datasets to develop a privacy protection method while maintaining ML accuracy. To enhance privacy, we strengthened our approach with Differential Privacy (DP) and Generative Adversarial Networks (GAN). We evaluated its effectiveness using six privacy metrics. Our approach demonstrates feasibility and efficacy, upholding privacy without compromising ML performance. Uncovering hidden sensitive information was emphasized, as extracted feature subsets can expose sensitive data with ML models. We demonstrate the method's effectiveness in addressing privacy concerns in an experimental study. The findings contribute to understanding privacy in AI and ML systems. The research offers insights and approaches to protect information, ensuring reliable ML outcomes. By advancing privacy knowledge, this work supports the responsible deployment of AI and ML technologies in safeguarding privacy.

Keywords: Data Privacy, Privacy Preservation, Machine Learning (ML), Artificial Intelligence (AI), Differential Privacy (DP), Generative Adversarial Networks (GAN), Privacy Metrics, Internet of Things (IoT)

MAKİNE ÖĞRENMESİNDE ENDÜSTRİYEL VERİ MAHREMİYETİNİN ÜRETKEK DÜŞMAN AĞLARI VE DİFERANSİYEL GİZLİLİK KULLANARAK KORUNMASI

ÖZET

Yapay Zeka (AI) ve Makine Öğreniminin (ML) hızla yaygınlaşması, mahremiyetin korunmasına ilişkin endişeleri artırdı. Bu teknolojiler, endüstriyel IoT, sosyal medya ve çevrimiçi platformlar gibi kaynaklardan kişisel ve hassas bilgiler içeren ve gizlilik riskleri getiren kapsamlı veri kümelerine dayanır. Güçlü gizlilik koruma önlemlerinin alınması, AI ve ML uygulama risklerini azaltmak için çok önemlidir. Bu tez, AI ve ML sistemlerinde gizliliğin korunmasını incelemektedir. Araştırmamız, ML doğruluğunu korurken bir gizlilik koruma yöntemi geliştirmek için herkese açık veri kümelerinden yararlandı. Gizliliği artırmak için, yaklaşımımızı Diferansiyel Gizlilik (DP) ve Üretken Düşman Ağları (GAN) ile güçlendirdik. Etkinliğini altı gizlilik ölçüsü kullanarak değerlendirdik. Yaklaşımımız, ML performansından ödün vermeden gizliliği koruyarak fizibilite ve etkinlik göstermektedir. Ayıklanan özellik alt kümeleri, ML modelleriyle hassas verileri açığa çıkarabildiğinden, gizli hassas bilgilerin ortaya çıkarılması vurgulanmıştır. Yöntemin mahremiyet endişelerini ele almadaki etkinliğini deneysel bir çalışmada gösteriyoruz. Bulgular, AI ve ML sistemlerinde gizliliğin anlaşılmasına katkıda bulunur. Araştırma, bilgileri korumak için içgörüler ve yaklaşımlar sunarak güvenilir ML sonuçları sağlar. Bu çalışma, gizlilik bilgisini ilerleterek, gizliliğin korunmasında AI ve ML teknolojilerinin sorumlu gelişimini destekler.

Anahtar Sözcükler: Veri Mahremiyeti, Gizlilik Koruma, Makine Öğrenimi, Yapay Zeka (AI), Diferansiyel Gizlilik (DP), Üretken Düşman Ağlar (GAN) , Gizlilik Metrikleri, Nesnelerin İnterneti (IoT)

TABLE OF CONTENTS

ACKNOWLEDGEMENT	v
ABSTRACT	vi
ÖZET	vii
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF SYMBOLS	xv
LIST OF ACRONYMS AND ABBREVIATIONS	xviii
1. INTRODUCTION	1
2. LITERATURE REVIEW	5
2.1 DP implementations with Data Protection approach	8
2.2 GAN implementations with Data Protection approach	11
2.3 GAN and DP Hybrid implementations	15
2.4 Hidden Sensitive implementations	15
2.5 Dimension Reduction	17
3. METHODOLOGY	18
3.1 Effects of Perturbation	18
3.2 Generative Adversarial Networks (GAN)	23
3.3 Differential Privacy (DP)	29
3.4 Privacy Metrics	34
3.4.1 Privacy Metric - Mean Squared Error (P_{MSE})	36
3.4.2 Privacy Metric - Normalized Variance (P_{VAR})	37
3.4.3 Privacy Metric - Directed Hausdorff (P_{HAUS})	37
3.4.4 Privacy Metric - Kullback-Leibler Divergence (P_{KL})	38
3.4.5 Privacy Metric - Procrustes(P_{PRO})	38
3.4.6 Privacy Metric - Pearson's Correlation Coefficient (P_{PCC})	39
3.5 The Proposed Method	39
3.6 Privacy Preservation with Dimension Reduction	45
3.7 Cost Analysis	47

4. NUMERICAL EXPERIMENTS	48
4.1 Experimental Environment	48
4.2 Vanilla-GAN Calibration Experiments	51
4.3 DP Experiments	56
4.4 Performance Analysis of Privacy Metrics	57
4.5 Experimental Exploration of Privacy Preservation Methods	61
4.6 Vanilla Gan versus CTGAN Experiments	70
4.7 Proposed Method Experiments	73
4.8 PCA Experiments	84
4.9 Hidden Sensitive Data Experiments	88
4.10 Computational Cost Experiments	92
5. CONCLUSIONS	94
5.1 Scope and Limitations	94
5.2 Future Directions	95
BIBLIOGRAPHY	96
APPENDIX A: Extended Version in Chapter 4.5 - Experimental Explo-	
ration of Privacy Preservation Methods	105
APPENDIX B: Extended Version in Chapter 4.6 - Vanilla Gan versus	
CTGAN Experiments	108
APPENDIX C: Extended Version in Chapter 4.7 - Proposed Method	
Experiments	111
APPENDIX D: Extended Version in Chapter 4.9 - Hidden Sensitive Data	
Experiments	147

LIST OF FIGURES

Figure 1.1	Conceptual representation of data transmission to ML	2
Figure 2.1	Plausible Deniability	8
Figure 2.2	Generative Adversarial Networks	13
Figure 3.1	The Sigmoid function.	18
Figure 3.2	IRIS Dataset Training - Algorithm 1 Application	21
Figure 3.3	IRIS Dataset Test - Algorithm 1 Application	22
Figure 3.4	MNIST Experiment of GAN	24
Figure 3.5	Generative Adversarial Networks	26
Figure 3.6	Illustration of the proposed approach with histograms.	43
Figure 3.7	A Sample IoT Pipeline Illustration	47
Figure 4.1	Reference Chart of Colors and Notations Used in Experiments .	51
Figure 4.2	Calibration for Minimum Loss (Vanilla - GAN)	54
Figure 4.3	PDF Comparison of Synthetic vs. Original	55
Figure 4.4	PDF Comparison: Original vs. DP (ϵ variations)	56
Figure 4.5	Performance Analysis of Privacy Metrics	59
Figure 4.6	Variations Among 10 Tests (PP Methods)	63
Figure 4.7	PP Methods - Logistic Regression	65
Figure 4.8	PP Methods - Support Vector Machine	67
Figure 4.9	PP Methods - Random Forest	68
Figure 4.10	PP Methods	69
Figure 4.11	Variations Among 10 Tests - (Vanilla GAN vs CTGAN)	71
Figure 4.12	Variations Among 10 Tests (X_{HGD})	76
Figure 4.13	Accuracy vs. Privacy Analysis of X_{HGD} (Linear Regression) . .	77
Figure 4.14	Comprehensive Bar Subplot Analysis (X_{HGD})	77
Figure 4.15	Dataset Correlation Heatmaps	80
Figure 4.16	Accuracy vs. Privacy Analysis of X_{HG} (Linear Regression) . . .	82
Figure 4.17	Comprehensive Bar Subplot Analysis (X_{HG})	82
Figure 4.18	Accuracy vs. Privacy Analysis of X_{HD} (Linear Regression) . . .	83
Figure 4.19	Comprehensive Bar Subplot Analysis (X_{HD})	83

Figure 4.20	PDF Comparison: Original vs. PCA Variations	86
Figure 4.21	Accuracy vs. Privacy Analysis of PCA (Linear Regression) . . .	86
Figure 4.22	Comprehensive Bar Subplot Analysis (PCA)	87
Figure 4.23	Comprehensive Bar Subplot Analysis (Case Study)	91
Figure 4.24	Resource Usage Analysis: CPU and RAM	92
Figure C.1	Variations Among 10 Tests (X_{HGD}) - Wind Turbine Dataset . .	111
Figure C.2	Variations Among 10 Tests (X_{HGD}) - Steam Production Dataset	114
Figure C.3	Variations Among 10 Tests (X_{HGD}) - Energy Efficiency Dataset	117
Figure C.4	Variations Among 10 Tests (X_{HGD}) - Synchronous Motors Dataset	120
Figure C.5	Variations Among 10 Tests (X_{HG}) - Wind Turbine Dataset . . .	123
Figure C.6	Variations Among 10 Tests (X_{HG}) - Steam Production Dataset .	126
Figure C.7	Variations Among 10 Tests (X_{HG}) - Energy Efficiency Dataset .	129
Figure C.8	Variations Among 10 Tests (X_{HG}) - Synchronous Motors Dataset	132
Figure C.9	Variations Among 10 Tests (X_{HD}) - Wind Turbine Dataset . . .	135
Figure C.10	Variations Among 10 Tests (X_{HD}) - Steam Production Dataset .	138
Figure C.11	Variations Among 10 Tests (X_{HD}) - Energy Efficiency Dataset .	141
Figure C.12	Variations Among 10 Tests (X_{HD}) - Synchronous Motors Dataset	144

LIST OF TABLES

Table 4.1	GAN Neural Network Configuration Table	52
Table 4.2	Privacy Metrics Performance Analysis Table	60
Table 4.3	Privacy Preservation Methods Experiments Table	71
Table 4.4	Vanilla GAN vs. CTGAN Performance Comparison Table (Mean Values)	72
Table 4.5	Feature Prediction Performance Results	74
Table 4.6	Test Results: Wind Turbine and Steam Production	78
Table 4.7	Test Results: Energy Efficiency and Synchronous Motors	79
Table 4.8	PCA Reconstruction Errors	85
Table 4.9	Hidden Sensitive Data - Case Study R -Squared Values	90
Table A.1	Experiments in Chapter 4.5 - Accuracy Values	105
Table A.2	Experiments in Chapter 4.5 - P_{MSE} , P_{VAR} , and P_{HAUS} Metrics .	106
Table A.3	Experiments in Chapter 4.5 - P_{KL} , P_{PRO} , and P_{PCC} Metrics . .	107
Table B.1	Experiments in Chapter 4.6 - Accuracy Values	108
Table B.2	Experiments in Chapter 4.6 - P_{MSE} , P_{VAR} , and P_{HAUS} Metrics .	109
Table B.3	Experiments in Chapter 4.6 - P_{KL} , P_{PRO} , and P_{PCC} Metrics . .	110
Table C.1	Experiments in Chapter 4.7 - GAN + DP - Wind Turbine	112
Table C.2	Experiments in Chapter 4.7 - GAN + DP - Wind Turbine	113
Table C.3	Experiments in Chapter 4.7 - GAN + DP - Steam Production . .	115
Table C.4	Experiments in Chapter 4.7 - GAN + DP - Steam Production . .	116
Table C.5	Experiments in Chapter 4.7 - GAN + DP - Energy Efficiency . .	118
Table C.6	Experiments in Chapter 4.7 - GAN + DP - Energy Efficiency . .	119
Table C.7	Experiments in Chapter 4.7 - GAN + DP - Synchronous Motors	121
Table C.8	Experiments in Chapter 4.7 - GAN + DP - Synchronous Motors	122
Table C.9	Experiments in Chapter 4.7 - GAN Only - Wind Turbine	124
Table C.10	Experiments in Chapter 4.7 - GAN Only - Wind Turbine	125
Table C.11	Experiments in Chapter 4.7 - GAN Only - Steam Production . .	127
Table C.12	Experiments in Chapter 4.7 - GAN Only - Steam Production . .	128
Table C.13	Experiments in Chapter 4.7 - GAN Only - Energy Efficiency . .	130

Table C.14	Experiments in Chapter 4.7 - GAN Only - Energy Efficiency . . .	131
Table C.15	Experiments in Chapter 4.7 - GAN Only - Synchronous Motors .	133
Table C.16	Experiments in Chapter 4.7 - GAN Only - Synchronous Motors .	134
Table C.17	Experiments in Chapter 4.7 - DP Only - Wind Turbine	136
Table C.18	Experiments in Chapter 4.7 - DP Only - Wind Turbine	137
Table C.19	Experiments in Chapter 4.7 - DP Only - Steam Production . . .	139
Table C.20	Experiments in Chapter 4.7 - DP Only - Steam Production . . .	140
Table C.21	Experiments in Chapter 4.7 - DP Only - Energy Efficiency	142
Table C.22	Experiments in Chapter 4.7 - DP Only - Energy Efficiency	143
Table C.23	Experiments in Chapter 4.7 - DP Only - Synchronous Motors . .	145
Table C.24	Experiments in Chapter 4.7 - DP Only - Synchronous Motors . .	146
Table D.1	Experiments in Chapter 4.9 - GAN + DP - Wind Turbine	147
Table D.2	Experiments in Chapter 4.9 - GAN + DP - Wind Turbine	148
Table D.3	Experiments in Chapter 4.9 - GAN + DP - Steam Production . .	149
Table D.4	Experiments in Chapter 4.9 - GAN + DP - Steam Production . .	150
Table D.5	Experiments in Chapter 4.9 - GAN + DP - Energy Efficiency . .	151
Table D.6	Experiments in Chapter 4.9 - GAN + DP - Energy Efficiency . .	152
Table D.7	Experiments in Chapter 4.9 - GAN + DP - Synchronous Motors	153
Table D.8	Experiments in Chapter 4.9 - GAN + DP - Synchronous Motors	154
Table D.9	Experiments in Chapter 4.9 - GAN Only - Wind Turbine	155
Table D.10	Experiments in Chapter 4.9 - GAN Only - Wind Turbine	156
Table D.11	Experiments in Chapter 4.9 - GAN Only - Steam Production . .	157
Table D.12	Experiments in Chapter 4.9 - GAN Only - Steam Production . .	158
Table D.13	Experiments in Chapter 4.9 - GAN Only - Energy Efficiency . .	159
Table D.14	Experiments in Chapter 4.9 - GAN Only - Energy Efficiency . .	160
Table D.15	Experiments in Chapter 4.9 - GAN Only - Synchronous Motors .	161
Table D.16	Experiments in Chapter 4.9 - GAN Only - Synchronous Motors .	162
Table D.17	Experiments in Chapter 4.9 - DP Only - Wind Turbine	163
Table D.18	Experiments in Chapter 4.9 - DP Only - Wind Turbine	164
Table D.19	Experiments in Chapter 4.9 - DP Only - Steam Production . . .	165
Table D.20	Experiments in Chapter 4.9 - DP Only - Steam Production . . .	166

Table D.21	Experiments in Chapter 4.9 - DP Only - Energy Efficiency	167
Table D.22	Experiments in Chapter 4.9 - DP Only - Energy Efficiency	168
Table D.23	Experiments in Chapter 4.9 - DP Only - Synchronous Motors . .	169
Table D.24	Experiments in Chapter 4.9 - DP Only - Synchronous Motors . .	170



LIST OF SYMBOLS

$\mathcal{A}$	Randomized algorithm
b	Parameter in Laplace equation
cov	Covariance
$d(x, y)$	Euclidean distance between x and y
D_{loss}	Loss function
D_{output}	Output of the D for the combined real and generated samples
$D_{outputGenerated}$	Output of the D for the generated samples
E	Expected value
$\exp$	Exponential function
f	Function
F	Feature of a database
$g(z)$	Sigmoid Function, z is a linear combination of features and weights
G_{loss}	Loss function
$h_{\theta}(x)$	Hypothesis Function
$input_{cols}$	List of input columns
k	Number of features in lower dimension
l_1	Distance
Lap	Laplace function
m	Number of observations
n	Number of features
n_s	Number of sensitive features
$\mathbb{N}$	Domain of all non-negative integers, including zero
O	Computational cost
P	Probability distribution
P_{ALL}	Privacy Metrics - General
P_{HAUS}	Privacy Metric- Directed Hausdorff
P_{KL}	Privacy Metric - Kullback-Leibler Divergence
P_{MSE}	Privacy Metric - Mean Squared Error
P_{PCC}	Privacy Metric - Pearson's Correlation Coefficient

P_{PRO}	Privacy Metric - Procrustes
P_{VAR}	Privacy Metric - Normalized Variance
p_z	Probability distribution of random noise z
Q	Probability distribution
$\mathcal{R}$	Range
$\mathbb{R}$	Set of real numbers
$randn$	Random number
S	Probability space
$\mathcal{S}$	Subset
S_Φ	Subset of selected private features
T	Orthogonal transformation matrix
$Table_{cols\&preds}$	Input columns generates highest <i>R-Squared</i> for each feature of X_O .
$Time_{collect}$	Duration of data collection
$Time_{execute}$	Execution time of the proposed method operation
tr	Trace operator
V	Eigenvectors
W	PCA space
X	Input Features
X_D	DP applied dataset
X_G	Synthetic Dataset
X_{HGD}	Proposed method dataset (GAN+DP)
X_{HG}	Proposed method dataset (GAN only)
X_{HD}	Proposed method dataset (DP only)
X_O	Original Dataset
X_{Per}	Perturbated dataset
$X_{reduced}$	Reduced dataset
$\hat{X}_O$	Reconstructed X_O
$\tilde{X}$	Mean-centered data
y	Target Feature
y_{col}	Target column
y_{pred}	Predicted output (a scalar between 0 and 1)

y_{true}	True label (0 or 1)
α	Scalar parameter
δ	Probability of DP failure
ϵ	Level of privacy in a DP
λ	Eigenvalues
μ	Mean of all samples
σ	Standard deviation
σ^2	Statistical variance
Σ	Covariance matrix
θ	Logistic Regression Learning Parameter

LIST OF ACRONYMS AND ABBREVIATIONS

Acc	Accuracy
AI	Artificial Intelligence
BCE	Binary Cross-Entropy
CPU	Central Processing Unit
CTGAN	Conditional Tabular GAN
D	Discriminator
DA	Data Anonymization
DCGAN	Deep Convolutional GAN
DL	Deep Learning
DP	Differential Privacy
G	Generator
GAIL	Generative Adversarial Imitation Learning
GAN	Generative Adversarial Networks
GDPR	General Data Protection Regulation
IoT	Internet of Things
IIoT	Industrial Internet of Things
IMDb	International Movie Database
ML	Machine Learning
NDA	Non-Disclosure Agreement
NN	Neural Network
OS	Operating System
PCA	Principal Component Analysis
PP Methods	Experimental Exploration of Privacy Preservation Methods
RF	Random Forest
RAM	Random Access Memory
SVM	Support Vector Machine
WEC	Wind Energy Converter
WGAN	Wasserstein GAN

1. INTRODUCTION

The emerging field of Artificial Intelligence (AI) and its prominent subset, Machine Learning (ML), have profoundly impacted our daily lives and businesses. ML systems are the subject of extensive research, with increasing applications in various domains, including the Internet of Things (IoT), health, and finance. These innovative solutions have found their way into numerous aspects of our private and professional lives, significantly transforming our operations. AI excels in image and voice recognition, language processing, information presentation, ML algorithms, and expert systems. The widespread adoption of these services necessitates the utilization of vast amounts of data, and as AI continues to evolve, its data processing capabilities are becoming increasingly refined. For instance, the availability of free face image processing applications has led to a surge in user subscriptions, showcasing the appeal and prevalence of AI-powered services in our everyday routines.

Over the last decade, cloud-based solutions have become increasingly popular as storage and computation platforms for ML algorithms, with cloud platforms storing data for individuals and institutions that use these facilities. As a result, cloud platforms not only gather but also generate private data. Health data are typically considered confidential, but when patients seek to determine their likelihood of developing cancer-based on genetic information, highly accurate estimations can be made using Cloud ML prediction models. However, this process exposes the patient's genetic information and the model's cancer prediction outcome to potential risks. Similarly, health insurance companies may collect similar data to assess the risk level of their customers.

Financial institutions may use credit applicants' social media accounts in the fintech industry to determine their risk level. With increasing data sources, ML functions can make inferences using trained models, resulting in more accurate results. The

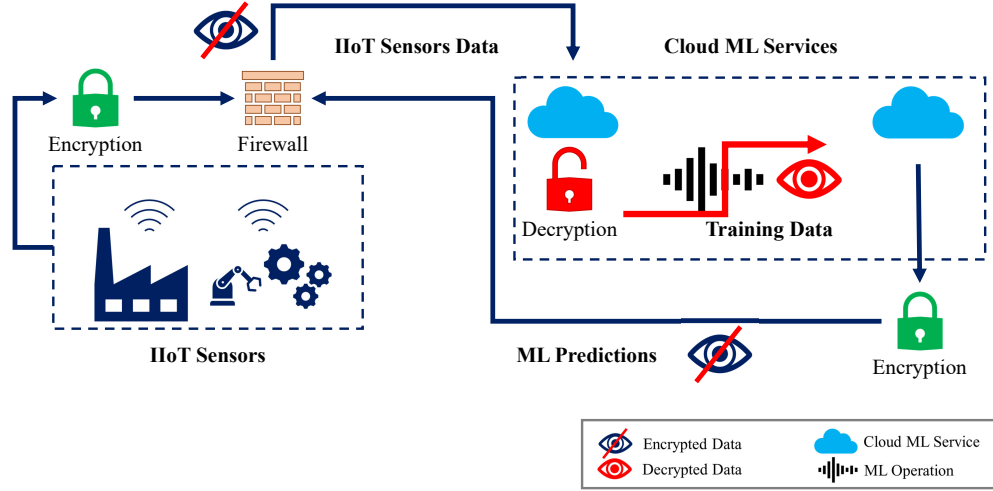


Figure 1.1 Conceptual representation of data transmission to ML

manufacturing industry also utilizes ML prediction to monitor industrial production processes through sensor devices. Internet of Things (IoT) technology enables the collection of massive amounts of data to forecast production tool aging or failures. Examples of such applications include equipment failure detection, supply chain optimization, performance monitoring, and predictive maintenance (Fernandes, Corchado, and Marreiros 2022). IoT serves as an infrastructure connecting the physical world of company processes with the Internet (Bisio et al. 2018), but IoT devices typically have limited computing power and small memories (Dalipi et al. 2016). Data analysis is often performed offline using cloud-based services. Still, data transfer from production to processing on the cloud can be a significant bottleneck, with the transmitted data potentially vulnerable to privacy leaks (X. Xu et al. 2018; Tahsien, Karimipour, and Spachos 2020). Additionally, IoT data may contain implicit or explicit knowledge of specific production areas gained from experience, which is critical to unlocking new opportunities and revealing how industries operate. However, recent cyber-security issues have highlighted the need for strict privacy protocols to maintain IoTs' competitive advantage and profitability (Bakhshi, Balador, and Mustafa 2018; Micro 2021). As Figure 1.1 shows, secure data transmission to third-party entities can be achieved through encryption. However, there are instances where ML applications require access to decrypted data to

perform optimally, which may lead to third parties gaining unwarranted access to the sensitive information contained in IoT data that captures details of the production process. Figure 1.1 presents a conceptual representation of data transmission from a manufacturing area where Industrial IoT (IIoT) sensors collect data at regular intervals. In this abstract depiction, the collected data undergo encryption before leaving their source and are transmitted in encrypted form to the cloud platform for ML operations. However, during the ML training process, the data must be decrypted to facilitate the optimal training of the model. Once the ML training is completed, the predictions can be encrypted on the cloud platform before being transmitted back to the IIoT owner company. Since the ML training process requires decrypted data, it results in the exposure of the entire dataset, thereby exposing sensitive information to potential breaches.

A typical ML process splits the dataset into training and test subsets for X and y . X represents the input features or independent variables, and y represents the target or dependent variable we want to predict. More specifically, X is a matrix of size (m samples, n features), where m samples is the number of instances or samples in the dataset, and n features is the number of input features or variables for each sample. On the other hand, y is a vector of size (m samples, 1), where each element of the vector represents the target value or label for the corresponding instance or sample in X . We can call X as the input features or independent variables, and y as the target variable or dependent variable.

In an IoT dataset, we typically have multiple sensors or devices that collect data from various sources, such as temperature, pressure, vibration, raw material usage, and other environmental factors. Although it depends on the specific data and the relationships between the features, it is possible to use IoT data to predict other features, such as raw material usage. In general, ML models are trained to learn the patterns and relationships between the input features and the target variable, so if there is a relationship between the IoT data and the raw material usage, it may be possible to train a model to predict the raw material usage based on the

IoT data. This allows us to learn the relationships and interactions between the selected features and the target variable and make accurate predictions on new, unseen, hidden-sensitive data. It can cause a considerable risk, such as getting a forecast of a critical raw material may give a competitive advantage to rivals. It may lead to disclosing a production recipe.

We can observe numerous applications of AI and ML in our daily lives, and their usage is expected to increase exponentially. However, these applications can potentially access and analyze our data without our consent, making it imperative to prioritize the safe use of private data. While cryptographic solutions exist for data protection, they often entail high computational costs and bandwidth requirements and may be limited as ML typically requires decrypted data. In addition to employing cryptographic solutions for transmission security, perturbation-based methods can be utilized to ensure the processing-time safety of private data.

In this thesis, we aimed to develop a method that performs both privacy preservation and high performance on ML predictions. We also sought to protect the case of hidden sensitive data. Protecting private data and using public ML services require novel approaches beyond traditional encryption and anonymization methods. In Chapter 2, we have discussed the available studies on data protection. As initial steps, we developed a straightforward algorithm to add noise to a dataset for privacy preservation and tested it on well-known datasets such as Iris and Pima Diabetes. Chapter 3 explains what and how to accomplish steps to developing a novel method. Chapter 4 also provides detailed accounts of the experiments conducted, offering an in-depth understanding of the study's methodology and outcomes.

2. LITERATURE REVIEW

Data privacy has become a significant concern in recent years, particularly with the rapid advancements in AI technology. As data security grows, it is essential to address privacy issues effectively. While numerous studies have explored data protection in AI, the focus has predominantly revolved around infrastructure and sensor device security. However, it is crucial to extend this research to encompass the domain of AI, where data privacy concerns are particularly prominent. In the context of AI, the extensive analysis of attack types related to IoT by Tahsien and colleagues is essential to acknowledge (Tahsien, Karimipour, and Spachos 2020). They identified passive attacks as a significant threat involving an unauthorized collection of victims' data without their consent. Another crucial aspect they highlighted is privacy leakage, which occurs when private IoT data is shared with third parties without the knowledge or awareness of the data owners. By delving deeper into the intersection of AI and IoT, we can explore how AI can play a pivotal role in addressing these privacy challenges. By developing advanced algorithms and techniques, AI can enhance the security and privacy of IoT devices, ensuring that sensitive data remains protected.

In recent years, the exponential growth of data and advancements in AI have brought forth significant concerns regarding data privacy. As AI technology continues to permeate various domains, including the IoT, addressing the privacy challenges associated with IoT devices becomes imperative. The widespread adoption of IoT solutions has made the cloud a favored platform for data storage and ML operations within this ecosystem (Samie, Bauer, and Henkel 2019). However, this dependence on the cloud introduces new security risks, particularly concerning IoT devices' large-scale data mining activities when interacting with cloud services (Stout and Urias 2016).

While considerable research and proposals exist to safeguard stored IoT data in the cloud, the majority of these efforts focus on encryption techniques for data at rest and establishing secure connections between IoT sensors and cloud storage (Qi et al. 2020), (Sharma et al. 2021), (X. Xu et al. 2018), (Samie, Bauer, and Henkel 2019). However, it is essential to note that the data in the cloud needs to be decrypted during ML operations. This decryption process poses privacy risks, prompting data owners to sign Non-Disclosure Agreements (NDAs) with cloud providers as a legal preventative measure. Despite such precautions, a high privacy risk remains, as empirical studies have shown that machine-learning-as-a-service platforms can lead to data leakage, even within their training datasets (Shokri et al. 2017). Alternative approaches to cloud data security have been proposed to mitigate these risks. One such technique involves sending partially processed data to the cloud instead of raw data. Researchers have suggested employing Deep Learning (DL) operations to ensure irreversibility and enhance privacy protection. Based on the irreversibility of partially processed Neural Network (NN) computations, this methodology presents a unique perspective on IoT-Cloud data security. The limitations and inconveniences associated with homomorphic encryption techniques, as discussed by Jeong and colleagues, are also acknowledged (Jeong, Lee, and Moon 2017). As IoT devices continue to generate and exchange vast amounts of data, ensuring robust data privacy in the context of AI and the IoT remains a critical concern. While encryption and secure connections play crucial roles in protecting IoT data, exploring novel techniques such as partial processing and irreversibility can strengthen the security and privacy of IoT data stored and processed in the cloud.

The usage of Generative Adversarial Networks (GAN) has been shown by Hitaj and colleagues to potentially compromise DL methods and potentially result in the disclosure of privacy-sensitive information (Hitaj, Ateniese, and Perez-Cruz 2017). Researchers have explored the distributed/federated learning approach to address such security concerns. Zeng and colleagues introduced a comprehensive framework for federated learning, specifically designed to ensure privacy and security in the context of data (Zeng et al. 2023). However, Qi and colleagues pointed out that

this approach may not be suitable for IIoT platforms due to the high rate at which data is generated, potentially leading to a waiting queue when collecting data among clusters (Qi et al. 2020).

In the existing literature, several approaches have been proposed to leverage encrypted data in ML operations, including homomorphic encryption (Gilad-Bachrach et al. 2016) and cryptographic techniques (McMahan et al. 2017). However, comparing these techniques with standard ML methods that process decrypted data can result in significant computational costs. The homomorphic encryption approach aims to maintain data in an encrypted form while performing ML prediction operations using DL algorithms. Although this technique enables ML operations on encrypted datasets, it still requires decrypted data for training the underlying patterns (Gilad-Bachrach et al. 2016), (Jeong, Lee, and Moon 2017).

Cryptographic techniques are secure multiparty computations that utilize DL methods for distributed or federated learning. However, establishing and implementing a distributed computing environment may prove prohibitively expensive in industrial production settings (Long et al. 2021). In addition, sensors' rapid generation of records necessitates fast and efficient processing (Qi et al. 2020). Furthermore, DL models often need stronger explainability due to their black-box nature, posing challenges for industrial data analysis where interpretability is crucial (Islam et al. 2022).

It is worth noting that even in secure cloud systems, recent hardware-centric cyber-attack techniques have emerged, focusing on exploiting cache-based side channels such as Meltdown and Spectre. These attacks aim to extract sensitive information from shared computational resources (Sari, Demir, and Kucuk 2019). Therefore, ensuring data privacy and security in cloud environments requires robust protection against side-channel attacks. While approaches like homomorphic encryption and cryptographic techniques offer potential solutions for maintaining data privacy in ML operations, there are trade-offs regarding computational costs, scalability,

explainability, and vulnerability to hardware-centric cyber-attacks.

The following subsections provide detailed explanations of the implementations of data protection using DP, GAN, and PCA methods.

2.1 DP implementations with Data Protection approach

In 2006, Cynthia Dwork (Dwork 2006) introduced a technique known as Differential Privacy (DP) that enables the collection of information without compromising privacy. DP utilizes plausible deniability to generate synthetic data that provides formal privacy guarantees. Specifically, DP adds noise to the data to prevent the identification of individual records.

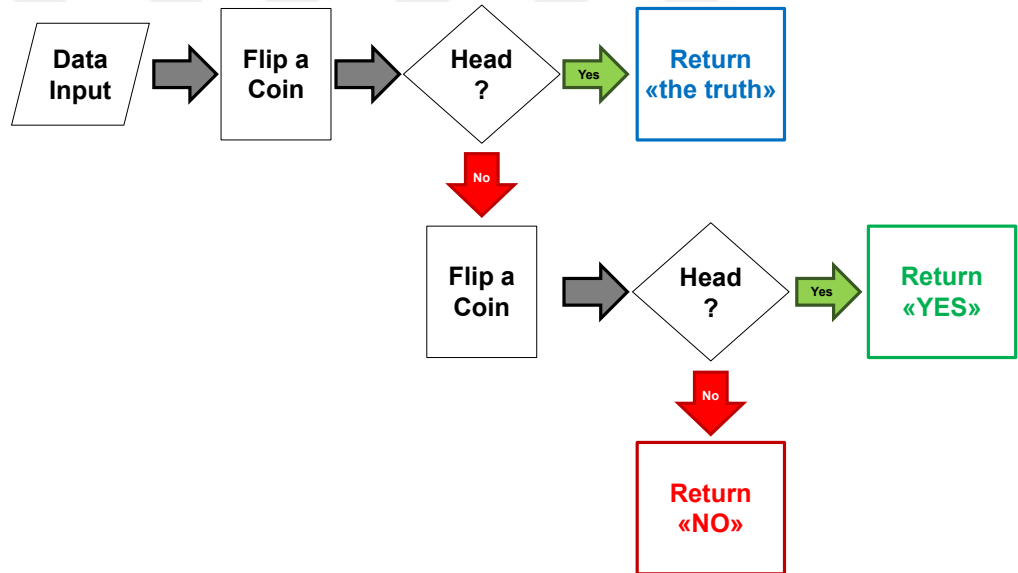


Figure 2.1 Plausible Deniability

To illustrate how DP works, consider a poll service that asks participants whether they vote for political party A, with YES and NO buttons available as responses. The DP algorithm flips a coin before sending their response to the database. If the coin lands heads, the real answer is sent to the server; otherwise, the algorithm flips a second coin, sending YES if it lands tails or NO if it lands heads. While the answers can be seen on the database, no one can trust individual records since the algorithm adds some noise. For instance, the record for a participant may state that

they support Party A, but there is at least a one in four chance that they do not. This is plausible deniability (Figure 2.1), and it is especially useful for collecting data on sensitive behaviors such as drug use.

DP differs from Data Anonymization (DA) techniques. While there is ongoing research on DA, such as the work presented in (Majeed and Lee 2020), it is important to note that DA primarily focuses on removing individual names or identities from datasets, which can create vulnerabilities against linkage attacks.

Linkage attacks pose a significant threat to the effectiveness of DA techniques. These attacks exploit the possibility of combining seemingly anonymous data from different sources to reveal the real identities of individuals. In 1997, Sweeney demonstrated that even after anonymization, the remaining data could still be potentially disclosed through a linkage attack (Sweeney 1997). Remarkably, a related study showcased the possibility of identifying 87% of all Americans using just three pieces of information: zip code, birthday, and gender. Furthermore, in 2008, it was proven that people could be identified from the supposedly anonymized Netflix dataset published in 2006 by leveraging publicly available user data from the International Movie Database (IMDb) (Narayanan and Shmatikov 2006).

These examples underscore the limitations of relying solely on data anonymization for privacy protection, as linkage attacks can compromise the confidentiality of supposedly anonymous data. As a result, alternative approaches such as DP, which incorporates random noise injection, have gained prominence as a more powerful means of preserving privacy in data sharing and analysis.

Husnoo and colleagues conducted a survey in 2021 on privacy preservation in IoT-enabled systems, discussing the application and implementation of DP in IoT infrastructures (Husnoo et al. 2021). In another survey from 2021, Jiang and colleagues explored DP implementation alternatives specifically for IIoT environments, summarizing them into seven groups: IoT-related industrial logistics, smart cities, bioengineering, autonomous vehicles, intelligent manufacturing, blockchain in in-

dustrial IoT, and social networks in industrial IoT (Jiang et al. 2021). While each group warrants detailed investigation, it is evident that the need for data protection has emerged as a critical topic, with DP being one of the appealing approaches. However, DP in the context of IIoT is still in the research phase, particularly in addressing latency reduction to enhance overall efficiency (Arachchige et al. 2020; Jiang et al. 2021). Several concerns surrounding DP for IoT applications include the complexity of integrating business and industrial systems, managing big data, addressing real-time constraints, optimizing privacy preservation, and developing industrial applications (Jiang et al. 2021). Researchers like Yang and his team focused on confirming the data utility of DP implementation in IoT (Yang et al. 2017), while Liang and his team proposed an edge computing-based deep learning method that relocates deep learning processes between cloud servers and edge solutions (Liang et al. 2020). Xu and colleagues presented GANobfuscator, an approach that utilizes GAN and DP to mitigate information leakage in IoT systems (C. Xu et al. 2019). The IoT environment inherently contains sensitive data, and some data owners are reluctant to share this data with cloud platforms due to legitimate concerns. Load balancing of DP application across nodes in edge training remains an open challenge (Jiang et al. 2021). Feng and colleagues introduced an edge computing solution for privacy preservation (Feng et al. 2020), while Usman and colleagues developed RaSEC, an intelligent framework for data protection (Usman, Jolfaei, and Jan 2020). Real-time implementation of DP algorithms poses a problem for IIoT, requiring fast computing and transmission capabilities. Langarica and colleagues investigated fault identification in industrial motors using fast computing IIoT approaches (Langarica, Ruffelmacher, and Núñez 2019). Preserving cloud data privacy is one of the primary challenges in IIoT, with Hu and colleagues proposing a fog computing framework as a potential solution (Hu et al. 2017). Alternative studies have also explored fog computing in this context (Feng et al. 2020; X. Xu et al. 2020). Our approach, employing GAN and DP techniques, offers a novel perspective on addressing this problem.

DP algorithms use the Laplace distribution to spread the data wider and increase

anonymity levels. Despite the benefits, DP is not widely used in practice, with companies only using it for a small percentage of their data collection. When DP is used on a small dataset, the result may be inaccurate due to the added noise. Additionally, DP requires the identification of private information to be effective, and the implementation can be complex, especially for high-dimensional datasets that require additional computation resources.

While DP is a promising approach for preserving data privacy, it has limitations. Further research is needed to refine the approach and overcome its implementation challenges, particularly for high-dimensional datasets. Additionally, for some sensitive data, such as specific health-related information, adding noise may not be sufficient to prevent identity disclosure. In such cases, a synthetic copy of the confidential data can be created to mitigate the risk of disclosure. This approach can provide more effective privacy protection, particularly for data that could reveal sensitive personal information. While DP is a valuable technique, it is important to consider other privacy-preserving approaches, such as synthetic data generation, to ensure comprehensive privacy protection.

2.2 GAN implementations with Data Protection approach

In 2014, Goodfellow and colleagues introduced the concept of a GAN as an adversarial process consisting of two models: the Generator (G) and the Discriminator (D) (Goodfellow et al. 2014). The G aims to capture the data distribution, while the D distinguishes between real and synthetic samples produced by the G . The authors discuss the impact of the number of layers and neurons in the G and D networks on the performance of the GAN model. They explain that the G takes a random noise vector as input, typically with 100 dimensions, enabling it to learn complex patterns in the data. However, the dimensions of the noise vector can be adjusted based on the dataset's complexity and the specific task.

Figure 2.2 illustrates the operation of a GAN in which handwritten digits of the

MNIST dataset are used to train D and G Models. The MNIST dataset is a widely used benchmark dataset for image recognition tasks, consisting of 60,000 training images and 10,000 test images of handwritten digits from zero to nine. During the training phase of the GAN, the D model is trained on the MNIST training data to distinguish between real and synthetic images. The G model is also trained on the MNIST training data to generate synthetic images that are similar to the real images. The D and G models are trained in an adversarial manner, where the G model tries to generate images that fool the D model, and the D model tries to distinguish between real and synthetic images. This adversarial training process continues until the G model can generate synthetic images that are indistinguishable from real images. In the prediction phase, the D model is used to classify handwritten digits in the MNIST test set, as shown in Figure 2.2. In this example, handwritten digits 6 and 2 are shown as test input data on the left side of the D model, and the model's predictions are shown on the right side. This prediction process is similar to a normal ML prediction process, where a trained model is given input data and produces an output. On the other hand, the G model is used to generate synthetic images that are similar to the real images, as shown on the left side of the model in Figure 2.2. These synthetic images are generated from random noise input, and the G model is trained to produce images that are similar to the MNIST training data. The output of the G model is then compared to the output of the D model to ensure that the synthetic images generated by the G model are indistinguishable from real images.

Furthermore, the authors highlight the importance of balancing the G and D . If the G is too weak, it may produce poor samples, whereas an overpowering D can dominate the G . The authors suggest increasing the number of layers and neurons in the G to improve its performance and decreasing the same for the D . They emphasize the need for careful balancing to achieve optimal performance in GAN models.

Radford and associates introduce the Deep Convolutional GAN (DCGAN) in their 2015 paper and analyze network architecture modifications' impact on synthesized

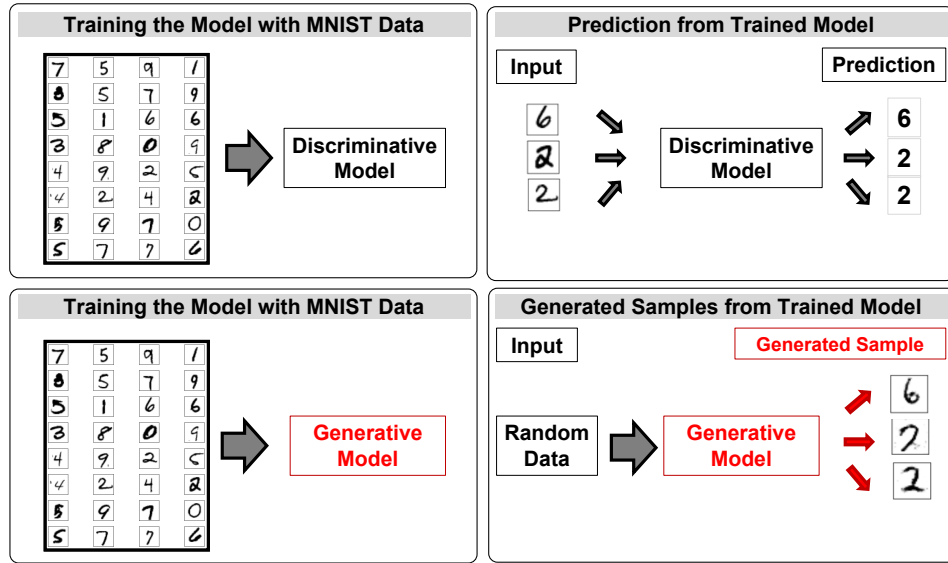


Figure 2.2 Generative Adversarial Networks

images' quality (Radford, Metz, and Chintala 2015). They note that increasing the number of neurons in both the G and D can result in a more powerful GAN that captures underlying data patterns and generates realistic artificial data. However, augmenting both models can increase the risk of overfitting the training data, leading to poor performance on new data. To address this challenge, the authors propose architectural principles for designing DCGAN that strike a balance between these concerns, ensuring stable and high-quality GAN training.

Arjovsky and colleagues discuss the calibration of D and G models in GAN in their 2017 paper, introducing the Wasserstein GAN (WGAN) (Arjovsky, Chintala, and Bottou 2017). They explore the delicate balance between the D 's capacity and the overall stability of the GAN. Increasing the number of neurons in the D can improve its ability to distinguish real and fake data, but an overly powerful D may reject actual data, compromising GAN performance. To overcome this issue, the authors propose a novel metric for measuring the distance between real and synthetic data distributions, leading to more reliable and stable GAN training outcomes.

Brock and their team present a method for training GAN on a large scale in their 2018 paper and examine the balance between the G 's capacity and the diversity of

its output (Brock, Donahue, and Simonyan 2018). They argue that increasing the number of neurons in the G enables it to learn complex data patterns and produce diverse synthetic data. However, an excessively complex G may generate data that closely resembles the training data, lacking diversity in its outputs. To address this, the authors suggest introducing a regularization term in the GAN loss function, encouraging the G to produce more varied outcomes.

Furthermore, GAN has been employed in several studies as a means of privacy protection by generating synthetic versions of original data (Choi et al. 2017; Beaulieu-Jones et al. 2019; Li et al. 2020; Han and Xue 2021).

In addition to the GAN algorithm, alternative methods exist for generating synthetic data that adhere to statistical properties. Ping and colleagues demonstrated using Bayesian correlation in a data synthesizer to achieve this (Ping, Stoyanovich, and Howe 2017). However, the primary advantage of employing GAN over other statistical models is its effectiveness in approximating the true data distribution. Xu and colleagues showcased the capability of GAN to generate high-quality synthetic data, benefiting the field of data science (Xu and Veeramachaneni 2018). They introduced the synthetic Conditional Tabular GAN (CTGAN) specifically for generating tabular data, demonstrating superior performance in learning accurate distributions compared to Bayesian solutions (L. Xu et al. 2019).

While GAN usage for data protection in the IoT remains limited, some notable instances exist in the literature. Huang and his team mentioned using GAN for protecting medical data in the IoT context (Huang et al. 2021). The authors highlighted the scarcity of studies on privacy preservation in industrial IoT. They employed Generative Adversarial Imitation Learning (GAIL), a variant of GAN combined with backward reinforcement learning, as an alternative method. Another study by Shahid and associates utilized GAN for IoT network traffic generation, particularly for generating categorical data, and integrated it with a WGAN autoencoder (Shahid et al. 2020).

2.3 GAN and DP Hybrid implementations

There are hybrid approaches that combine GAN and DP for privacy preservation. However, to the best of our knowledge, there needs to be more research on applying these approaches to IoT-enabled systems (Husnoo et al. 2021). For instance, Xu and colleagues introduced GANobfuscator, a technique that utilizes GAN and DP to protect the privacy of data types rich in semantics, such as images, text, audio, and video (C. Xu et al. 2019). Liu and his team proposed a method that combines DP and GAN to safeguard sensitive data, including patient medical records (Liu et al. 2019).

Studies addressing hidden sensitive data focus on data mining rule associations. Cheng and associates emphasized the potential for confidential data disclosure through proper mining and proposed a rule-hiding approach (Cheng et al. 2016). In the early stages of research on avoiding association rules, data sanitization was employed as a method (Atallah et al. 1999). Jones and colleagues presented a method for protecting clinical medical data by leveraging the combination of GAN and DP (Beaulieu-Jones et al. 2019).

2.4 Hidden Sensitive implementations

In addition to the mentioned data-related risks, it is crucial to consider the potential presence of overt and covert sensitive information within the data. It is particularly relevant when specific subsets of features extracted from the data, in combination with suitable ML models, can reveal sensitive data. Malekzadeh and associates addressed these privacy threats in cloud-assisted applications utilizing raw sensor data from personal wearable or portable devices (Malekzadeh et al. 2020). However, it is crucial to acknowledge that similar threats also exist in industrial systems. For instance, in an industrial setting, raw data from Industrial Internet of Things (IIoT) devices related to material consumption can inadvertently disclose production costs, including profit margins, and expose confidential product recipes.

Moreover, insights into the structure of the production area can be inferred from IIoT data, potentially providing competitors, attackers, or potential investors with knowledge about production bottlenecks or areas that can be optimized. These concerns are significant deterrents, preventing companies from sharing their data with third parties, impeding collaboration efforts, and hindering the potential for data-driven innovation. Disclosing direct or hidden sensitive data within datasets entails substantial risks that extend beyond personal wearable or portable devices to industrial systems, where the exposure of proprietary information and potential competitive advantages can harm businesses. Protecting data privacy and establishing secure mechanisms for data sharing are essential for fostering trust, enabling collaboration, and fully realizing the potential of data in driving innovation and industrial advancements.

Sharing a dataset intended for a specific use case, such as predictive maintenance, can lead to extracting different predictions or insights for other purposes. It is possible because the dataset contains valuable information about various aspects of a company's operations. By applying different algorithms or models to the shared dataset, it becomes feasible to uncover additional knowledge and make predictions in diverse domains. For example, a dataset may contain details about machinery, operating conditions, and maintenance records for predictive maintenance purposes. However, it may also contain information about raw material usage, production recipes, or other relevant variables. Through thorough data exploration and analysis, relationships, patterns, and correlations can be discovered, enabling the generation of predictions in different areas.

This utilization of shared data allows companies to effectively leverage their existing data resources, extracting maximum value by employing AI and ML techniques to derive insights and make predictions across multiple operational domains. Nevertheless, valuable information within the data, including hidden-sensitive data, poses a significant security risk. Data mining malware attacks represent malicious activities in which attackers target ML training data to extract sensitive information

or manipulate the learning process. Shokri and his team highlighted membership inference attacks as a potential source of information leakage (Shokri et al. 2017). Similarly, studies by Khosravy and colleagues (Khosravy et al. 2022) and Fredrikson and colleagues (Fredrikson et al. 2014) have emphasized the risks associated with model inversion attacks on patients’ genomic privacy. These studies underscore the risks associated with sharing data for one purpose while highlighting the potential for deriving additional predictions or insights from such data.

2.5 Dimension Reduction

Dimension reduction serves as an alternative approach to privacy preservation, and one commonly employed technique is Principal Component Analysis (PCA). PCA has found wide application in various domains, including data privacy and security, with several articles and research papers delving into its use for privacy preservation. The objective of these studies is to leverage PCA while ensuring the privacy of sensitive data (Fan et al. 2021), (Wang and Chang 2018).

The utilization of PCA involves reducing the dimensionality of the data while preserving essential statistical properties. This reduction in dimensionality contributes to the obfuscation of sensitive information present in the original dataset, thereby reducing its vulnerability to privacy breaches. By transforming the data into a lower-dimensional space while retaining key characteristics, PCA aids in mitigating the risk of exposing sensitive information, thus safeguarding individuals’ privacy.

PCA reconstruction is a technique of inverting transformation using the retained principal components and helps us approximate the original dataset (Sang, Shen, and Tian 2011).

3. METHODOLOGY

Our research aimed to achieve two objectives in ML predictions: safeguarding privacy and achieving high performance. To accomplish this, we developed several algorithms that effectively balanced these goals. We incorporated perturbation techniques that introduced differentiation in the original dataset to protect privacy. Additionally, we explored the creation of synthetic data and applied the Laplace method of data privacy protection. Our methodology for achieving these objectives is discussed in detail in this chapter. We also tested the hidden-sensitive case on our proposed methods. The following section presents our findings and numerical results from the experiments. Overall, our algorithms effectively balanced the goals of privacy and performance, resulting in accurate predictions while protecting sensitive information.

3.1 Effects of Perturbation

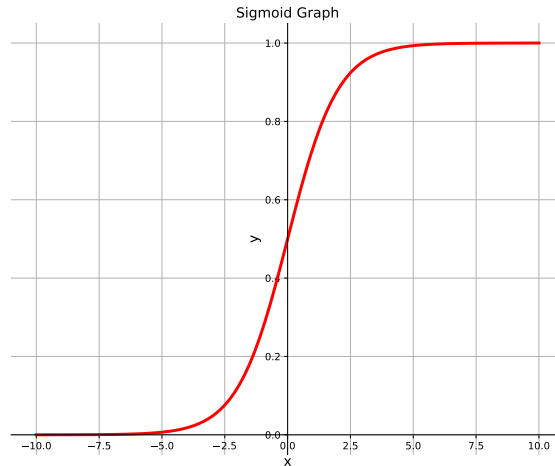


Figure 3.1 The Sigmoid function.

As an initial exploration, our primary aim was to devise a simple algorithm that could effectively introduce noise into a dataset to preserve privacy. We employed the widely used Iris dataset and implemented the Logistic Regression Classification technique to showcase this approach. As illustrated in Figure 3.1, the sigmoid

function is a commonly used mathematical function that depicts the relationship between the Logistic Regression algorithm and the dataset. This function transforms the raw input data into a probability score, which can be used to categorize the data. The sigmoid function has a distinctive S-shaped curve and is characterized by a range of 0 to 1, where 0 represents complete absence, and 1 signifies complete presence. By incorporating this function into our algorithm, we aimed to introduce a controlled amount of noise into the dataset without significantly altering its overall characteristics. This allowed us to achieve our primary goal of safeguarding privacy while maintaining the overall integrity of the dataset. By employing this approach, we successfully demonstrated our algorithm's feasibility and effectiveness in preserving sensitive data's privacy. The mathematical model of the theta (θ) parameters of the Iris dataset is presented below:

$$h_{\theta}(x) = g(\theta^T x) = g(z) = \frac{1}{1 + e^{-z}} \quad (3.1)$$

$$\theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

In Equation 3.1, $h_{\theta}(x)$ represents the hypothesis function, $g(z)$ is the sigmoid function, and z corresponds to the linear combination of features and weights. Here, θ denotes the parameter vector, and x represents the feature vector.

Algorithm 1 accomplishes this objective by generating a perturbed version of the original dataset, with a designated level of noise introduced. To achieve this, Algorithm 1 first identifies each feature's minimum and maximum boundaries in the original dataset and then adds a constant noise level, such as $\alpha = 0.2$, within these boundaries. The Iris dataset is utilized for our investigation, and Logistic Regression is employed to analyze how perturbation affects both the training and test datasets. Our objective was to explore the impact of perturbations on the training dataset's learning parameters and the effect of perturbations on the predictions derived from the test dataset.

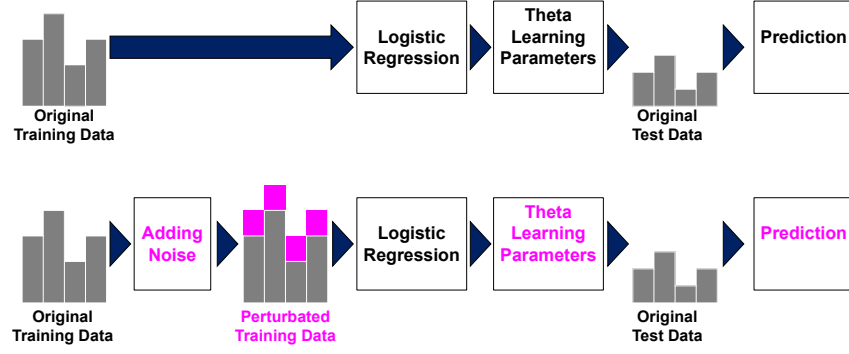
To evaluate the efficacy of Algorithm 1, we conducted a trial on the Iris dataset using the following steps:

Algorithm 1 The algorithm we used for perturbation

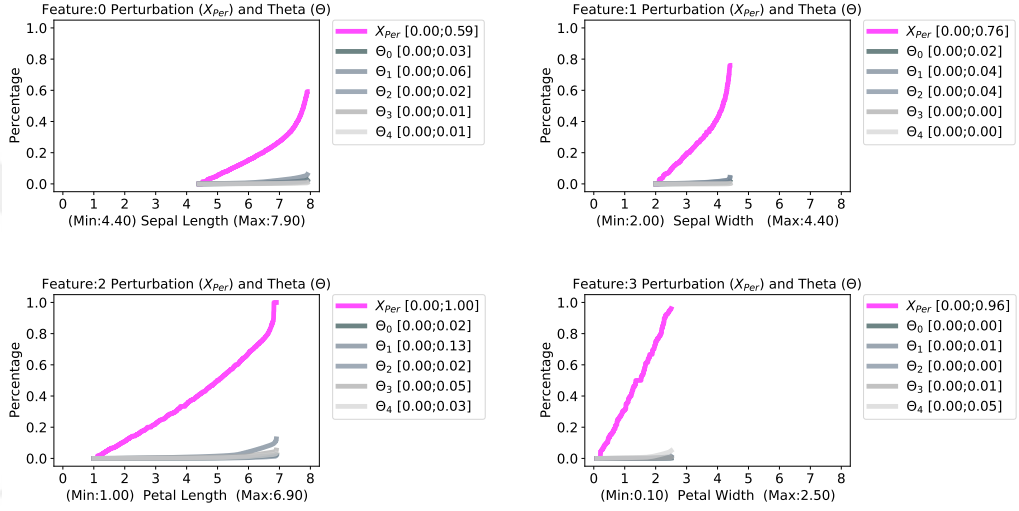
Require: Original dataset: $X_O \in \mathbb{R}^{m \times n}$.**Ensure:** Perturbed dataset: X_{Per} .

```
1: for  $j = 0$  to  $n$  do
2:    $feature_{min}(j), feature_{max}(j)$ 
3: end for
4: for  $i = 0$  to  $m$  do
5:   for  $j = 0$  to  $n$  do
6:      $noise = 0$ 
7:     for  $k = feature_{min}(j)$  to  $feature_{max}(j)$  do
8:        $noise += \alpha$ 
9:        $X_{Per}(i, j) += noise$ 
10:    end for
11:  end for
12: end for
13: return  $X_{Per}$ 
```

1. In the initial stages of our study, we aimed to investigate the effect of modifications in the learning parameters of ML algorithms. For this purpose, we applied Algorithm 1 to a copy of the training dataset, resulting in both an original dataset (X_O) and a perturbed dataset (X_{Per}). We then performed Logistic Regression on each dataset and recorded the theta θ -learning parameters. The original dataset was used to evaluate the predictive performance of the two training methods. To illustrate this experiment, we utilized the Iris dataset and applied the Logistic Regression classification method. Figure 3.2 depicts the experimental procedure. The top row of the figure shows the application of Logistic Regression to the original Iris dataset training set X_O , and the resulting theta θ -learning parameters are collected. The test dataset of Iris was then used to obtain predictions. The remaining row of the first part of Figure 3.2 illustrates our approach to introducing noise to the training set of the Iris dataset X_{Per} . Logistic Regression was applied after the addition of noise,



(a) Training Perturbation Algorithm

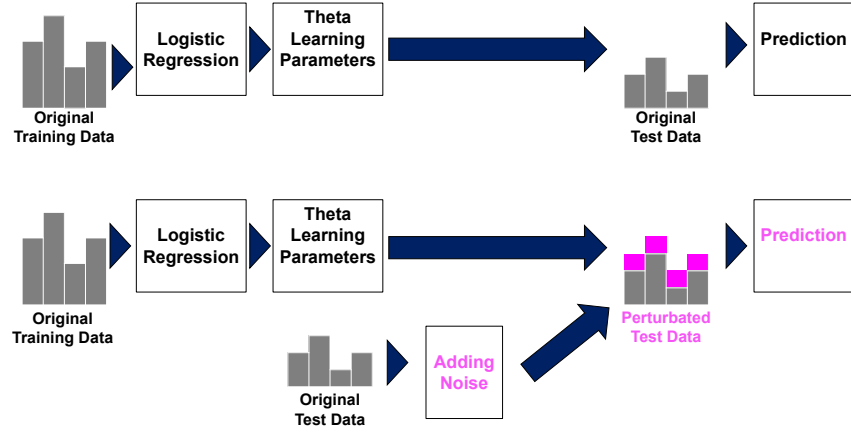


(b) θ for X_O and X_{Per} on the Iris Data.

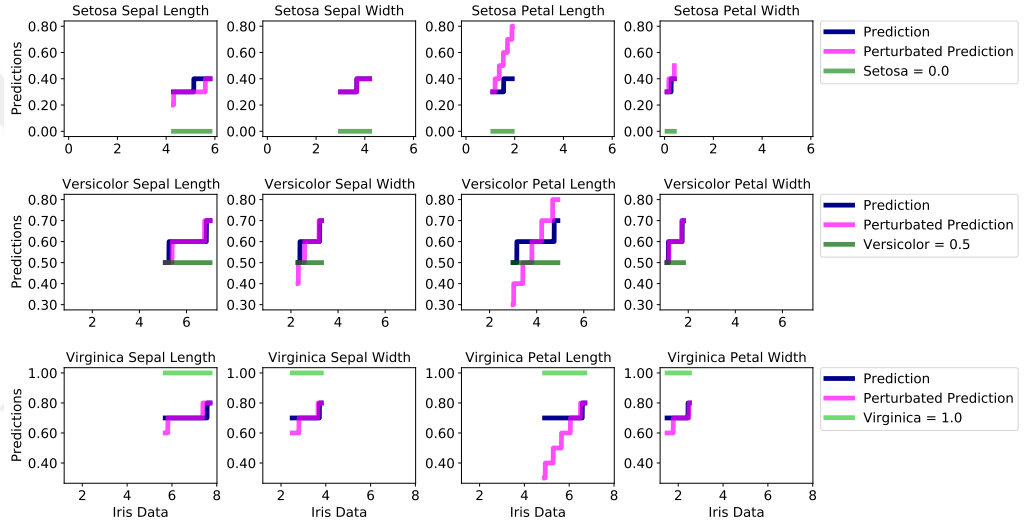
Figure 3.2 IRIS Dataset Training - Algorithm 1 Application

and the resulting theta θ -learning parameters were collected from the X_{Per} . The same test dataset used in the top row was also used to generate predictions, and the outcomes were compared in the second part of Figure 3.2. As shown in the figure, the percentage of perturbation indicates the deviation from the original Iris dataset, and as this percentage increases, the learning parameters display a reactive pattern. The results demonstrate that the addition of noise to the dataset can affect the learning parameters and, in turn, the predictive performance of the method. Therefore, our study shows that perturbation techniques can be used to improve the privacy of ML algorithms.

2. In the second phase of our study, we aimed to investigate changes in prediction accuracy as a result of adding noise to a test dataset. To accomplish this, we employed Algorithm 1 on a copy of the test dataset, generating both an



(a) Test Perturbation Algorithm



(b) Predictions

Figure 3.3 IRIS Dataset Test - Algorithm 1 Application

original and perturbed version of the dataset. The method was trained using the original training dataset X_O , and Logistic Regression was applied. Both the original and perturbed test datasets were used to assess predictive performance. The first part of Figure 3.3 illustrates this trial. Here, we used the original Iris Test dataset and the perturbed version of the dataset to evaluate Logistic Regression's predictive performance. The outcomes of the two predictions are compared, demonstrating the impact of noise addition on the method's accuracy. In the second part of Figure 3.3, the predictions from both datasets are displayed. This figure illustrates that adding noise affects the predictions due to the visual differences between the original predictions and the perturbed predictions. It is

important to note that the degree of perturbation in the test dataset is critical in determining the magnitude of the effect on the predictive performance. As the degree of perturbation increases, there is a greater impact on the method’s accuracy, as evidenced by the divergence between the predictions of the original and perturbed test datasets. Therefore, to maintain a balance between privacy preservation and predictive performance, it is essential to carefully select the degree of perturbation in the test dataset. Overall, our study demonstrates that Algorithm 1 can be a useful technique for maintaining data privacy while also preserving predictive performance. By introducing controlled noise to the datasets, we can limit third-party access to sensitive data while maintaining the predictive power of the method.

Upon analyzing the results of the trial experiment, we observed that adding noise affects both the training and prediction phases. As a result, we focused on developing a new approach that could protect private data and ensure privacy against ML methods.

3.2 Generative Adversarial Networks (GAN)

Our research led us to explore new approaches for protecting private data and ensuring privacy against ML methods. In this pursuit, we focused on two key techniques: DP and GAN. By leveraging GAN, we can create synthetic copies of confidential datasets, while DP allows us to preserve the privacy of these datasets. Our research identifies these two techniques as crucial targets for achieving our privacy preservation goals.

The first part of our approach involves the use of GAN. GAN consists of two NN - the G and D - that work against each other (Goodfellow et al. 2014). GAN is capable of generating synthetic datasets (X_G) that share similar statistical properties with a given test dataset (X_O). During training, GAN simultaneously trains G and D . G generates synthetic copies of the input values using random noise $z \sim p_z(z)$

and attempts to create fake outputs $x \sim p_{data}(x)$ that have a similar distribution to the given dataset, where p_z represents the probability distribution of random noise. Meanwhile, D learns how to distinguish real from synthetic data by calculating the probability output that comes from the given dataset rather than from G . GAN trains D to maximize the probability of assigning the correct label to training examples and samples from G . Additionally, GAN trains G to minimize $\log(1 - D(G(z)))$ simultaneously. The objective function V represents the two training procedures:

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (3.2)$$

where E represents the expected value.

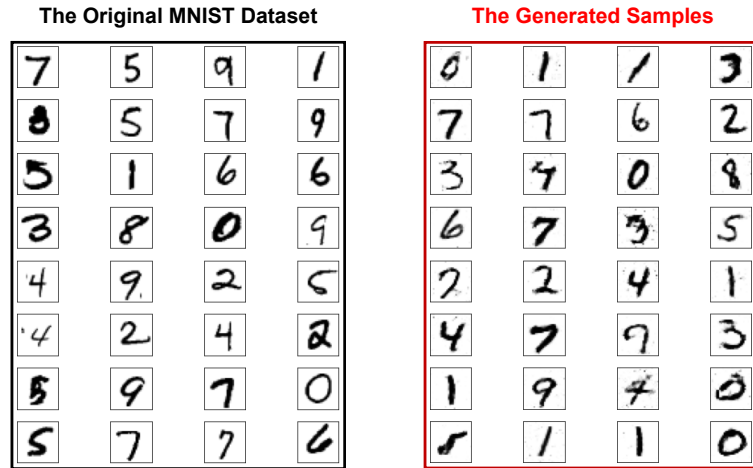


Figure 3.4 MNIST Experiment of GAN

The MNIST dataset, a widely recognized benchmark dataset in deep learning, comprises 70,000 images depicting handwritten digits (0 - 9), each measuring 28 x 28 pixels in size. The objective of the MNIST experiment utilizing GAN is to train a GAN model on the dataset, thereby generating new images that are visually comparable to those in the original dataset. In the MNIST experiment, the GAN model comprises two principal components: the G and the D . The G is responsible for producing fake images by taking random noise as input, while the D evaluates

the authenticity of the real and the fake images. During training, the G generates fake images presented to the D alongside real images from the MNIST dataset. The D is trained to accurately identify real images as real and fake images as fake. The G is then updated based on feedback from the D to produce better-quality fake images that are increasingly difficult for the D to distinguish. Training is continued until G produces fake images indistinguishable from the real images in the MNIST dataset. Once the model is trained, it can generate new images similar to those in the original dataset but which are not a part of it. In our study, we employed the vanilla-GAN algorithm and applied it to the MNIST dataset to generate new images that are visually similar to those in the original dataset. Our results, as shown in Figure 3.4, demonstrate the effectiveness of GAN techniques in creating synthetic data that is visually convincing. The generated samples in the figure clearly resemble the original MNIST dataset, demonstrating the ability of the GAN model to learn the underlying patterns and generate new samples that are consistent with the original dataset. Overall, our study provides evidence of the potential of GAN models in generating high-quality synthetic data for various applications in the field of ML.

In the GAN model, the D is responsible for classifying whether an input data point is real or fake. The D takes input from a real or generated data point. It then processes the input through a series of layers that perform linear and non-linear transformations to the input. The output of the D is a single value representing the probability of the input data point being real or fake. During training, the D learns to increase the probability of real data points and decrease the probability of fake data points.

The G , on the other hand, is responsible for generating new synthetic data points that are visually similar to the real data points from the dataset. It takes input from a vector of randomly generated numbers. The output of the G is a generated data point that is then fed to the D to evaluate whether it is real or fake. During training, the G learns to produce generated data points that can fool the D into classifying them as real data points.

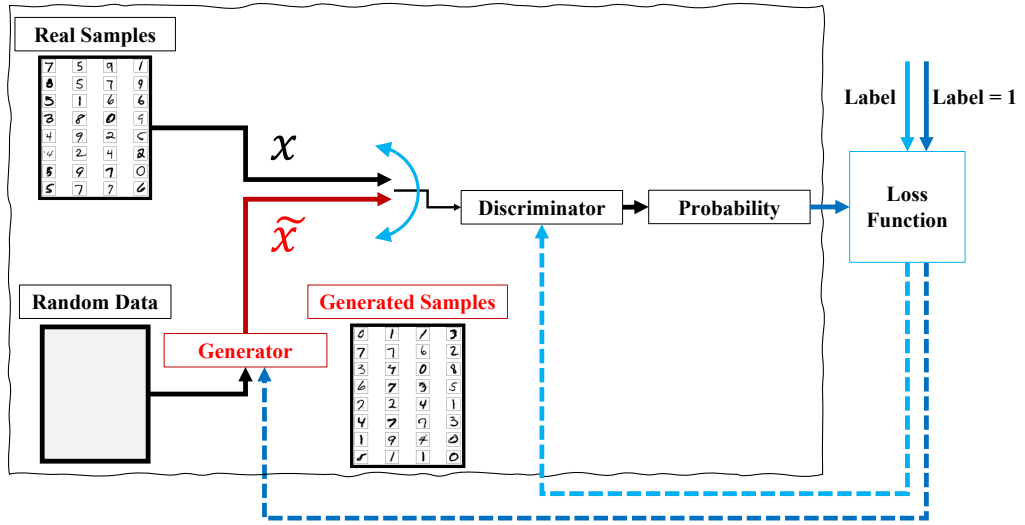


Figure 3.5 Generative Adversarial Networks

The training process for the D and G in a GAN model is iterative. The D is trained on a batch of real and fake data points during each iteration. The G is then updated based on the feedback from the D . This feedback is used to adjust the weights of the G 's layers to produce better-quality generated data points. The process continues until the D can no longer distinguish between real and fake data points, and the G can produce synthetic data points that are visually convincing (Figure 3.5).

The GAN algorithm introduced by Ian Goodfellow in his 2014 paper is called "Vanilla" because it is the simplest version of the GAN algorithm and has no additional features or modifications. The original paper by Goodfellow and associates (Goodfellow et al. 2014) demonstrated the effectiveness of GAN in generating synthetic images, but the same principles can also be applied to tabular data. The number of layers and neurons in each network can affect the performance of the GAN model in generating synthetic data. A more complex network with more layers and neurons can learn more complex features and patterns in the data, leading to better performance in generating synthetic data. Increasing the number of neurons in the D or G can impact the GAN's performance in different ways:

1. Increasing the number of neurons in the D : It can make the D more powerful

and better able to distinguish between real and fake data. However, if the D becomes too powerful, it may start to reject data that is actually real, leading to a decrease in the GAN's overall performance (Arjovsky, Chintala, and Bottou 2017).

2. Increasing the number of neurons in the G : This can allow the G to learn more complex patterns in the data and generate more realistic fake data. However, if the G becomes too complex, it may start to generate data that is too similar to the training data and lack diversity in its output (Brock, Donahue, and Simonyan 2018).
3. Increasing the number of neurons in both D and G : This can lead to a more powerful GAN that is better able to learn the underlying patterns in the data and generate more realistic synthetic data. However, increasing both at the same time can also increase the risk of overfitting the training data and decrease the GAN's generalization ability to new data (Radford, Metz, and Chintala 2015).

We used the Binary Cross-Entropy (BCE) loss function to measure the performance of the GAN models. The BCE loss measures the difference between the predicted output and the true label. It is a commonly used loss function for binary classification problems, where the output can take on values between 0 and 1. For a single sample, it is calculated as follows:

$$BCE(y_{pred}, y_{true}) = -y_{true} \cdot \log(y_{pred}) - (1 - y_{true}) \cdot \log(1 - y_{pred}) \quad (3.3)$$

where y_{pred} is the predicted output (a scalar between 0 and 1) and y_{true} is the true label (0 or 1).

We used the loss function to calculate both the D loss and the G loss. For the D , the true labels are 1 for real samples and 0 for generated samples. For the G , the true labels are 1 for all generated samples (since the goal of the G is to produce samples that fool the D into thinking they are real). So, in the D training phase, the BCE loss is calculated between the output of the D and the true labels for both the real and generated samples. The BCE loss is defined as ($D_{loss} = \text{loss function}(D_{output},$

"all samples labels")) where D_{output} is the output of the D for the combined real and generated samples, and "all samples labels" are the true labels for the combined real and generated samples. In the G training phase, the BCE loss is calculated between the output of the D for the generated samples and the true labels for the real samples. The BCE loss is defined as ($G_{loss} = \text{loss function}(D_{outputGenerated}, \text{"real samples labels"})$) where $D_{outputGenerated}$ is the output of the D for the generated samples, and "real samples labels" are the true labels for the real samples.

We used a test dataset to use for the calibration of the Vanilla-GAN model. We preferred the dataset of Pima Diabetes that could be publicly accessible¹. In the calibration trials, we aimed to find the optimum model of GAN by comparing their results. The numeric results and experiment architecture are explained in Chapter 4.2.

Although GAN models have traditionally been associated with image-generation tasks, recent research has revealed their potential for generating synthetic tabular data, opening up new possibilities for using GAN models as valuable tools in various applications. However, GAN models used for tabular data generation differ somewhat from those used for image generation tasks. One example of a GAN-based approach to creating synthetic tabular data is the CTGAN (L. Xu et al. 2019) method, which employs mode-specific normalization and processes each column independently to generate synthetic datasets that follow a tabular data distribution. This approach differs from traditional GAN that work with image data. In tabular data generation tasks, the GAN model's G takes random noise as input, similar to image generation tasks. However, the G produces a synthetic tabular data point instead of a synthetic image. The D is responsible for evaluating the authenticity of both real and fake tabular data points. The primary objective of training a GAN model in this context is to produce a G capable of generating synthetic tabular data points visually similar to the real ones. In the process, the D is trained to accurately distinguish between real and fake data points, resulting in a GAN model that can generate visually

1. <https://www.kaggle.com/uciml/pima-indians-diabetes-database>

convincing synthetic tabular data. We performed tests to compare Vanilla-GAN and CTGAN effects on Pima Diabetes dataset. The details can be seen in Chapter 4.6. We preferred to use CTGAN in further experiments.

3.3 Differential Privacy (DP)

DP was introduced by Dwork as a technique to preserve privacy while analyzing sensitive data using random noise (Dwork 2006). The DP technique perturbs the data by adding noise so that it does not significantly affect the overall probability distribution of the original data. Since the level and distribution of the noise are known, they can be compensated during statistical evaluation.

The (ϵ, δ) -DP framework is defined by Dwork, where ϵ represents DP, and δ is the probability of error, such as disclosing information. Here, δ defines the probability of DP failure. Consider a randomized algorithm $\mathcal{A}$, where $\mathbb{N}$ is the domain of all non-negative integers, including zero, and x and y are any two datasets such that $x, y \in \mathbb{N}$. The algorithm $\mathcal{A}: \mathbb{N} \rightarrow \mathcal{R}$ with range $\mathcal{R}$ satisfies (ϵ, δ) -DP for any subset of outputs $\mathcal{S} \in \mathcal{R}$ with the following definition,

$$Pr[\mathcal{A}(x) \in \mathcal{S}] \leq \exp(\epsilon) Pr[\mathcal{A}(y) \in \mathcal{S}] + \delta \quad (3.4)$$

where ϵ corresponds to the maximum l_1 distance between x and y , such as $\|x - y\|_1 = \sum_{x,y \in \mathbb{N}} |x_i - y_i| \leq 1$. This means that as ϵ decreases, the maximum allowable difference between the original and the perturbed data decreases, resulting in a higher level of privacy. However, as ϵ decreases, the amount of noise that needs to be added to the data to achieve this level of privacy also increases. Therefore, a trade-off between privacy and data utility must be balanced. Furthermore, the probability of privacy failure decreases as the value of δ decreases. In other words, the likelihood of sensitive information being disclosed is reduced. However, as δ decreases, the amount of noise that needs to be added to the data to achieve this level of privacy also increases, potentially compromising the accuracy and usefulness of the data. Again, finding the right balance between privacy and utility is crucial when using DP techniques.

In order to increase anonymity in DP, the Laplace distribution can be used to spread the data over a broader range (Dwork, Roth, et al. 2014). Typically, the Laplace distribution adds noise centered at 0 with the parameter $b = \frac{1}{\epsilon}$, such that $Lap(x | b) = \frac{1}{2b} \exp(-\frac{|x|}{b})$. The DP can be defined as a function $f: \mathbb{N} \rightarrow \mathcal{R}$ with the Laplace distribution, as follows,

$$\mathcal{A}_f(\mathbb{N}) \triangleq f(\mathbb{N}) + Lap(x | b). \quad (3.5)$$

Various privacy metrics were proposed to evaluate the performance of the proposed algorithm in preserving privacy. Dwork proposed the concept of ϵ -DP, which measures the level of privacy preservation in the DP technique. This metric ensures that any observer of the output of the DP algorithm cannot determine with high probability whether a particular individual's data is included in the dataset. DP is a powerful technique that can preserve privacy while analyzing sensitive data. It provides a way to add random noise to the data, reducing the risk of disclosing sensitive information. The technique defines the (ϵ, δ) -DP framework, ensuring that the data is protected from disclosure with a certain probability level. The Laplace distribution is a commonly used method for adding noise to the data in DP.

When (ϵ, δ) are decreased, the privacy level of the data is increased. This means the risk of disclosing sensitive information is reduced, and the data becomes more secure. However, decreasing (ϵ, δ) also increases the amount of noise added to the data. As a result, the accuracy of the data may be reduced, and the utility of the data may be compromised. On the other hand, when (ϵ, δ) is increased, the privacy level of the data is decreased, which means that the risk of disclosing sensitive information is higher. However, the accuracy of the data may be improved, and the utility of the data may be enhanced. It is important to balance privacy and utility and to choose appropriate values for (ϵ, δ) depending on the specific use case and requirements.

DP can fail when the added noise is too small, and an attacker can use statistical analysis to deduce private information from the data. DP can also fail when the added noise is too large, and the data becomes unusable or meaningless. Additionally,

DP can fail when data is combined with other datasets not subject to DP protection. It is important to carefully choose the value of ϵ and δ based on the specific use case and data sensitivity to balance privacy and utility.

The Laplace mechanism is a commonly used method for adding noise to data in DP because it has desirable mathematical properties that make it well-suited for privacy preservation. Specifically, the Laplace distribution is symmetric and has heavy tails, which means that it can add sufficient noise to a dataset to protect the privacy of individuals while preserving the statistical utility of the data. Moreover, the Laplace distribution has a well-defined and controllable sensitivity parameter, which allows us to adjust the amount of noise added to a dataset based on the level of privacy protection required. This sensitivity parameter is directly related to the privacy guarantee provided by DP. While other distributions could be used for adding noise in DP, the Laplace mechanism was extensively studied and was a well-understood mathematical foundation. As such, it is a popular choice for adding noise to DP applications. However, other distributions or noise models may be more appropriate depending on the specific application and privacy requirements. Other probability distributions can be used in DP besides the Laplace distribution. For example, the Gaussian distribution can also be used for this purpose. One advantage of using the Gaussian distribution is that it can produce more precise results than the Laplace distribution when the sensitivity of the query is high. Another distribution explored in the context of DP is the geometric distribution. This distribution is used in cases where the query output is a count of some event, such as the number of individuals in a database with a certain medical condition (Kasiviswanathan et al. 2011). Finally, the Cauchy distribution has also been proposed as an alternative to the Laplace distribution in DP. This distribution has a heavier tail than the Laplace distribution, which can produce more extreme values (Kairouz, Oh, and Viswanath 2016). Notably, the probability distribution choice depends on the specific use case and the desired level of privacy and accuracy. Each distribution has strengths and weaknesses and may be more appropriate in certain scenarios. Each of these distributions has its characteristics and advantages. The Gaussian distribution is widely used in statistical

analysis and ML. Compared to the Laplace distribution, the Gaussian distribution has a larger spread, meaning it can add more noise to the data while preserving privacy. However, it has the drawback of not being robust to outliers, meaning that the noise added to the data could be too large and result in inaccurate data. The Exponential distribution is a good option for cases where the data has a skewed distribution. This distribution can be used to add noise with a long tail that captures the skewness of the data. However, the Exponential distribution has the limitation of being sensitive to the scale of the data, meaning that it may add too much or too little noise depending on the scale of the data. Overall, each of these distributions has its trade-offs in terms of privacy preservation and accuracy of the data. The choice of which distribution to use depends on the data's specific characteristics and the application's privacy requirements. Dwork chose the Laplace distribution because it balances privacy and utility well and is computationally efficient to implement (Dwork 2008). However, using other distributions may be more appropriate depending on the specific privacy requirements of the application.

Algorithm 2 The algorithm we used for Differential Privacy

Require: Original dataset: $X_O \in \mathbb{R}^{m \times n}$.

Ensure: DP dataset: X_D .

```

1: for  $j = 1$  to  $n$  do
2:   for  $k = 1$  to  $m$  do
3:      $\theta = \text{randn}(0, 1)$ 
4:     if  $(\theta \geq 0.5)$  then
5:        $X_D(k, :) = X_O(k, :) + \text{Laplace}(\epsilon)$ 
6:     end if
7:   end for
8: end for
9: return  $X_D$ 

```

In this study, we tested DP, where the noise is added to all columns. The DP approach is shown in Algorithm 2. For ease of implementation, it employs the ϵ parameter in a manner that differs from its conventional use in the DP framework.

In standard DP, increasing ϵ adds less noise to the output, thereby providing a weaker privacy guarantee but potentially higher accuracy. Algorithm 2, on the other hand, operates in a manner that is distinct from standard DP. Here, as ϵ increases, more Laplace noise is added to each record, resulting in stronger privacy. This is because Algorithm 2's Laplace mechanism is designed to increase the magnitude of the added noise in proportion to the ϵ value. While the behavior of Algorithm 2 may appear unreasonable, it is nevertheless a valid implementation of the DP framework. According to Algorithm 2, increasing ϵ leads to more noise, resulting in stronger privacy guarantees.

Algorithm 2 is a DP mechanism that takes as input an original dataset X_O and returns a DP version of the dataset X_D . The $\text{Laplace}(\epsilon)$ in step 5 refers to adding Laplace noise with a scale of ϵ to the data element $X_O(k, :)$ in the k^{th} row of the X_D dataset. The $\text{randn}(0, 1)$ in step 3 refers to generating a random number from a Gaussian distribution with a mean of 0 and a standard deviation of 1, which is used to randomly decide whether to add noise to the k^{th} row of X_D . Specifically, if the random number is greater than or equal to 0.5, then the noise is added to the k^{th} row; otherwise, no noise is added. The purpose of adding randomness in this way is to ensure that the mechanism satisfies DP. The for loops in lines 1 – 8 are used to iterate over all data elements in the X_O dataset. Overall, this algorithm implements the randomized response technique, where noise is added to the data with some probability to mask the true values while ensuring DP. The algorithm is designed to provide DP. The addition of Laplace noise with a scale of ϵ to each data element in the X_D dataset is a standard method for achieving DP, and the use of the $\text{randn}(0, 1)$ random number G to decide whether to add noise provides a probabilistic way of achieving privacy while still preserving the utility of the data.

δ is not explicitly used in the Laplace mechanism for DP. The Laplace mechanism adds noise to the output of a query proportional to the sensitivity of the query and inversely proportional to the privacy parameter ϵ . The value of δ is used in other DP mechanisms, such as the Gaussian mechanism or the advanced composition

theorem, to control the probability of the mechanism violating privacy guarantees. The algorithm is also formulated in such a way that the privacy parameter ϵ can be tuned to achieve a desired level of privacy, allowing the user to balance privacy and utility based on their specific needs. The Laplace mechanism used in this algorithm is based on adding random noise drawn from the Laplace distribution with a scale parameter of sensitivity divided by ϵ . δ is a parameter that specifies the upper bound on the probability that the algorithm fails to satisfy DP. It is not explicitly used in this implementation of the Laplace mechanism. In other words, δ is not used in this algorithm because it uses only the ϵ parameter to control the amount of noise added to the data. The choice of ϵ determines the strength of privacy protection, with greater values of ϵ providing stronger privacy guarantees but more noisy data. We used the algorithm to implement DP and applied tests related to the DP. The test results are explained in Chapter 4.3.

Recognizing that simply implementing a DP mechanism is insufficient to guarantee privacy protection in all scenarios, it is crucial to influence such mechanisms' effectiveness. These factors include the sensitivity of the protected data, the required level of privacy, and the implementation details. Therefore, thoroughly considering these factors is crucial when designing and implementing any DP mechanism. Our proposed method incorporates GAN to address these challenges. In Chapter 4, we describe the results of our experiments that evaluated the outcomes of different combinations of GAN and DP mechanisms. As we develop a new method, measuring its effectiveness in both ML prediction performance and privacy preservation is essential. While ML methods can be assessed by comparing prediction and test results, we must apply some privacy preservation metrics.

3.4 Privacy Metrics

The proposed methodology involves generating perturbed sensitive data from the original dataset. As such, it is imperative to employ a set of reliable evaluation metrics to gauge the effectiveness of the proposed method and the degree to which it

deviates from the original data. To this end, we utilized six distinct metrics in our tests, each providing a unique perspective on the method's performance:

- (i) **Mean Square Error (P_{MSE})** an error-based measure commonly used to evaluate the accuracy of a method's predictions (Wagner and Eckhoff 2018).
- (ii) **Normalized Variance (P_{VAR})** is a similarity measure that assesses the degree of similarity between two datasets (Wagner and Eckhoff 2018).
- (iii) **Directed Hausdorff (P_{HAUS})**, is a metric space used to compare the dissimilarity between two datasets (Piramuthu 2005).
- (iv) **Kullback-Leibler Divergence (P_{KL})** is an information gain or loss metric that measures the divergence between two probability distributions (Wagner and Eckhoff 2018), (Galas et al. 2017).
- (v) **Procrustes (P_{PRO})** is a shape comparison metric that compares the probability distribution's shape between two datasets (Gower 1975).
- (vi) **Pearson's Correlation Coefficient (P_{PCC})** is an information gain or loss metric that measures the degree of linear correlation between two variables (Wagner and Eckhoff 2018).

If the generated perturbed data significantly deviates from the original data, the resulting P_{MSE} , P_{VAR} , P_{HAUS} , P_{KL} , P_{PRO} , and P_{PCC} metrics will indicate non-zero values. Conversely, if the metrics produce zero values, it can be inferred that the privacy protection measures could have been more effectively implemented. It should be noted that these metrics serve as quantitative measures to evaluate the degree of privacy protection afforded by the proposed approach. A non-zero value indicates that the perturbed data differs from the original data, and thus, the privacy protection measures are effective. Conversely, a zero value suggests no difference between the original and perturbed data, indicating that the proposed privacy protection measures may not be robust enough to protect sensitive data.

The six metrics we utilized - P_{MSE} , P_{VAR} , P_{HAUS} , P_{KL} , P_{PRO} , and P_{PCC} - were chosen for their distinct measurement approaches. Specifically, P_{MSE} and P_{HAUS}

use Euclidean distance to measure differences and metric distances between datasets, respectively. P_{VAR} , on the other hand, uses statistical variance to assess the dispersion between two datasets. P_{PCC} measures the linear dependency between datasets, while P_{KL} uses entropy to measure statistical distances. Lastly, P_{PRO} analyzes the shape of two datasets. To illustrate the significance of these metrics, consider the scenario where the same constant value is added to each item in a dataset. In this case, P_{MSE} , P_{KL} , and P_{HAUS} metrics will indicate a difference between the new and original datasets, while P_{VAR} , P_{PCC} , and P_{PRO} metrics may not detect any difference. Although this modification creates a distinct dataset, its original version can still be easily analyzed. However, when all six metrics behave similarly, we can have a high degree of confidence in the effectiveness of privacy protection measures. This is because each metric defined in this section measures and compares a unique property of the transformed dataset with the original dataset. By utilizing multiple metrics, we can evaluate the differences between datasets from various perspectives and, thus, develop a more comprehensive understanding of the level of privacy protection.

3.4.1 Privacy Metric - Mean Squared Error (P_{MSE})

P_{MSE} is a widely used statistical metric, which we utilize as a privacy metric to measure the Euclidean distance between the features of the original and perturbed datasets. The P_{MSE} metric effectively ensures better privacy when its value diverges from zero to higher values. Higher P_{MSE} values indicate a more significant divergence between the original and perturbed datasets, implying a higher degree of privacy protection. In contrast, lower P_{MSE} values suggest that the original and test datasets are more similar, which can compromise the privacy of sensitive data.

$$P_{MSE} = \frac{1}{m} \sum_{x \in X_O} \|x - y\|^2, \quad (3.6)$$

Here, X_O is the original dataset ($x \in X_O$), X_{Per} is the perturbed dataset ($y \in X_{Per}$), and m corresponds to the number of observations.

3.4.2 Privacy Metric - Normalized Variance (P_{VAR})

P_{VAR} , which measures the distribution difference between the original dataset X_O and the perturbed dataset X_{Per} (Oliveira and Zaiane 2010), is based on the statistical variance σ^2 . The metric reveals the degree of dispersion between the two datasets, as expressed by:

$$P_{VAR} = \frac{\sigma^2(X_O - X_{Per})}{\sigma^2(X_O)} \quad (3.7)$$

In our approach, P_{VAR} serves as a privacy metric that ensures stronger privacy protection when it has a higher variance. This is because higher P_{VAR} values indicate greater dispersion between the original and test datasets, indicating greater privacy preservation. Conversely, lower P_{VAR} values imply that the two datasets are more similar, which could compromise the privacy of sensitive data. Therefore, a higher P_{VAR} value is desirable to enhance the privacy protection of our approach.

3.4.3 Privacy Metric - Directed Hausdorff (P_{HAUS})

P_{HAUS} is a metric that measures the distances between two subsets in metric space and determines the similarity of the corresponding datasets based on their positions in metric space (Piramuthu 2005). When the two datasets are perfectly matched, P_{HAUS} equals zero. However, if P_{HAUS} is higher than zero, the two datasets can be considered independent and offer higher privacy. Notably, higher values of P_{HAUS} indicate better privacy protection. P_{HAUS} can be defined as the maximum set distance to the nearest point in the other set, as shown in:

$$P_{HAUS} = \max_{x \in X_O} (\min_{y \in X_{Per}} (d(x, y))). \quad (3.8)$$

In this formula, x and y refer to points in sets X_O and X_{Per} , respectively. The metric between these points, such as the Euclidean distance between x and y denoted by $d(x, y)$, can be any relevant metric.

3.4.4 Privacy Metric - Kullback-Leibler Divergence (P_{KL})

P_{KL} divergence, also known as relative entropy, is a metric used to measure statistical distances between datasets by comparing their probability distributions. It works by comparing two probability distributions, P and Q , which belong to the same probability space S .

$$P_{KL} = P_{KL}(X_O || X_{Per}) = \sum_{x,y \in S} P(x) \log_2 \left(\frac{P(x)}{Q(y)} \right) \quad (3.9)$$

Here, X_O and X_{Per} represent the original and perturbed datasets, respectively. The formula for P_{KL} divergence shows that the relative entropy from Q to P can be used as an indicator of how different the distribution of X_{Per} is from the distribution of X_O . Thus, for better privacy protection, the P_{KL} divergence metric needs to have a value that is different from zero.

3.4.5 Privacy Metric - Procrustes(P_{PRO})

P_{PRO} analysis is a statistical method used to analyze the distribution of a set of shapes. It measures the similarity between two datasets based on the distribution of their shapes. Let X_O and X_{Per} be $(p \times k)$ dimensional matrices, and assume that we transform X_O into X_{Per} using an orthogonal transformation matrix T that minimizes the sum of squares of the residual matrix (Gower 1975). This is the basic idea behind P_{PRO} analysis. However, this method can also be employed to measure privacy by examining the variance of the residual matrix. The calculation given in equation 3.10 corresponds to the trace value of this matrix. If the trace value is equal to zero, we can conclude that X_O and X_{Per} are identical after the transformation, and no privacy protection has been achieved. Therefore, higher values for this metric indicate better privacy preservation.

$$P_{PRO} = tr((X_OT - X_{Per})^T (X_OT - X_{Per})) \quad (3.10)$$

3.4.6 Privacy Metric - Pearson's Correlation Coefficient (P_{PCC})

The P_{PCC} , or Pearson Correlation Coefficient, is a privacy metric that measures the linear dependence between two variables or datasets. It calculates the correlation between the original dataset X_O and the perturbed dataset X_{Per} using the formula shown in (Wagner and Eckhoff 2018):

$$P_{PCC} = \left| \left| \frac{cov(X_O, X_{Per})}{\sigma(X_O)\sigma(X_{Per})} \right| - 1 \right| \quad (3.11)$$

The cov term represents the covariance between X_O and X_{Per} , and the σ terms represent their respective standard deviations. When there is a robust linear relationship between the two datasets, the P_{PCC} will approach either one or negative one, depending on the direction of the relationship. To standardize the output of the P_{PCC} , we take the absolute value of the formula, subtract one, and then retake the absolute value. This transformation ensures that the P_{PCC} ranges from zero to one, with zero indicating no privacy protection and one indicating perfect privacy protection.

3.5 The Proposed Method

In this study, we performed the DP approach by randomly adding noise obtained from Laplace distribution to only sensitive features. Although the sensitive features can be determined by the data owner, Algorithm-3 can be used to detect the most sensitive features.

Algorithm 3 performs Linear Regression on each column of the original dataset (X_O), treating that column as the target variable, and identifies the set of input features that yield the highest R^2 value. This set of input features represents the dataset's most sensitive features. The algorithm operates as follows: It initiates a loop that iterates through each column of X_O . For each column, it designates that column as the target variable (y_{col}) and employs all other columns as input features. It then initiates another loop that iterates through the remaining columns of X_O , excluding the target column (y_{col}). During each iteration of the inner loop, the algorithm

includes the current column in the list of input columns ($input_{cols}$) and performs Linear Regression on X and y using the resulting set of input columns. Subsequently, the algorithm calculates the R^2 value of the resulting Linear Regression model and stores the maximum R^2 value along with its corresponding set of input columns. Once the inner loop concludes, the algorithm returns the maximum R^2 value for the current target column (y_{col}). After the outer loop completes, the algorithm provides a list of the maximum R^2 values for each column of X_O , which represent the dataset's most sensitive features.

Following the identification of the most sensitive features in the dataset, the proposed method can be executed using Algorithm 4 to protect the sensitive data. The proposed method incorporates a DP approach, as depicted in lines 10 to 17 of Algorithm 4. Additionally, we also tested DP (X_D), where noise was added to all columns. The DP approach is presented in Algorithm 2. The subsequent subsection provides a brief introduction to the privacy metrics utilized by the proposed algorithm to evaluate its effectiveness in preserving privacy.

Let $X_O \in \mathbb{R}^{m \times n}$ represent any data, where m and n represent sample and feature size, respectively. The primary motivation of this study is to ensure the privacy protection of (X_O) without compromising the accuracy (i.e., R -Squared score in continuous datasets) of the ML applications. The primary implementation-related assumptions for the proposed method can be listed as follows:

- (a) In order to protect sensitive data, it is important for data owners to explicitly determine the sensitivity of the features in their dataset. This can depend on various factors such as business needs, market dynamics, and competitive advantages. For instance, a raw material that is accessible through a limited number of vendors can be a competitive advantage, and its remaining inventory can be considered sensitive information. Similarly, any critical production machine maintenance data, production routes, and data that can disclose recipes need to be protected from unauthorized access. Since companies have

Algorithm 3 The algorithm we used for detecting Predictable Features

Require: Original dataset: $X_O \in \mathbb{R}^{m \times n}$

Ensure: Result table: $\text{Table}_{cols\&preds}$.

```
1: for  $i = 1$  to  $n$  do
2:    $y_{col} \leftarrow X_O[:, i]$ 
3:    $\text{input}_{cols} \leftarrow \emptyset$ 
4:   for  $j = 1$  to  $n$  do
5:     if  $X_O[:, j] \neq X_O[:, y_{col}]$  then
6:        $\text{input}_{cols} \leftarrow \text{input}_{cols} \cup X_O[:, j]$ 
7:        $X \leftarrow X_O[:, \text{input}_{cols}]$ 
8:        $y \leftarrow X_O[:, y_{col}]$ 
9:        $\text{model} \leftarrow \text{Linear Regression}()$ 
10:       $\text{model.fit}(X, y)$ 
11:       $y_{pred} \leftarrow \text{model.predict}(X)$ 
12:       $R\text{-Squared} \leftarrow \text{ComputeR2}(y, y_{pred})$ 
13:    end if
14:  end for
15: end for
16:  $\text{Table}_{cols\&preds} \leftarrow (R - \text{Squared}), X_O[:, \text{input}_{cols}]$ 
```

prior knowledge of their competitive advantages, they can define which features are sensitive and take necessary measures to prevent their disclosure. In this regard, the Algorithm 3 can be a useful tool to detect the predictivity degree of features in the dataset. By using this algorithm, data owners can identify which features have high predictivity and thus might contain sensitive information. At the end of this step, the addresses of the private features need to be declared, ensuring that they are protected from unauthorized access.

- (b) The proposed method transforms the selected features into synthetic and randomly noise-added ones.
- (c) This approach only ensures privacy protection and is considered a pre-processing operation on data. Its implementation can be adapted to fit into any cost-

Algorithm 4 The algorithm we used for Privacy-Preserving Method

Require: Original dataset: $X_O \in \mathbb{R}^{m \times n}$,

The subset of selected private features: S_Φ ,

The number of sensitive features in the original dataset: n_s ,

Ensure: Hybrid method dataset: X_{HGD} ,

CTGAN dataset: X_G .

```

1:  $X_H \leftarrow X_O$ 
2:  $X_G \leftarrow \text{CTGAN}(X_O)$ 
3: for  $i = 1$  to  $n$  do
4:   for  $j = 1$  to  $n_s$  do
5:     if  $(i = j)$  then
6:        $X_{HG}(:, i) \leftarrow X_G(:, j)$ 
7:     end if
8:   end for
9: end for
10: for  $j = 1$  to  $n_s$  do
11:   for  $k = 1$  to  $m$  do
12:      $\theta = \text{randn}(0, 1)$ 
13:     if  $(\theta \geq 0.5)$  then
14:        $X_{HGD}(k, :) = X_{HG}(k, :) + \text{Laplace}(\epsilon)$ 
15:     end if
16:   end for
17: end for
18: return  $X_{HGD}$ 

```

effective computational server located in the business environment. For example, IoT data are considered high-rate; in most cases, it needs to be processed in real-time or nearly real-time. The proposed approach can be realized in a pipeline with a data collection procedure. In that way, while the algorithm produces the privacy-preserved copy of the current data, edge servers can continue collecting data from the production line's sensors.

- (d) The privacy-preserved data will be transmitted to a cloud environment or third parties to perform ML analysis.

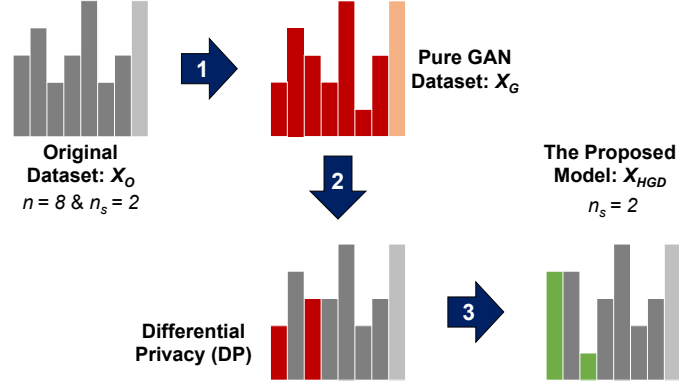


Figure 3.6 Illustration of the proposed approach with histograms.

(n, n_s : number of features, sensitive features, respectively.)

The implementation of the proposed approach is considered ($n_s=2$) in the illustration given in Figure 3.6. The original dataset (X_O) is gray. The first step is to create a synthetic copy (X_G) of (X_O) with the CTGAN function. (X_G) is shown in red color. Two sensitive columns ($n_s=2$) are in the example in Figure 3.6. In step 2, the sensitive columns in the (X_O) are replaced by (X_G). The DP function runs over the sensitive marked columns as the final step. These steps are listed in Algorithm 4. The first step in the proposed method is to create a synthetic copy (X_G) of the original data (X_O). We run the GAN operation for the whole dataset to protect the probability densities of the original dataset. Step-1 corresponds to line-2, Step-2 defines the lines between 3 and 9, and Step-3 represents the lines from 10 to 17 in Algorithm 4.

Our primary motivation for using GAN is the ability of GAN to learn the distribution of the original dataset. The GAN approximates the data distribution, whereas it creates different content. Although GAN can help protect data privacy with this property, our experiments show that the CTGAN (X_G) application may not preserve the accuracy. Therefore it is required to design a hybrid approach that takes only the valuable part of the GAN. We perform the GAN operation with the CTGAN function, a GAN model for tabular data distribution, and can learn better distributions than

Bayesian networks (L. Xu et al. 2019).

In the second step, the selected sensitive features from the original dataset are replaced with synthetic ones. Then, the new dataset (X_{HG}) is formed by taking the sensitive features from the synthetic dataset (X_G) and the remaining ones from the original dataset (X_O). At this step, the (X_{HG}) is the dataset without a random noise method. We investigated whether this data (X_{HG}) alone can be enough to accomplish our targets. We added it to our test list, where the experimental results section shows the results. Then DP algorithm is implemented to the marked sensitive features to add randomized perturbation to these features. The resulting dataset is shown by (X_{HGD}) in Algorithm 4. Thus, we not only reconstruct the sensitive columns synthetically but also add noise to them randomly with the help of the DP approach.

Another problem we want to emphasize is to prevent deducing sensitive results from a subset of (X_O). Our approach aims to create overall protection for privacy, and we also consider the hidden sensitive data. Different methods, such as statistical or ML, can be done by separating sensitive data from a subset of (X_O). For instance, it is possible to train a Linear Regression model with selected inputs from the subset of IoT data and get predictions for different purposes by adversarial Linear Regression operations. We performed tests for hidden sensitive data and assessed the proposed method's performance in the experimental results section. The results in the related section show that the proposed method ensures overall privacy protection of the IoT data. With the application of our proposed approach as a preprocessor, the IoT dataset or its subsets can be used only for the purpose it is prepared. It means they cannot be used for different adversarial AI operations to extract sensitive information regarding companies.

3.6 Privacy Preservation with Dimension Reduction

An alternative approach for privacy preservation is feature reduction, and PCA is a commonly used technique in this context. PCA transforms the original columns of a dataset into a new set of orthogonal variables called principal components.

While dimension reduction through PCA can lead to information loss in the original features, it also helps remove noise and preserve privacy. By representing the dataset in a lower-dimensional space defined by the principal components, the specific values of the original features become less directly accessible in the reduced dataset.

However, it is essential to note that PCA alone does not guarantee complete privacy protection or removal of sensitive information. Even though the individual values of the original features may not be directly visible in the reduced dataset, there are still potential privacy risks that need to be considered (Fan et al. 2021). Our analysis examined whether PCA could be utilized as an auxiliary tool for privacy preservation. We explored the concept of inverse PCA or PCA reconstruction, which involves approximating the original dataset by performing an inverse transformation using the retained principal components. The reconstructed dataset serves as an approximation of the original data but is not a replica.

During the application of PCA, a specific number of principal components are selected for dimensionality reduction. The PCA reconstruction process involves using the retained principal components in an inverse transformation. It includes multiplying the retained principal components by their corresponding eigenvectors and summing them up. It is important to emphasize that the goal of PCA reconstruction is not to recover the original dataset fully but rather to provide a reasonable approximation while reducing dimensionality. As a result, a reconstruction error is associated with the approximation, as it may only capture some of the intricate details and variations in the original data. The reconstruction error signifies the information loss incurred during the dimensionality reduction process. It should be noted that the reconstructed dataset may have different values than the original dataset, particularly

for reduced features. Instead, the reconstructed dataset aims to represent the patterns captured by the retained principal components as accurately as possible.

Mathematically, the dimensionality reduction using PCA can be represented as follows (Dash, Liu, and Yao 1997). Let X_O be the original dataset with $X_O \in \mathbb{R}^{m \times n}$, and our goal is to obtain a lower-dimensional PCA space denoted as $W \in \mathbb{R}^{m \times k}$, where $\mathbb{R}^n \rightarrow \mathbb{R}^k$. The mean of all samples is calculated as $\mu = \frac{1}{m} \sum_{i=1}^m x_i$, and the mean-centered data is denoted as $\tilde{X} \in \mathbb{R}^{m \times n}$, given by $\tilde{X} = x_1 - \mu, x_2 - \mu, \dots, x_m - \mu = \tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_m$. Here, we define the covariance matrix Σ as $\Sigma = \frac{1}{m-1} \tilde{X} \tilde{X}^T$. The eigenvectors V and eigenvalues λ are obtained by solving the equation $V \Sigma = \lambda V$, where the eigenvectors V are non-zero vectors and the eigenvalues λ are scalar values. The PCA space W is represented by $W = v_1, v_2, \dots, v_k$, where each eigenvector represents one principal component. The reduced dataset $X_{reduced} \in \mathbb{R}^{m \times k}$ is given by the equation:

$$X_{reduced} = W^T \cdot \tilde{X} = \sum_{i=1}^m W^T (x_i - \mu) \quad (3.12)$$

The original data X_O can be reconstructed, denoted as $\hat{X}_O \in \mathbb{R}^{m \times n}$, using the following equation:

$$\hat{X}_O = W \cdot Y + \mu = \sum_{i=1}^m W y_i + \mu \quad (3.13)$$

As a final note for the PCA calculation, we can calculate both the reconstructed error and robustness of the W as $X_O - \hat{X}_O$ and (total variance of W)/(total variance of X_O), respectively.

We conducted experimental trials using the reduced dataset denoted as $X_{reduced} \in \mathbb{R}^{m \times k}$, where we considered values of k to be 2, 3, and 4. Furthermore, to generate data sets augmented with DP and based on PCA reconstruction, we applied the DP Algorithm 2. These augmentations' outcomes can be observed in Chapter 4.8.

3.7 Cost Analysis

The time complexity of Algorithm 4 is $O(n_s(n + m))$. Since n_s is generally much smaller than n and m , the algorithm's complexity primarily depends on the maximum of n and m . However, it is important to note that the preprocessing step, involving the CTGAN implementation with its two deep NN (G and D) (Goodfellow et al. 2014), has a complexity of $O(m^3)$.

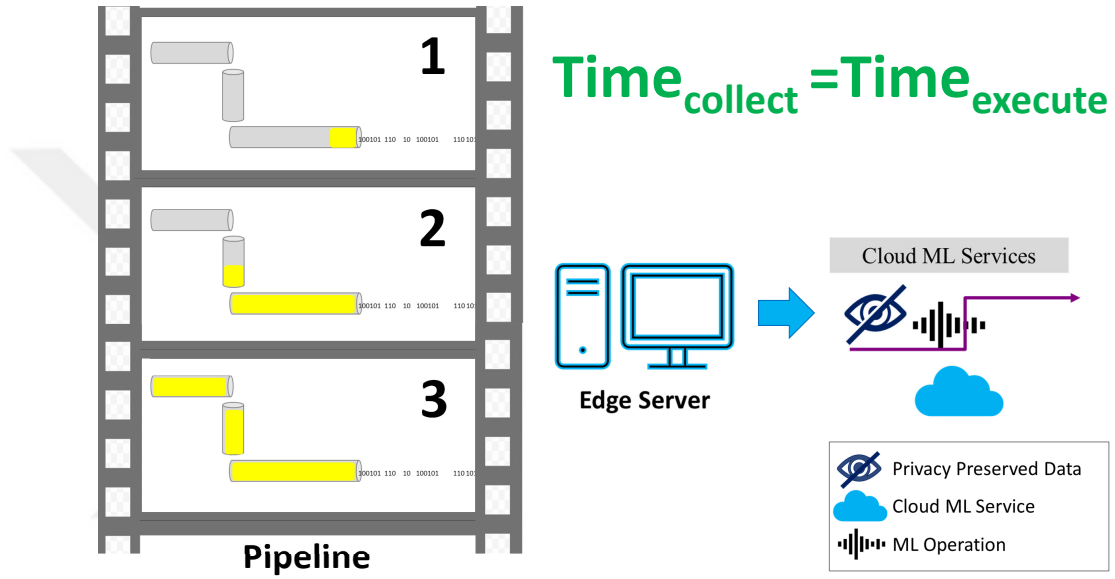


Figure 3.7 A Sample IoT Pipeline Illustration

In the proposed approach, the algorithm can be integrated into a pipeline structure during the process of ML data collection. It allows the CTGAN application to be performed with only a few data samples. As a result, the duration of data collection ($Time_{collect}$) and the execution time of the proposed method operation ($Time_{execute}$) can overlap. Figure 3.7 illustrates a sample data collection operation, which can be applied in an IoT or similar context. The figure visually depicts the steps of the proposed method in parallel with the data collection duration. Step one corresponds to the first two lines of Algorithm 4, step two encompasses the lines from 3 to 9, and the remaining part of the algorithm is represented by step three.

We monitored the usage of computer processors and memory over a specific period of 300 seconds. Chapter 4.10 shows the results of the experiments.

4. NUMERICAL EXPERIMENTS

4.1 Experimental Environment

Before delving into the details of the experiments, we would like to provide an overview of the experimental setup and environment. By providing this contextual information, we aim to comprehensively understand the experimental design and the tools utilized in our research.

All experiments were conducted on a notebook computer running the Windows 10 Operating System (OS), utilizing the Anaconda-Jupyter notebook environment. The computer system was equipped with an Intel Core i7 processor running at a clock speed of 2.7 GHz, 16 GB of RAM, and an integrated graphics card featuring Intel HD graphics 620, utilizing the system’s memory.

We employed two different dataset generation approaches to accomplish various testing objectives. Firstly, a vanilla GAN implementation was utilized. Secondly, we utilized the CTGAN function from the CTGANSynthesizer library, which specializes in synthesizing datasets. These approaches were employed to generate synthetic versions of the datasets under investigation.

We used the random and Laplace functions in the NumPy library to ensure the necessary DP calculations. These functions provided the means to incorporate the required privacy measures during the dataset generation process.

The experiments conducted in our study utilized a selection of widely accessible and publicly available datasets. These datasets included the Iris dataset, the Pima Diabetes dataset, the Wind Turbine dataset, the Energy Efficiency dataset, and the Synchronous Motors dataset. We also used a Steam Production dataset from a

confectionery manufacturing plant.

The Iris dataset holds significant prominence in ML and data analysis. It encompasses measurements of four distinct characteristics of three species of Iris flowers: Setosa, Versicolor, and Virginica. The dataset comprises 150 instances, each characterized by four attributes: (F_0 - Sepal Length), (F_1 -Sepal Width), (F_2 -Petal Length), and (F_3 -Petal Width). We used the Iris dataset in the experiments of privacy metric performances.

Similarly, the Pima Indians Diabetes dataset is a widely employed benchmark dataset for classification tasks in ML. Originally sourced from the National Institute of Diabetes and Digestive and Kidney Diseases, this dataset encompasses personal health information of female patients aged at least 21 years and of Pima Indian heritage. Its objective revolves around predicting the presence or absence of diabetes based on specific diagnostic measurements included within the dataset. The dataset comprises 768 observations, featuring eight input features (F_0 -Pregnancies), (F_1 -Glucose), (F_2 -Blood Pressure), (F_3 -Skin Thickness), (F_4 -Insulin), (F_5 -BMI), (F_6 -Diabetes Pedigree Function), and (F_7 -Age) and one target variable (0 or 1, indicating the presence or absence of diabetes). We used the Pima Indians Diabetes dataset in the experiments of GAN calibration, Vanilla-GAN and CTGAN comparison, DP's ϵ calibration, and privacy preservation method.

We used the remaining four datasets in the proposed method, hidden sensitive data case study, and dimension reduction experiments. Additionally, we used the Wind Turbine dataset to investigate metric privacy performances.

The Wind Turbine dataset (Soontronchai 2019) encompasses $m = 49,000$ observations and $n = 66$ columns collected between 2014 and 2015. The dataset includes measurements of a Wind Energy Converter (WEC) and various temperature measurements from inverter cabinets, front and rear bearings, rotors, and blades. Our experiments focused on the WEC as the dependent variable and considered 8,655 WEC-related observations until July 2014. The selected features for analysis in-

cluded (F_0 -WEC average wind speed), (F_1 -WEC average rotation), (F_2 -WEC average power), (F_3 -WEC average reactive power), (F_4 -WEC average available power from wind), (F_5 -WEC operating hours), and (F_6 -WEC production). We employed a Linear Regression model to predict the (F_6 -WEC production).

The Steam Production dataset originates from a food and confectionery manufacturing plant, encompassing 2,556 observations and 11 columns. The dataset features variables such as (F_0 -Natural Gas Amount), (F_1 -Natural Gas Pressure), (F_2 -Steam Pressure), (F_3 -Produced Hot Water Amount), (F_4 -Tank Amount of Raw Material), (F_5 -Steam Temperature), (F_6 -Filling Water Temperature), (F_7 -Flue Temperature), (F_8 -Inlet Temperature), (F_9 -Outlet Temperature), and (F_{10} -Steam Recovery). Our study regarded steam production as the dependent variable and established an experimental method using the Linear Regression approach based on the F_{10} prediction. The experimental results for steam production are presented in Table 4.6.

Additionally, the Energy Efficiency (Tsanas and Xifara 2012) and Synchronous Motors (Bayindir and Kahraman 2021) datasets were included in our analysis and are publicly available. The Energy Efficiency dataset focuses on energy analysis. It encompasses 768 observations and 9 features, including (F_0 -Relative Compactness), (F_1 -Surface Area), (F_2 -Wall Area), (F_3 -Roof Area), (F_4 -Overall Height), (F_5 -Orientation), (F_6 -Glazing Area), (F_7 -Glazing Area Distribution), and (F_8 -Cooling Load). The output variable in this dataset is the (F_9 -Heating Load). For our analysis, we employed a Linear Regression model to predict the F_9 , and the results obtained are reported in Table 4.7.

On the other hand, the Synchronous Motors dataset represents real-time machine data operating environments and consists of 557 rows and 5 columns. The columns in the Synchronous Motors dataset include (F_0 -Load Current), (F_1 -Power Factor), (F_2 -Power Factor Error), (F_3 -Changing of Excitation Current of a Synchronous Machine), and (F_4 -Excitation Current of a Synchronous Machine). Our analysis used a Linear Regression function to predict the excitation current F_4 . The results

of this analysis are also presented in Table 4.7.

We utilized the Sklearn library for ML modeling, employing algorithms such as Logistic Regression, Support Vector Machine (SVM), and Random Forest (RF) for datasets with binary or multi-class target variables while employing Linear Regression for datasets with continuous or numerical target variables. Visualizations in our experiments were created using the Matplotlib, Seaborn, and Plotly libraries.

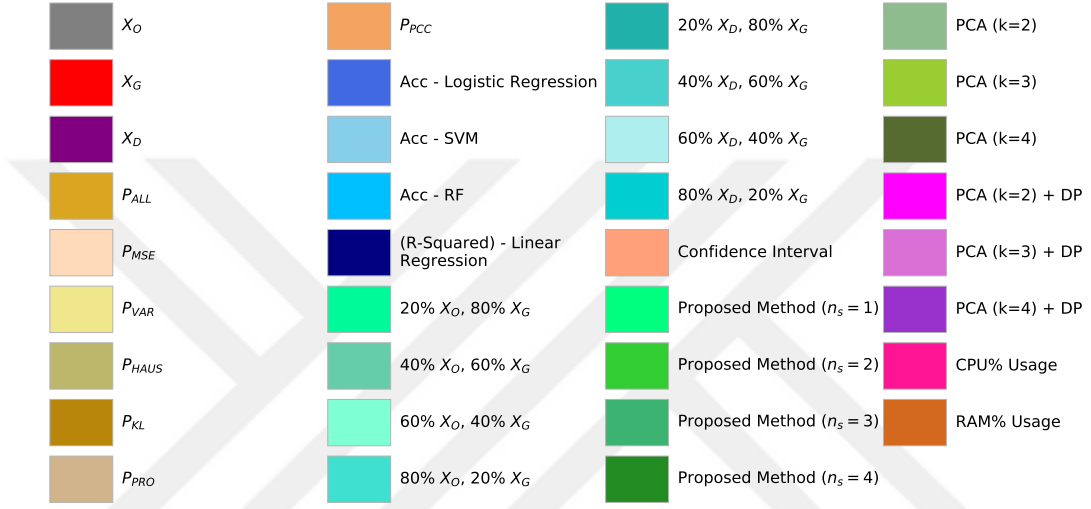


Figure 4.1 Reference Chart of Colors and Notations Used in Experiments

We performed a series of tests and visualized the results through plots in all experiments conducted in this thesis. We used fixed colors to identify and compare different items throughout the analysis quickly. Figure 4.1 depicts the color table.

4.2 Vanilla-GAN Calibration Experiments

In general, the ideal number of neurons and layers in the D and G depends on the complexity of the dataset and the specific GAN architecture used. It is important to find a balance between having enough capacity to capture the important features of the data while avoiding overfitting and poor generalization. Finding the right balance between complexity and generalization in the D and G networks is essential. The best-suited number of layers and neurons can also depend on the specific characteristics of the dataset and the complexity of the underlying data distribution.

We used the Pima Indians Diabetes dataset to test for the calibration of the Vanilla-GAN model. Experimentation and network architecture tuning are often necessary to achieve the best performance. We experimented and fine-tuned to determine the appropriate number of layers in the D and G for the Pima Indians dataset. We started with a simple architecture and gradually increased the complexity as needed. We also monitored the model’s performance during training and adjusted the architecture accordingly.

Starting the G with 100 neurons is common in GAN (Goodfellow et al. 2014). The G takes a random noise vector as input and generates synthetic data. By starting with a noise vector of 100 dimensions, we provide enough capacity for the G to learn complex patterns in the data. However, the number of dimensions for the noise vector can be adjusted based on the complexity of the dataset and the task at hand. In practice, the number of neurons in the G can also be adjusted based on the complexity of the data and the desired quality of the generated samples. In our tests, we started the G with 100 neurons and set the output neurons to 8 due to the number of features of the Pima Indians Diabetes dataset.

Table 4.1 GAN Neural Network Configuration Table

Discriminator and Generator			
Case	Hidden Layers	Neurons per Layer	Adversarial Loss
1	0	8	3.2401
2	1	16	1.7308
3	2	32, 16	0.7806
4	3	64, 32, 16	3.0936
5	4	128, 64, 32, 16	1.3819
6	5	256, 128, 64, 32, 16	0.6934
7	6	512, 256, 128, 64, 32, 16	0.7151
8	7	1024, 512, 256, 128, 64, 32, 16	67.5000
9	8	2048, 1024, 512, 256, 128, 64, 32, 16	95.0183

We used the BCE loss function to measure the performance of the GAN models. Table 4.1 provides a comprehensive overview of the architecture used for various cases in Vanilla GAN tests on the Pima Indian dataset. The table sheds light on the impact of different GAN model architectures on their performance and emphasizes the significance of selecting an appropriate architecture for a given task. The table presents different cases with varying numbers of hidden layers and neurons per layer,

along with their corresponding adversarial loss. Adversarial loss is a measure of the discrepancy between the distribution of real images in the training set and the distribution of generated images by the G . Lower values of adversarial loss indicate that the GAN model performs better in generating realistic images. The table shows that as the number of hidden layers and neurons per layer increases, the adversarial loss initially decreases, indicating the improved performance of the GAN model in generating realistic images. However, beyond a certain point, increasing the number of layers and neurons leads to overfitting, resulting in a higher adversarial loss. Thus, the table provides valuable insights into the trade-off between the number of layers and neurons in a GAN model and its ability to generate realistic images. It also underscores the significance of refining the architecture of a GAN model for a specific dataset to attain peak performance.

Notably, the best performance in terms of adversarial loss is achieved in case 6, where the D and G networks have five hidden layers with 256, 128, 64, 32, and 16 neurons per layer, respectively. This suggests that a balance needs to be achieved between the network's complexity and the amount of training data available to achieve the best possible performance. Furthermore, the D and G architecture used in Case 6 was able to effectively generate synthetic data that closely resembles the real data, indicating that this architecture is suitable for generating realistic synthetic data. Overall, Table 4.1 provides valuable insights into the architecture of GAN models and their impact on their performance in generating realistic synthetic data. It highlights the importance of refining the architecture of a GAN model for a given dataset and task to achieve the best possible performance.

The visualization in Figure 4.2 corresponds to the results presented in Table 4.1. The X-axis of each plot displays epoch numbers ranging from 0 to 100, and the Y-axis shows the loss values for each case from case 1 to case 9. The final loss values for each case are displayed at the top of each plot. In the first plot, corresponding to case 1, the loss function increases as the number of epochs increases, resulting in a final loss value of 3.2401. The second plot, corresponding to case 2, shows two peak

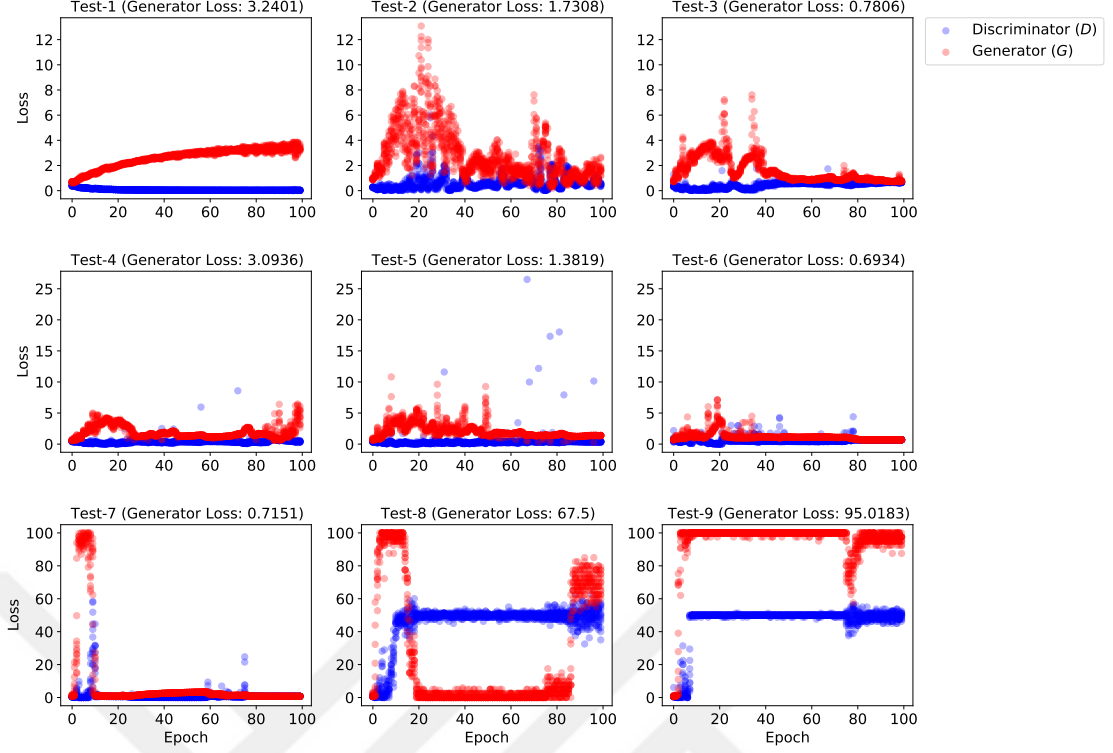


Figure 4.2 Calibration for Minimum Loss (Vanilla - GAN)

values at around 20 and 80 epochs, and it achieves a final loss value of 1.7308. Case 3 demonstrates a better performance than the previous two cases, and as expected, its loss value decreases as the number of epochs increases. The plot for case 3 displays a final loss value of 0.7806. Similarly, the remaining plots show improvements in performance, except for case 8 and case 9, which have a maximum loss value of 67.5000 and 95.0183, respectively. The plot for case 6 displays the best loss value of 0.6934, demonstrating that the difference between the real and generated images minimizes with increasing epoch numbers. The results underscore the importance of refining the architecture of a GAN model for a given dataset to achieve the best possible performance.

Our aim in the Vanilla-GAN Calibration Experiments was to identify the most appropriate configuration for the Vanilla-GAN model. We tested nine different configurations and calculated their corresponding adversarial losses. Ultimately, we identified the sixth configuration as having the lowest loss, and we used it in the subsequent experiment to assess its effectiveness in preserving privacy.

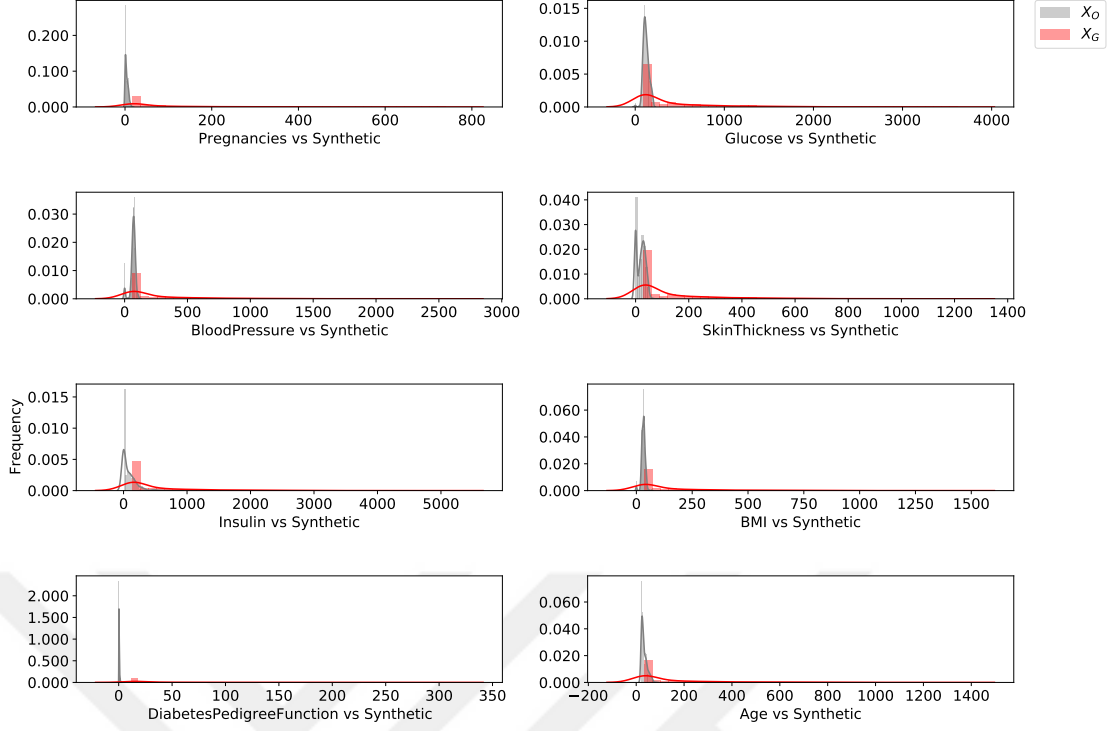


Figure 4.3 PDF Comparison of Synthetic vs. Original

After achieving one of the best loss values for the Vanilla-GAN model, we utilized Case 6 to generate a synthetic copy of the Pima Indians Diabetes dataset. Figure 4.3 compares the original dataset and its synthetic counterpart, which was generated using the Vanilla GAN Case 6 configuration. The figure displays each dataset feature in a separate plot, demonstrating both datasets' histogram and estimated probability density function. Upon comparing the two distributions, we observed shifts in the Vanilla-GAN model's (F_1 - glucose), (F_2 - blood pressure), (F_6 - diabetes pedigree function), and (F_8 - age). These shifts, or differences in the feature distributions' centers, indicate that the synthetic data may not accurately capture the original data's true distribution, leading to differences in mean or median values between the original and synthetic datasets. Moreover, we found discrepancies in the shapes of the two datasets' distributions in the features (F_2 - blood pressure), (F_3 - skin thickness), (F_5 - BMI), (F_6 - diabetes pedigree function), and (F_8 - age). The synthetic dataset exhibits a higher kurtosis shape than the original dataset, indicating a more peaked or clustered distribution, resulting in the synthetic data being more concentrated around a particular range of values. To address the effectiveness of the Vanilla GAN model, we compared it with CTGAN, a more sophisticated version of tabular GAN,

in Chapter 4.6. CTGAN is specifically designed for generating synthetic tabular data, while Vanilla GAN is a more general-purpose GAN that can be used for various types of data, including images, audio, and text. The synthetic dataset's performance in preserving privacy and maintaining high prediction rates needs to be evaluated through various metrics. In the following chapter, we present DP models and privacy metrics, which enable us to assess the performance of the proposed methods. We discuss the metrics and compare the models' performances over ten execution tests.

4.3 DP Experiments

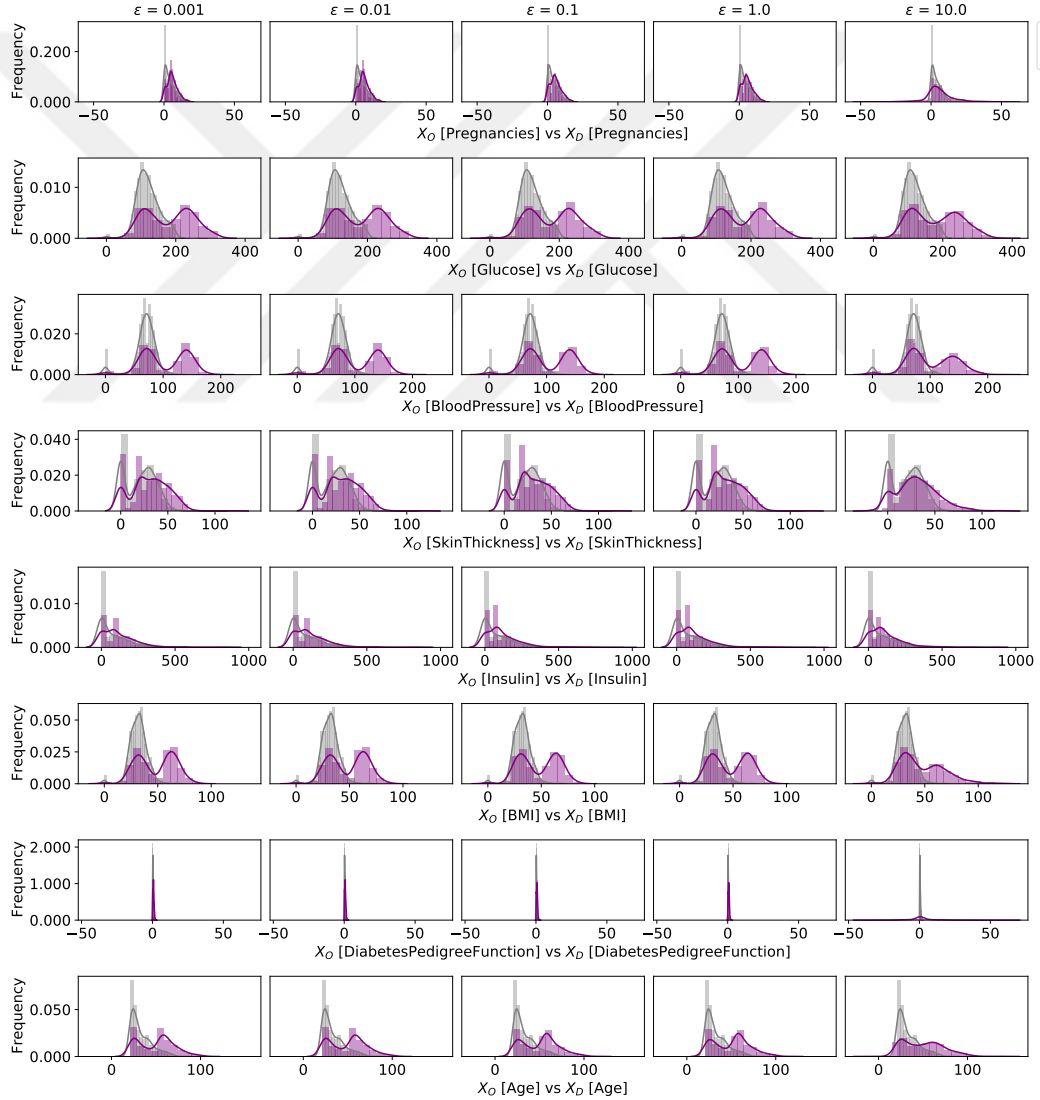


Figure 4.4 PDF Comparison: Original vs. DP (ϵ variations)

We have developed Algorithm 2 to generate a DP dataset, and in our DP experiments,

we used the Pima Indians Diabetes dataset. Figure 4.4 compares the original dataset with its DP counterpart for each feature, for different values of ϵ , where ϵ values vary between 0.001 and 10. The ϵ value affects the perturbation level, and an increase in ϵ corresponds to adding more noise. The impact of perturbation can be observed in the range of ϵ values between 0.001 and 10, where the shape and distribution centers of the two plots align when ϵ is 0.001. As the ϵ value increases, we observe a slight skewing of the plot towards higher values than the original dataset. Therefore, we have opted for the ϵ value of 0.01 for our further experiments. It is important to note that choosing an appropriate value of ϵ is crucial in DP data generation, as a low value of ϵ might not provide sufficient privacy protection, whereas a very high value of ϵ might significantly alter the statistical properties of the original dataset, rendering the DP data less useful for downstream analysis. Our choice of ϵ value strikes a balance between ensuring adequate privacy protection and maintaining the usefulness of the DP dataset. In our subsequent experiments, we assess the performance of our DP dataset and compare it with the original dataset using various metrics.

We utilized specific metrics to make meaningful comparisons between the original and perturbed datasets. Notably, these metrics help evaluate the impact of DP algorithms in real-world settings and understand their effectiveness in protecting sensitive data while maintaining analysis accuracy.

4.4 Performance Analysis of Privacy Metrics

We considered six distinct metrics to evaluate the level of privacy preservation achieved between the original dataset and its various modified versions. Multiple metrics are necessary due to their divergent behaviors and characteristics. By conducting different test scenarios, we systematically examined the response of each privacy metric, highlighting the importance of utilizing all six metrics concurrently to gain a comprehensive understanding of their performance in evaluating privacy measures. To conduct our analysis, we employed two distinct datasets: the Iris

dataset and the Wind Turbine dataset.

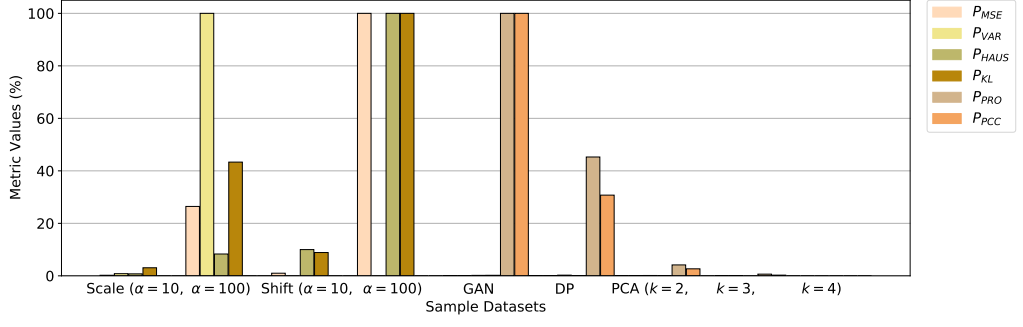
One of the experimental scenarios we investigated involved the introduction of noise through scaling and shifting operations. To control the magnitude of the noise, we employed a scalar parameter denoted as α . Firstly, we applied the scaling effect by multiplying each element of the tested datasets by $\alpha = 10$ and $\alpha = 100$, respectively. These modified datasets are visually represented in Figure 4.5 with the corresponding labels "x10" and "x100". Subsequently, we explored the shifting effect by adding $\alpha = 10$ and $\alpha = 100$ to each element of both datasets. The resulting datasets, reflecting the impact of the shifting operation, are illustrated in Figure 4.5 with labels "+10" and "+100".

In addition to the previously mentioned scenarios, we also explored using a synthetic copy generated by the GAN algorithm and applying DP techniques. To generate a synthetic copy of the Iris and Wind Turbine datasets, we employed the CTGAN algorithm, specifically designed to create synthetic data that closely mimics the characteristics of the original dataset. This allowed us to evaluate the effectiveness of the privacy preservation methods on the synthetic copy.

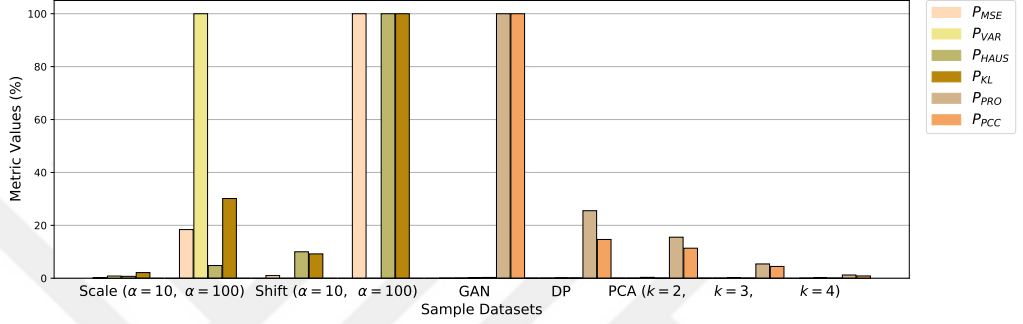
Furthermore, we incorporated the Laplace mechanism's DP function with a privacy budget of 0.01 to create a differentially private version of the tested datasets. By leveraging DP techniques, we aimed to enhance the privacy protection of the datasets while ensuring a controlled level of information disclosure.

Additionally, we conducted experiments involving PCA as a final test. Details of the PCA experiments are explained in subsequent chapters. We included PCA scenarios in the privacy metric experiments. In these scenarios, we reduced the dimensions of the tested datasets to 2, 3, and 4 using PCA. Subsequently, we performed an inverse PCA transformation to reconstruct the datasets to their original dimensions, thereby providing a reasonable approximation of the original datasets.

We conducted a series of operations to generate nine differentiated versions of the



(a) Iris Dataset



(b) Wind Turbine Dataset

Figure 4.5 Performance Analysis of Privacy Metrics

Iris and Wind Turbine datasets. The visible outcomes of these experiments are depicted in Figure 4.5. By analyzing the results presented in Table 4.2, which encompasses the metrics for both datasets, we can derive the following observations and interpretations regarding the behaviors of the items.

When we perform a scaling operation by $\alpha = 10$ and $\alpha = 100$, we can see that metrics of P_{KL} , P_{MSE} , P_{VAR} , and P_{HAUS} can capture differences while other metrics remain silent. The P_{VAR} metric shows the maximum difference for the $\alpha = 100$ operation in the two tested datasets. The P_{KL} metric shows a difference of around 40 % for the same operation. Similarly, the shifting operation by $\alpha = 10$ and $\alpha = 100$ is captured by the metrics of P_{HAUS} , P_{KL} , and P_{MSE} , where they reach 100 % in the second shift operation in both datasets.

The P_{VAR} values capture the variability or spread of the data. In both datasets, the P_{VAR} remains relatively low across all tests except the scaled "x100" version, indicating that the transformed versions maintain a similar level of variability

compared to the original data. However, the P_{VAR} values increase for certain transformations in the scaled "x100" version relative to the original data, suggesting a greater spread in the transformed data. The P_{HAUS} values quantify the dissimilarity between two sets of points. In both datasets, the P_{HAUS} values increase for the shifted "+100" version, indicating a more significant dissimilarity than the original data. It suggests that the transformed versions yield distributions farther from the original data. The P_{KL} values measure the difference between two probability distributions. Similarly, the P_{KL} values generally increase for scaled and shifted transformations in both datasets, signifying a more significant divergence from the original data distribution.

The P_{PRO} values assess the similarity between two shapes or datasets. The P_{PCC} values indicate the linear correlation between two variables. In both datasets, the P_{PRO} and P_{PCC} values show an increase for GAN and DP transformations. Additionally, PCA dimensions for 2 and 3 versions in the Wind Turbine dataset differ from the original dataset.

Table 4.2 Privacy Metrics Performance Analysis Table

Iris Dataset Metric Values %							Wind Turbine Dataset Metric Values %						
Tests	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}	Tests	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}
Original (X_O)	0.00	0.00	0.00	0.00	0.00	0.00	Original (X_O)	0.00	0.00	0.00	0.00	0.00	0.00
scale ($\alpha = 10$)	0.00	0.01	0.01	0.03	0.00	0.00	scale ($\alpha = 10$)	0.00	0.01	0.01	0.02	0.00	0.00
scale ($\alpha = 100$)	0.26	1.00	0.08	0.43	0.00	0.00	scale ($\alpha = 100$)	0.18	1.00	0.05	0.30	0.00	0.00
shift ($\alpha = 10$)	0.01	0.00	0.10	0.09	0.00	0.00	shift ($\alpha = 10$)	0.01	0.00	0.10	0.09	0.00	0.00
shift ($\alpha = 100$)	1.00	0.00	1.00	1.00	0.00	0.00	shift ($\alpha = 100$)	1.00	0.00	1.00	1.00	0.00	0.00
GAN	0.00	0.00	0.00	0.00	1.00	1.00	GAN	0.00	0.00	0.00	0.00	1.00	1.00
DP	0.00	0.00	0.00	0.00	0.41	0.26	DP	0.00	0.00	0.00	0.00	0.25	0.15
PCA ($k = 2$)	0.00	0.00	0.00	0.00	0.04	0.03	PCA ($k = 2$)	0.00	0.00	0.00	0.00	0.16	0.11
PCA ($k = 3$)	0.00	0.00	0.00	0.00	0.01	0.00	PCA ($k = 3$)	0.00	0.00	0.00	0.00	0.05	0.04
PCA ($k = 4$)	0.00	0.00	0.00	0.00	0.00	0.00	PCA ($k = 4$)	0.00	0.00	0.00	0.00	0.01	0.01

Our experimental findings reveal important insights into the performance of privacy metrics when differentiating datasets. Specifically, when we applied the scaling and shifting operations with scalar values of 10 and 100 to differentiate a dataset, the metrics P_{MSE} , P_{VAR} , P_{HAUS} , and P_{KL} effectively captured the differences between the original dataset and its transformed versions, enabling us to discern certain distinctions. However, it should be noted that the differences arising solely from the shift or scale effect could be easily analyzed, potentially leading to the disclosure of the

original data. Interestingly, the metrics P_{PRO} and P_{PCC} exhibited zero difference for all scale and shift operations, implying that these metrics could not detect variations caused by these specific transformations. However, they effectively captured the differences in the introduced GAN, DP, and PCA operations. This highlights the importance of considering multiple metrics simultaneously to assess privacy metrics' performance accurately.

Table 4.2 provides a comprehensive overview of the performance of the six privacy metrics on both the Iris and Wind Turbine datasets. It is evident from the table that the addition of a constant scalar (α) through the shifting operation had discernible effects on the P_{MSE} , P_{HAUS} , and P_{KL} metrics. In contrast, the P_{VAR} , P_{PRO} , and P_{PCC} metrics indicated no significant differences from the original dataset in these scenarios.

In conclusion, our experiments underscore the need to employ all six metrics to analyze privacy metrics' performance accurately.

4.5 Experimental Exploration of Privacy Preservation Methods

In this thesis, our primary objective is to enhance both privacy protection and accuracy values. We constructed methods employing GAN and DP techniques to achieve this goal to attain optimal results. The Pima Diabetes dataset was selected for our study. This section presents the experiments conducted on the Experimental Exploration of Privacy Preservation Methods (PP Methods) to assess the performance of methods developed using DP and GAN. For this experiment, we employed various classification algorithms, namely Logistic Regression, SVM, and RF ML Models, on the Pima Diabetes dataset. The dataset was divided into training sets (X -train and y -train) and testing sets (X -test and y -test), with dimensions of (560, 8), (140, 8), (560, 1), and (140, 1), respectively.

The PP Methods experiments encompassed the utilization of the following steps:

- (i) Original: We applied three classification algorithms to the original X -train dataset without any modifications. The accuracy values were the best results, but no privacy preservation was considered. The original dataset was used as the starting point, and other datasets were compared with it to measure the privacy preservation of the dataset. When the original dataset was reached, the six privacy metrics (P_{MSE} , P_{VAR} , P_{HAUS} , P_{KL} , P_{PRO} , and P_{PCC}) showed zero values, which proved it was the starting point for the experiment.
- (ii) GAN: Vanilla-GAN (case 6 in Table 4.1) was used to create a wholly synthetic copy of the X -train dataset. The three classification algorithms were applied to the dataset, and accuracy values were analyzed. The six privacy metrics were used to measure the differences between the GAN and the original dataset.
- (iii) DP: DP was applied to the original dataset with $\epsilon = 0.01$ in Algorithm 2, creating a DP dataset. Three classification algorithms were applied to the dataset, and accuracy values were analyzed. The six privacy metrics were used to measure the differences between the DP and the original dataset.
- (iv) 20% Original - GAN, 40% Original - GAN, 60% Original - GAN, and 80% Original - GAN: These were different versions of the mix of observations, with varying percentages of original and GAN observations.
- (v) 20% DP - GAN, 40% DP - GAN, 60% DP - GAN, and 80% DP - GAN: These were different versions of the mix of observations, with varying percentages of DP and GAN observations.

We initially created datasets in this section's experimental scenarios and applied three classification algorithms. We employed six privacy metrics to assess the differences between the generated and the original datasets. The experiments were conducted over ten repetitions.

Figure 4.6 presents the results of these executions on eleven datasets through nine plots. The first three plots display the accuracy deviations of Logistic Regression, SVM, and RF classifiers on the eleven datasets: Original, Vanilla-GAN, DP, and four combinations of Original + GAN datasets (at ratios of 20%, 40%, 60 %, and

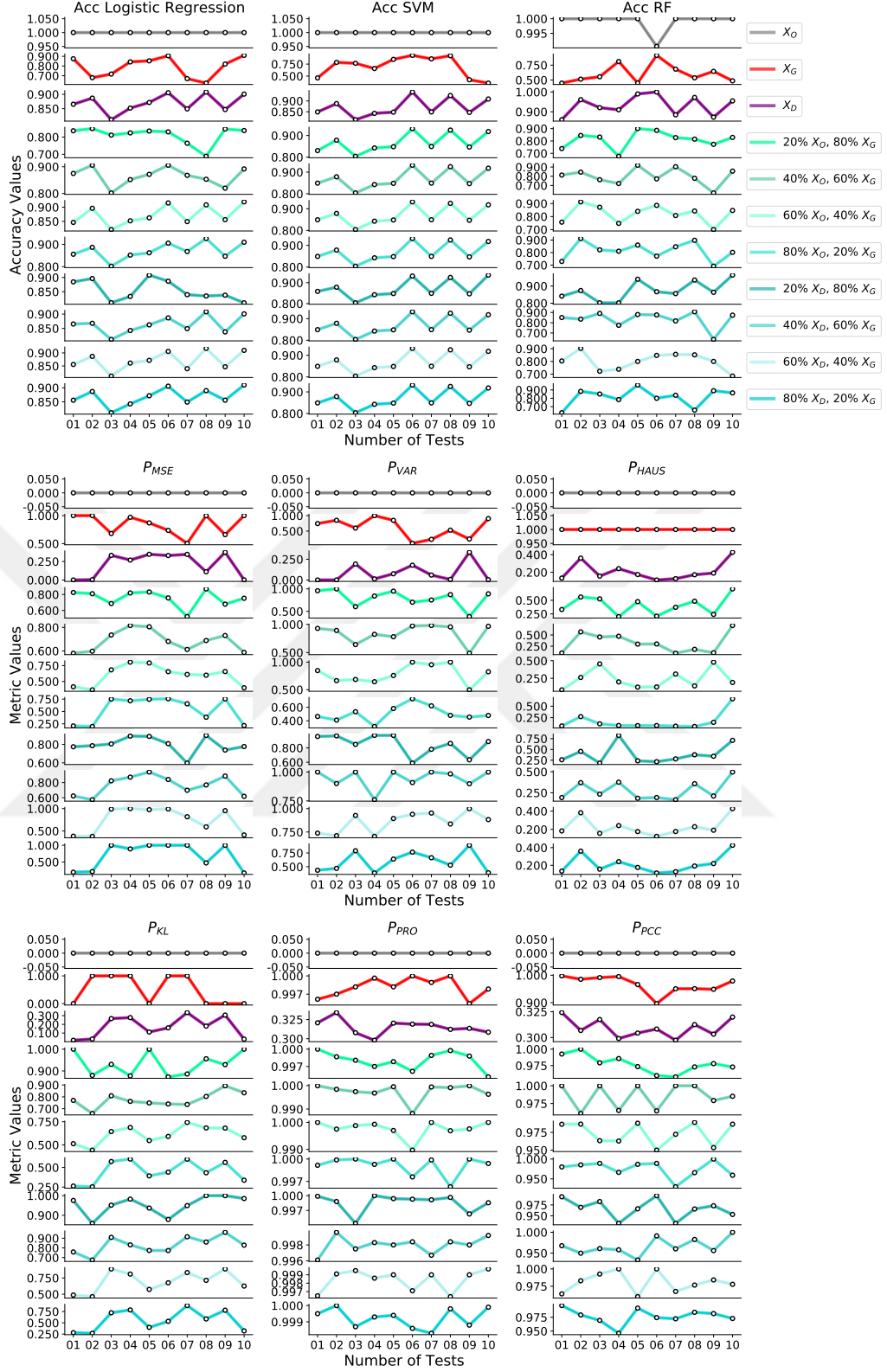


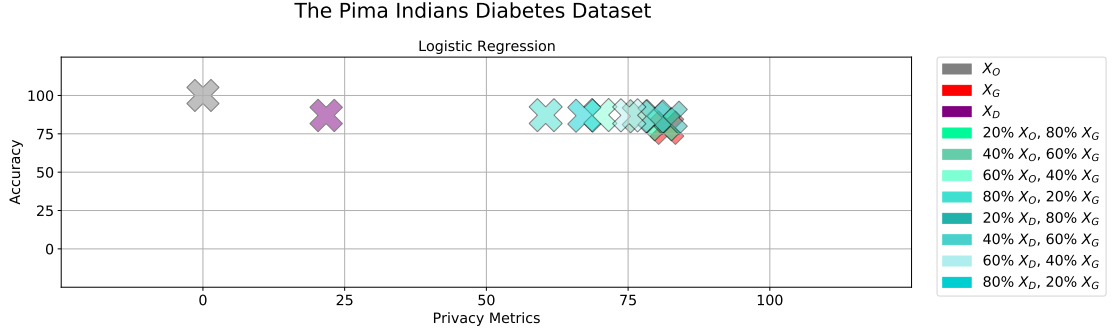
Figure 4.6 Variations Among 10 Tests (PP Methods)

80%). Similarly, four combinations of DP + GAN datasets were considered. To evaluate ML accuracy performance, we first analyze the accuracy results of the

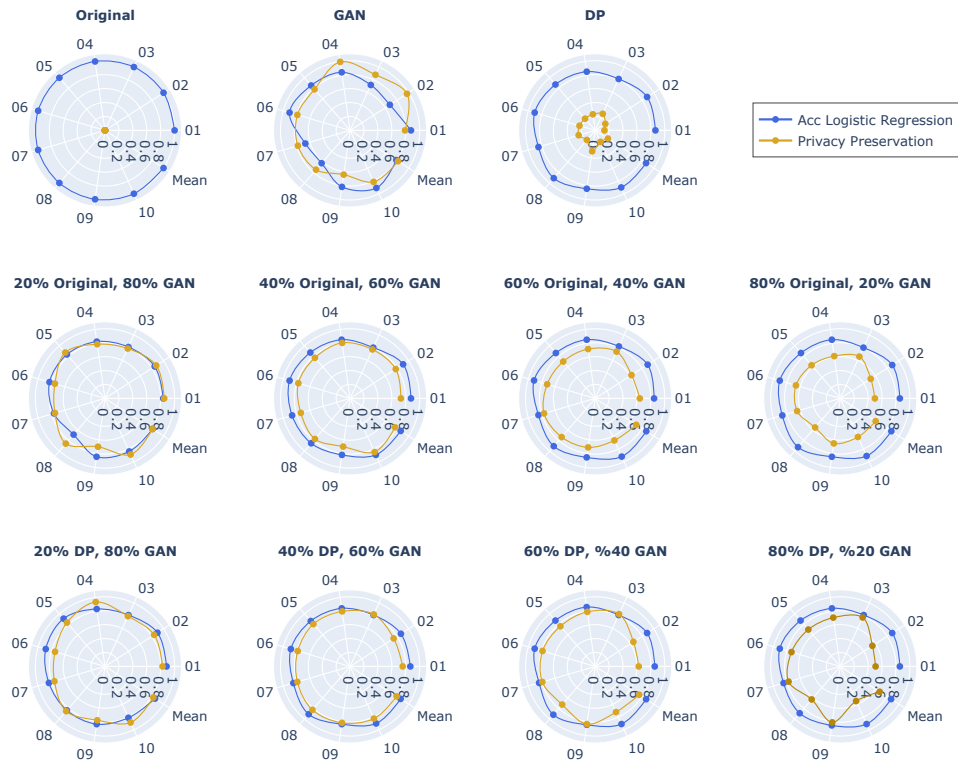
original dataset. As no perturbations are introduced by noisy elements, the accuracy values of Logistic Regression and SVM classifiers exhibit consistent behavior across the ten test cycles. Notably, the RF classifier achieves an accuracy value of 0.99050 on the sixth repetition. Appendix A.1 provides detailed information on all the test values. We utilized the Vanilla-GAN model to generate synthetic copies of the dataset. Examining the accuracy values of the three classifiers, Logistic Regression yields a minimum value of 0.6239 and a maximum value of 0.9099. SVM shows a range of minimum and maximum accuracy values of 0.3545 and 0.9320, respectively, while RF exhibits values of 0.4486 and 0.9143. Notably, SVM and RF classifiers demonstrate greater variability than Logistic Regression. The remaining plots in Figure 4.6 illustrate the privacy performance metrics across the ten test executions. The original dataset, which lacks noise or privacy protection, yields privacy metrics with zero values. On the other hand, the synthetic dataset attains the most successful privacy results among the six metrics. However, the other metrics exhibit greater variability except for the P_{HAUS} metric, which employs a metric space to measure dissimilarities. To ensure more stable results, we repeated synthetic data generation using CTGAN. Upon analyzing the performance of DP in the same figure, we observe increased deviations not only in the accuracy values of the three classifiers but also in the privacy metrics. This behavior can be attributed to DP’s utilization of the Laplace distribution, with an ϵ value of 0.01. The remaining datasets in this experiment consist of synthetic-GAN, DP, and original datasets, each at different ratios (i.e., 20%, 40%, 60%, and 80%). With these combinations, we aimed to achieve an optimal method that satisfies both accuracy and privacy expectations.

Figures 4.7, 4.8, and 4.9 provide a more detailed analysis of the relationship between accuracy and privacy preservation. Each figure focuses on a specific classification method applied to the eleven tested datasets. The plots present the average values of the six privacy metrics alongside the corresponding accuracy values over ten test repetitions.

In the first part of Figure 4.7, the accuracy values of Logistic Regression are plotted



(a) Accuracy vs. Privacy Analysis



(b) Accuracy vs. Privacy Analysis - 10 Test Repetitions

Figure 4.7 PP Methods - Logistic Regression

against the privacy metrics, represented as a percentage of the mean values from the ten repetitions. The original dataset, which lacks privacy protection measures, achieves a perfect accuracy of 100% but exhibits no privacy preservation. Although the remaining datasets demonstrate accuracy performances above 75%, they exhibit different privacy metric outcomes. The DP dataset performs the worst in privacy preservation, with a value of less than 25%. The other datasets range around 60%

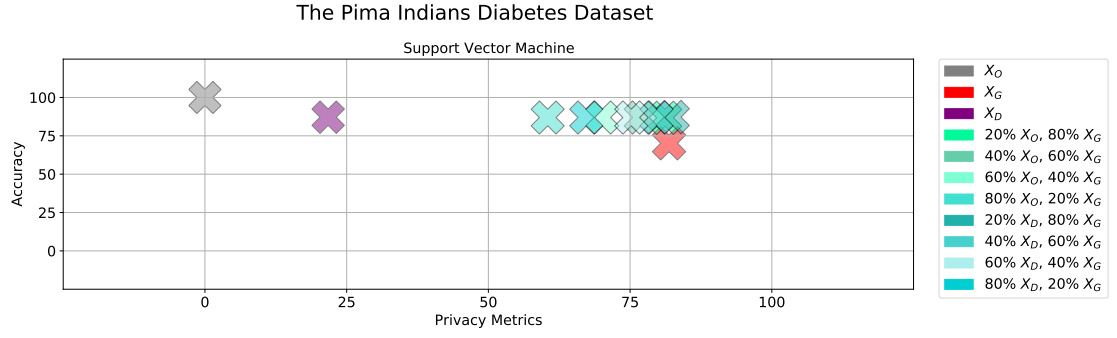
and 80% for privacy metrics. It is important to note that these values represent the mean of the six privacy metrics.

The second part of the same figure displays polar plots for each test repetition. Since the original dataset has no privacy protection, its accuracy values are plotted close to the edges of the polar plot, indicating 100% accuracy and zero privacy preservation. Conversely, the polar plot for the GAN dataset suggests better privacy preservation, although it exhibits some challenges in terms of accuracy performance. The polar plot for the DP dataset reveals the poorest privacy protection performance in the Pima Diabetes dataset when Logistic Regression is employed, with an ϵ value of 0.01. Among the combination datasets, the polar plots for the 20%, 40%, 60%, and 80% scenarios exhibit more promising results in terms of both Logistic Regression accuracy and privacy preservation. In the case of the Original+GAN and DP+GAN combinations, these four polar plots demonstrate accuracy and privacy preservation metrics of approximately 80%. However, there are certain gaps between accuracy and privacy metrics in the 60% and 80% combination scenarios.

Figure 4.8 provides insights into the performances of the SVM in terms of accuracy and privacy preservation across the ten test repetitions. In the first part of this figure, which displays the mean values, we observe some differences compared to Figure 4.7. The GAN dataset exhibits one of the best privacy protection outcomes but falls short in accuracy performance. The remaining datasets show similar trends as in the first part of the previous figure.

Moving to the second part of Figure 4.8, we find polar plots that illustrate the performance of each test repetition. Notably, the synthetic dataset demonstrates higher privacy preservation percentages than accuracy values, particularly in test executions between 01 and 04, as well as 09 and 10. It suggests that the synthetic dataset is more successful in maintaining privacy, albeit at the expense of accuracy.

Among the combination datasets, we observe better performances in terms of both SVM accuracy and privacy preservation for the original+GAN combination in the



(a) **Accuracy vs. Privacy Analysis**

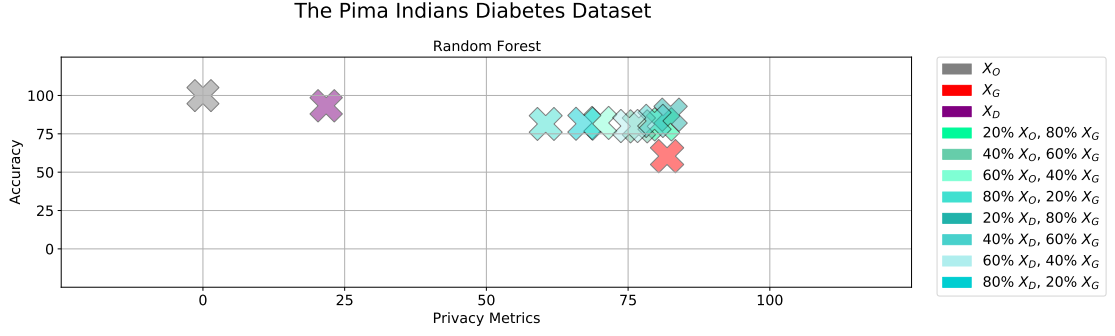


(b) **Accuracy vs. Privacy Analysis - 10 Test Repetitions**

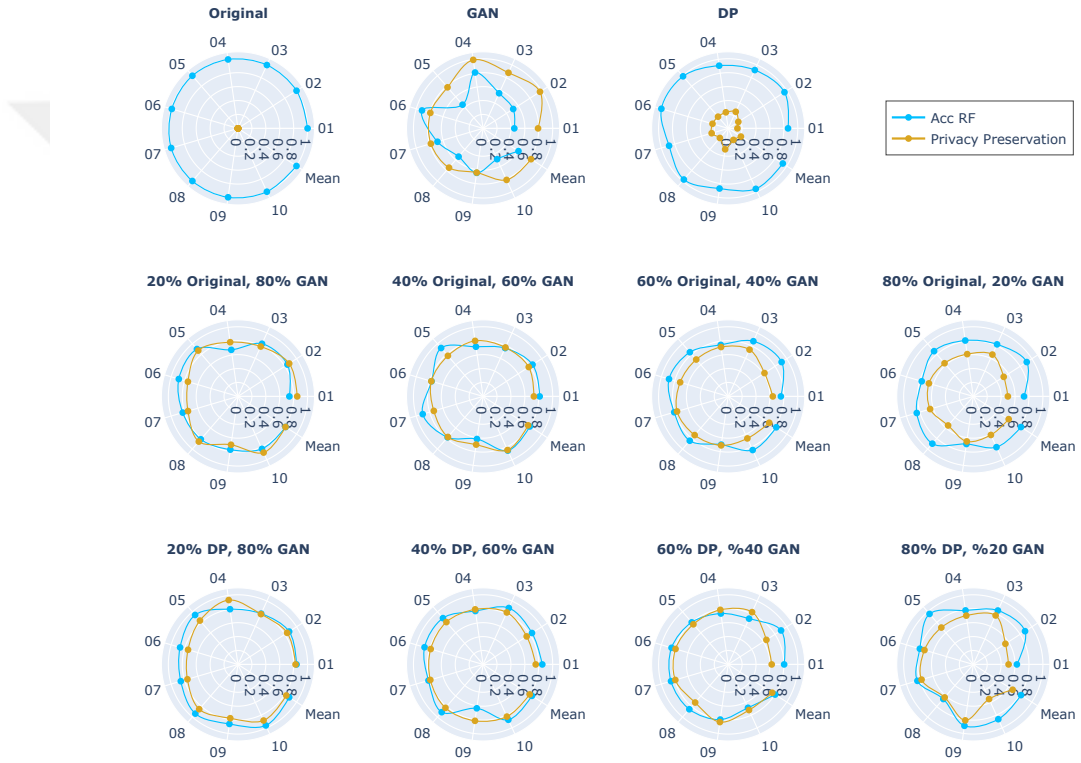
Figure 4.8 PP Methods - Support Vector Machine

20%, 40% cases, and the DP+GAN combination in the 20%, 40%, and 60% cases. These combinations balance accuracy and privacy preservation, showcasing their potential to be optimal solutions for both metrics.

Similarly, in Figure 4.9, we analyze the performance of the RF classifier over ten test executions. The mean values plotted in the first part of the figure reveal consistent



(a) Accuracy vs. Privacy Analysis



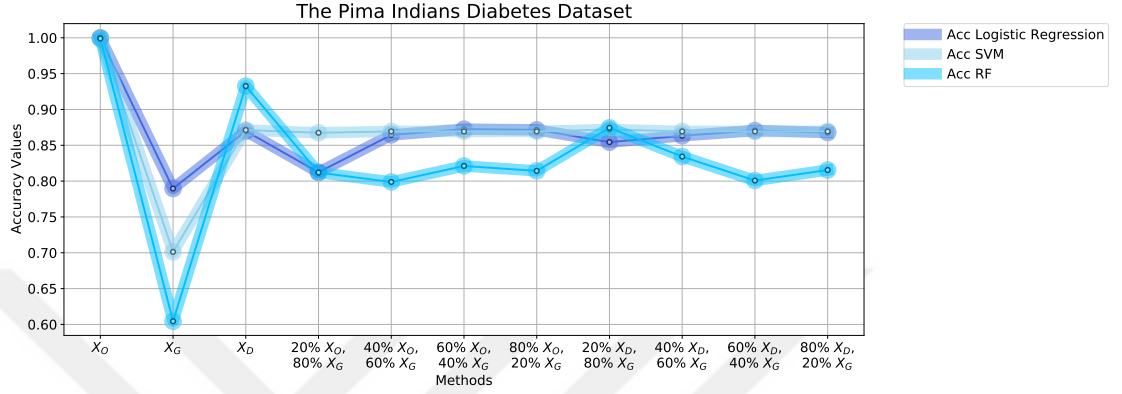
(b) Accuracy vs. Privacy Analysis - 10 Test Repetitions

Figure 4.9 PP Methods - Random Forest

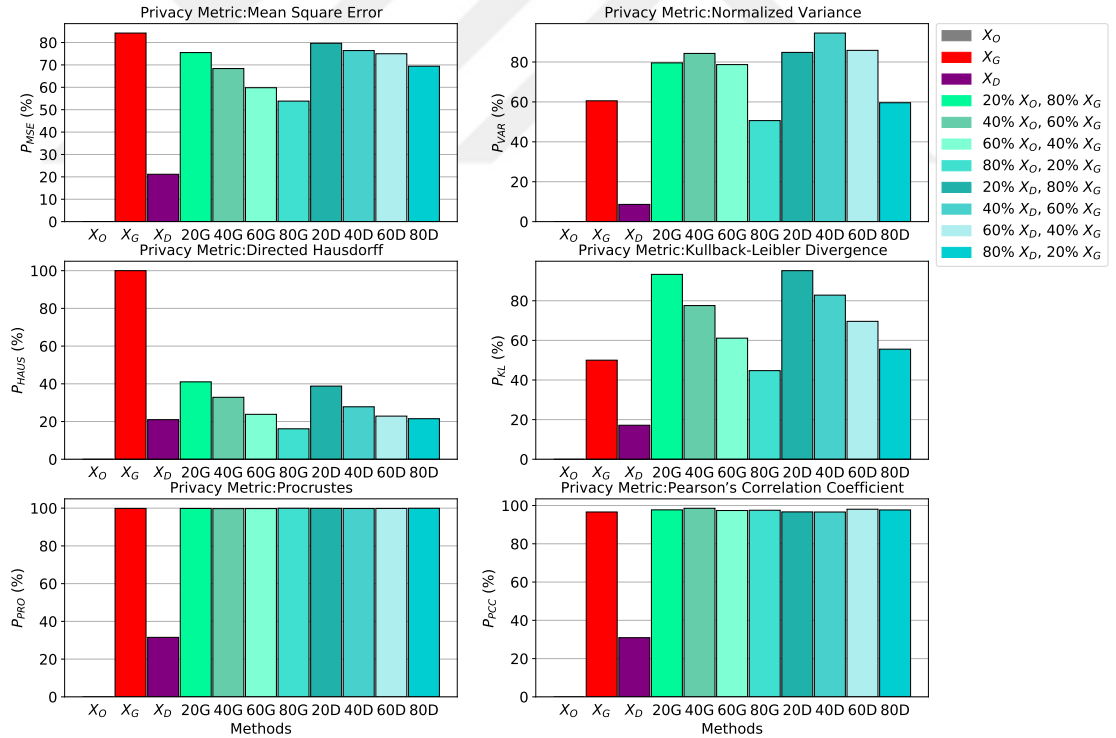
trends observed in the previous classifiers. The GAN dataset exhibits strong privacy preservation capabilities but struggles to achieve high accuracy. Conversely, the combination datasets show promising results in terms of both accuracy and privacy preservation.

The second part of Figure 4.9 provides a more detailed visualization of these outcomes.

The polar plot highlights that the GAN dataset excels in privacy protection but experiences accuracy failures in tests 5, 8, and 10. On the other hand, the DP dataset consistently falls short in privacy preservation across all ten tests. The combination datasets, except for the 80% case, demonstrate excellent performance in both original+GAN and DP+GAN scenarios.



(a) Accuracy Comparison: Logistic Regression, SVM, and RF



(b) Comprehensive Bar Subplot Analysis

Figure 4.10 PP Methods

To summarize the findings presented in Figures 4.7, 4.8, and 4.9, we provide an overview in Figure 4.10. The first part of this figure compares the mean accuracy

results of the three classifiers across the tested datasets. The original dataset achieves the highest accuracy values, while the GAN dataset performs the least favorably. The accuracy values of Logistic Regression and SVM are comparable in the combination datasets.

The second part of Figure 4.10 focuses on privacy metrics and their mean values. In this aspect, the GAN dataset demonstrates superior privacy preservation performance across almost all metrics, while the DP dataset consistently performs poorly in all six metrics. Notably, the 20% original+GAN and 20% DP+GAN combinations exhibit strong privacy preservation in all six cases. The experimental results highlight that both the DP and GAN methods can effectively enhance privacy preservation in classification algorithms while maintaining reasonable accuracy. Moreover, the mixed datasets combining GAN and DP demonstrate improved accuracy and privacy preservation compared to the individual methods.

In conclusion, the PP Methods experiments have provided compelling evidence that the DP and GAN methods can enhance the privacy preservation performance of classification algorithms while maintaining satisfactory accuracy. The combination of GAN and DP datasets has shown improvements in accuracy and privacy preservation compared to using the methods individually. These findings have significant implications for developing future privacy-preserving ML models. Therefore, our proposed method seeks to combine the strengths of GAN and DP, as indicated by the encouraging results obtained in this experiment.

However, before testing our proposed method, it is essential to thoroughly evaluate the use of CTGAN. The next section compares the vanilla-GAN and CTGAN models, which will inform our decision-making process.

4.6 Vanilla Gan versus CTGAN Experiments

We comprehensively compared Vanilla GAN and CTGAN to assess their performance. For Vanilla GAN, we utilized the configuration outlined in case 6 of Table 4.1, while

CTGAN was implemented following the methodology proposed by Xu and colleagues (L. Xu et al. 2019). Our evaluation was conducted on the Pima Indians Diabetes dataset, and we repeated the tests for a total of 10 executions. In each repetition, we generated synthetic copies of the original dataset using both Vanilla GAN and CTGAN.

Table 4.3 Privacy Preservation Methods Experiments Table

Vanilla GAN											
Tests	Original	GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN
Log.Reg. Accuracy	1.00000	0.78961	0.86978	0.81230	0.86446	0.87248	0.87157	0.85436	0.86325	0.87060	0.86786
SVM Accuracy	1.00000	0.70129	0.87128	0.86745	0.86934	0.86934	0.86934	0.87210	0.86934	0.86934	0.86934
RF Accuracy	0.99905	0.60445	0.93262	0.81193	0.79880	0.82114	0.81426	0.87440	0.83426	0.80081	0.81547
P_{MSE}	0.00000	0.84230	0.21159	0.75510	0.68376	0.59820	0.53834	0.79668	0.76411	0.74992	0.69430
P_{VAR}	0.00000	0.60545	0.08636	0.79562	0.84296	0.78702	0.50635	0.84807	0.94506	0.85830	0.59582
P_{HAUS}	0.00000	1.00000	0.21000	0.41041	0.32850	0.23820	0.16148	0.38786	0.27819	0.22872	0.21498
P_{KL}	0.00000	0.50000	0.17165	0.93309	0.77566	0.61114	0.44726	0.95182	0.82832	0.69602	0.55561
P_{PRO}	0.00000	0.99845	0.31528	0.99837	0.99748	0.99768	0.99929	0.99878	0.99802	0.99839	0.99923
P_{PCC}	0.00000	0.96614	0.30927	0.97723	0.98551	0.97383	0.97515	0.96657	0.96603	0.98056	0.97678

CTGAN											
Log.Reg. Accuracy	0.99798	0.88647	0.88755	0.87649	0.88082	0.87983	0.88861	0.88285	0.88184	0.88276	0.88477
SVM Accuracy	0.99898	0.88111	0.88864	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111
RF Accuracy	0.99212	0.85835	0.92700	0.84758	0.83209	0.83899	0.83616	0.85677	0.87218	0.88047	0.87139
P_{MSE}	0.00000	0.93216	0.26323	0.86975	0.79623	0.72168	0.66203	0.92350	0.89111	0.89725	0.84072
P_{VAR}	0.00000	0.75610	0.42676	0.70688	0.66365	0.61654	0.55335	0.87596	0.93911	0.99796	0.97078
P_{HAUS}	0.00000	0.68720	0.87835	0.66899	0.69990	0.73218	0.53354	0.73291	0.70422	0.76421	0.84258
P_{KL}	0.00000	0.98264	0.33486	0.89712	0.77865	0.71700	0.60551	0.96743	0.87272	0.87543	0.82872
P_{PRO}	0.00000	0.99757	0.31345	0.99789	0.99840	0.99836	0.99799	0.99793	0.99812	0.99756	0.99660
P_{PCC}	0.00000	0.97493	0.31273	0.97523	0.97361	0.97722	0.96494	0.98073	0.97026	0.97627	0.97433

One of the objectives of our test scenario was to evaluate the prediction performance of Vanilla GAN and CTGAN using three ML methods: Logistic Regression, SVM, and RF. By comparing the prediction performances of the two GAN models, we aimed to identify which model can generate a more accurate synthetic dataset, thereby improving the overall performance of the ML algorithms.

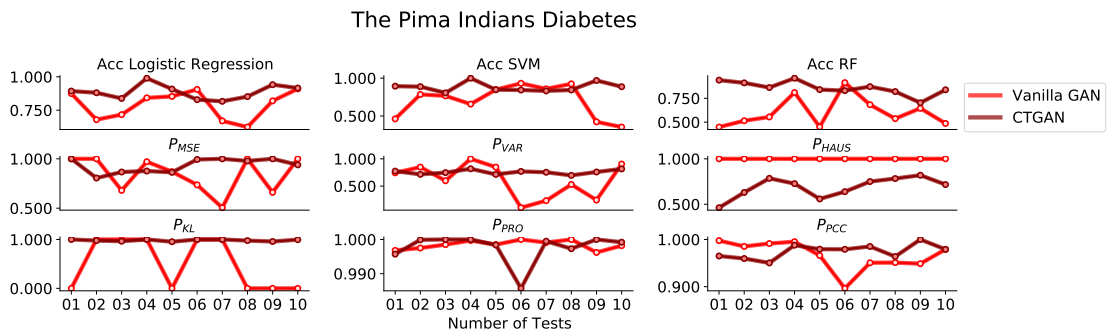


Figure 4.11 Variations Among 10 Tests - (Vanilla GAN vs CTGAN)

To visually demonstrate the accuracy comparisons between the two GAN methods

across the 10 executions, we present the results in the first three plots of Figure 4.11. Additionally, Table 4.4 provides the mean accuracy values obtained from the experiments. The table clearly illustrates the differences in accuracy between Vanilla GAN and CTGAN, particularly highlighting the benefits of CTGAN. Specifically, for Logistic Regression, SVM, and RF, Vanilla GAN achieved accuracies of 0.790, 0.701, and 0.604, respectively, while CTGAN demonstrated higher accuracies of 0.886, 0.881, and 0.858. Appendix B.1 provides detailed information on all the test values.

Table 4.4 Vanilla GAN vs. CTGAN Performance Comparison Table (Mean Values)

Model	Accuracy Log.Reg.	Accuracy SVM	Accuracy RF	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}
Vanilla GAN	0.790	0.701	0.604	0.842	0.605	1.000	0.500	0.998	0.966
CTGAN	0.886	0.881	0.858	0.932	0.756	0.687	0.982	0.997	0.974

In the second part of our test scenario, we employed six privacy metrics to quantify the differences between the original Pima Indians Diabetes dataset and its synthetic counterparts generated by Vanilla GAN and CTGAN. The procedure involved creating synthetic copies of the dataset using both GAN models and subsequently comparing these synthetic versions to the original dataset. This comparative analysis was conducted across 10 executions, and the results were evaluated using the six privacy metrics.

The remaining plots in Figure 4.11 illustrate the behavior of the privacy metrics for Vanilla GAN and CTGAN in each repetition, providing visual insights into their performance. Furthermore, Table 4.4 presents the mean values of the privacy metrics, which have been normalized to facilitate a percentage-based comparison. It is important to note that our privacy metrics are designed such that a value of zero indicates no difference between the original dataset and the compared dataset. Conversely, non-zero values indicate differentiation between the compared dataset and its original version. Upon examining the privacy metrics results, it becomes evident that the values obtained with CTGAN are generally higher than those achieved with Vanilla GAN. This indicates that CTGAN offers more privacy protection than Vanilla GAN, as it exhibits a greater ability to differentiate the synthetic dataset

from the original data.

4.7 Proposed Method Experiments

After employing GAN and DP techniques to enhance privacy protection and accuracy values on the Pima Diabetes dataset, we have ensured the proposed method's architecture needs to cover the GAN and DP combinations. We selected CTGAN for the synthetic data generation model and developed the proposed method as defined in Algorithm 4. In this section, we present the experiments of the proposed method versions. We selected four SCADA datasets (Wind Turbine, Steam Production, Energy Efficiency, Synchronous Motors) with continuous target variables, and because of that, we used the Linear Regression ML method. In the experiments, CTGAN was run with 100 epochs, and the parameter for the DP was set as $\epsilon = 0.01$. We applied traditional Linear Regression with a 0.2 ratio for splitting the data into train and test parts to solve the regression problem. The well-known *R-Squared* metric was used to evaluate the regression model accuracy. We measured the privacy protection efficiency with the six metrics P_{MSE} , P_{VAR} , P_{HAUS} , P_{KL} , P_{PRO} , and P_{PCC} .

The experimental setup consists of the following test scenarios. There is an additional case study to show the efficiency of the proposed algorithm on mentioned data which also has hidden sensitive features:

- (i) Original dataset (X_O)
- (ii) GAN, Synthetic dataset (X_G)
- (iii) DP dataset (X_D)
- (iv) Proposed method (GAN + DP) (X_{HGD}) with $n_s = 1 \dots 4$.
- (v) Alternative method with GAN (GAN Only) (X_{HG}) with $n_s = 1 \dots 4$ and finally
- (vi) Alternative method with DP (DP Only) (X_{HD}) with $n_s = 1 \dots 4$.

Selecting sensitive features within a dataset is crucial in our proposed method. This task is typically undertaken by the data owner, who understands the relative importance of each data component from a competitive business standpoint. To

assist in this process, we have introduced Algorithm 3, which allows for an assessment of the predictive capacity of individual features within the dataset.

In light of this, we have applied Algorithm 3 to four distinct datasets and documented the results in Table 4.5. We have limited our consideration to a maximum of four sensitive features for our study. However, it should be noted that the selection of sensitive features can be expanded by adjusting the parameter n_s . The source code employed in our research is publicly available (Hindistan 2023).

Table 4.5 Feature Prediction Performance Results

Wind Turbine Dataset			Steam Production Dataset		
Target	Input Features	<i>R-Squared</i>	Target	Input Features	<i>R-Squared</i>
F_0	$[F_1, F_2, F_3, F_4, F_5, F_6]$	0.909846	F_0	$[F_1, F_{10}, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9]$	0.612784
F_1	$[F_0, F_2, F_3, F_4, F_5, F_6]$	0.898608	F_1	$[F_0, F_{10}, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9]$	0.121494
F_2	$[F_0, F_1, F_3, F_4, F_5, F_6]$	0.925431	F_2	$[F_0, F_1, F_{10}, F_3, F_4, F_5, F_6, F_7, F_8, F_9]$	0.288768
F_3	$[F_0, F_1, F_2, F_4, F_5, F_6]$	0.618209	F_3	$[F_0, F_1, F_{10}, F_2, F_4, F_5, F_6, F_7, F_8, F_9]$	0.926371
F_4	$[F_0, F_1, F_2, F_3, F_5, F_6]$	0.939259	F_4	$[F_0, F_1, F_{10}, F_2, F_3, F_5, F_6, F_7, F_8, F_9]$	0.015930
F_5	$[F_0, F_1, F_2, F_3, F_4, F_6]$	0.971195	F_5	$[F_0, F_1, F_{10}, F_2, F_3, F_4, F_6, F_7, F_8, F_9]$	0.954873
F_6	$[F_0, F_1, F_2, F_3, F_4, F_5]$	0.969844	F_6	$[F_0, F_1, F_{10}, F_2, F_3, F_4, F_5, F_7, F_8, F_9]$	0.938801
			F_7	$[F_0, F_1, F_{10}, F_2, F_3, F_4, F_5, F_6, F_8, F_9]$	0.926604
			F_8	$[F_0, F_1, F_{10}, F_2, F_3, F_4, F_5, F_6, F_7, F_9]$	0.661345
			F_9	$[F_0, F_1, F_{10}, F_2, F_3, F_4, F_5, F_6, F_7, F_8]$	0.873554
			F_{10}	$[F_0, F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9]$	0.929339
Energy Efficiency Dataset			Synchronous Motors Dataset		
Target	Input Features	<i>R-Squared</i>	Target	Input Features	<i>R-Squared</i>
F_0	$[F_1, F_2, F_3, F_4, F_6, F_8]$	0.991398	F_0	$[F_1, F_2, F_3]$	0.592514
F_1	$[F_2, F_3]$	1.000000	F_1	$[F_2]$	1.000000
F_2	$[F_1, F_3]$	1.000000	F_2	$[F_1]$	1.000000
F_3	$[F_1, F_2]$	1.000000	F_3	$[F_4]$	1.000000
F_4	$[F_0, F_1, F_2, F_5, F_6, F_7, F_8, F_9]$	0.973561	F_4	$[F_3]$	1.000000
F_5	$[F_0, F_1, F_2, F_4, F_6, F_7, F_8, F_9]$	0.006745			
F_6	$[F_0, F_1, F_2, F_3, F_4, F_5, F_7, F_8, F_9]$	0.475088			
F_7	$[F_0, F_1, F_2, F_4, F_5, F_6, F_8, F_9]$	0.066359			
F_8	$[F_0, F_1, F_2, F_4, F_5, F_6, F_7, F_9]$	0.969085			
F_9	$[F_0, F_1, F_2, F_4, F_5, F_6, F_7, F_8]$	0.958607			

We experimented on the four specified datasets, carrying out ten repetitions for each experiment.

To visualize the dispersions observed in the proposed hybrid approach (X_{HGD}) with $n_s = 1 \dots 4$, we presented the repetition results for the Wind Turbine dataset in Figure 4.12. The repetition plots for the remaining three datasets can be found in

Appendix C, along with the corresponding numerical values in tables.

Figure 4.12 displays seven datasets: the Original (X_O), GAN copy (X_G), DP copy (X_D), and the proposed method GAN+DP versions with $n_s = 1 \dots 4$ ($X_{HGD} (n_s = 1)$, $X_{HGD} (n_s = 2)$, $X_{HGD} (n_s = 3)$, and $X_{HGD} (n_s = 4)$). The ML performance, measured by the *R-Squared* values, is presented for each dataset across the ten executions. Similarly, the performance of the six privacy metrics is demonstrated for all datasets except the original, as its privacy metric results are zero. The numeric results for the privacy metrics can be found in the table in Appendix C. In Figure 4.12, the mean values of the six metrics are used to illustrate the overall behavior across the test executions. Additionally, the mean and standard deviation values are calculated for each subplot, and the confidence interval area is highlighted with a light salmon color, assessing variability and precision in the execution results.

A closer examination of Figure 4.12 focusing on the *R-Squared* values reveals that X_O maintains a consistent value of 0.9689 across all repetitions. On the other hand, X_G and X_D exhibit variations during the repetitions, with most of the values falling within the confidence interval, except for the second repetition of X_G and the tenth execution of X_D . Notably, X_{HGD} shows differences in performance for $n_s = 2$ and $n_s = 3$, outperforming X_G and X_D in terms of *R-Squared* values throughout the test executions.

The privacy protection performance of the datasets follows a similar pattern, with X_{HGD} for $n_s = 1 \dots 4$ exhibiting variability across the repetitions, reaching a minimum point in the seventh test. Analyzing the executions of X_G and X_D provides further insights into this behavior, as the seventh test reveals a minimum point for accuracy and privacy metrics in X_G .

One can examine the details of the ten executions by referring to Figure 4.12 and the additional plots in Appendix C. Conducting multiple executions in our experiments allowed us to evaluate outcome consistency, ensure reliability, and facilitate statistical analysis and sensitivity assessments. The further plots utilize the mean values of

X_{HGD} : Wind Turbine Dataset

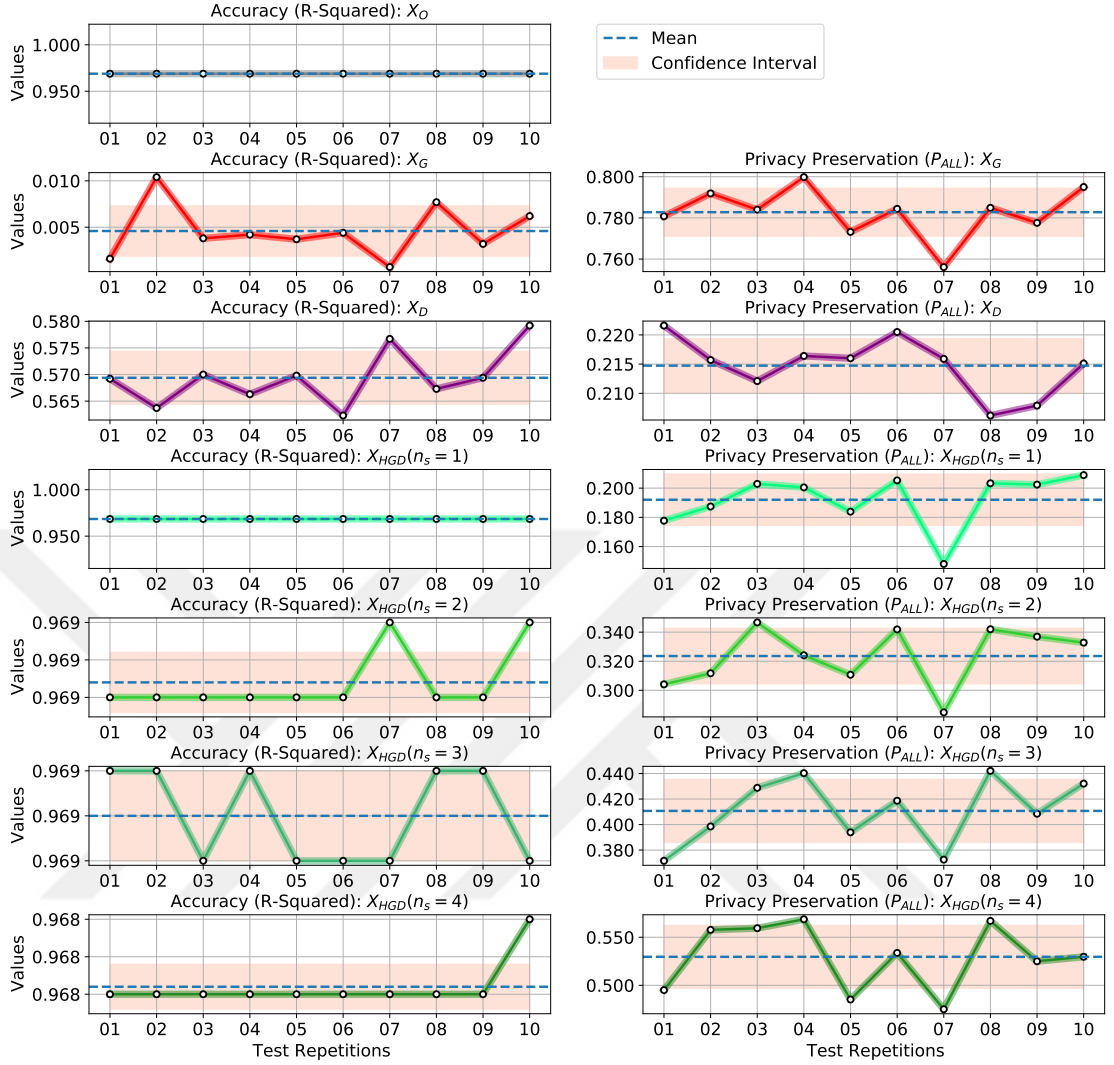


Figure 4.12 Variations Among 10 Tests (X_{HGD})

these repetitions to provide an overview of the results.

The performance of ML models in terms of both accuracy and privacy preservation are compared across seven tested methods and four datasets for the variable X_{HGD} , as depicted in Figure 4.13. Detailed results for the four datasets are presented in Tables 4.6 and 4.7.

Upon examining the Wind Turbine dataset, it is observed that CTGAN (X_G) exhibits the lowest performance in terms of *R-squared*, while DP (X_D) achieves less than 50% performance. CTGAN demonstrates superior privacy preservation capabilities,

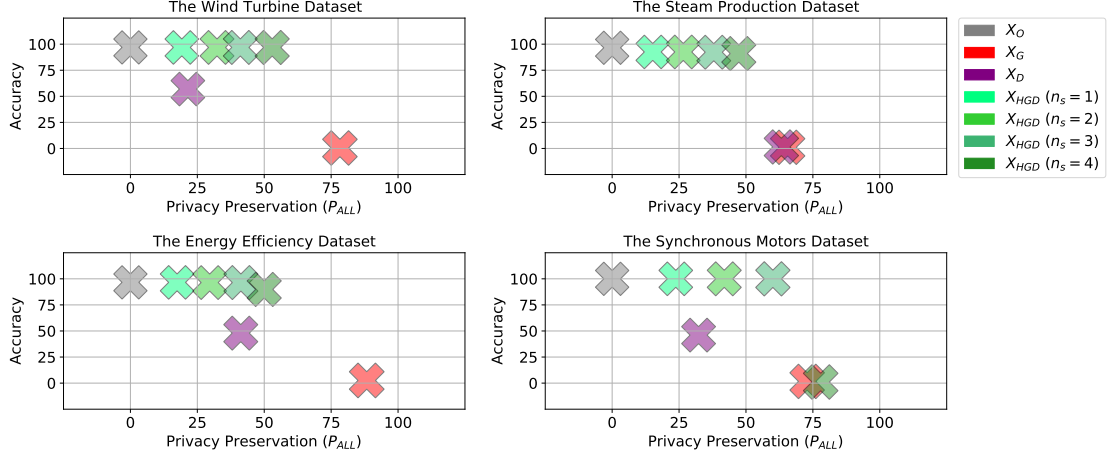


Figure 4.13 Accuracy vs. Privacy Analysis of X_{HGD} (Linear Regression)

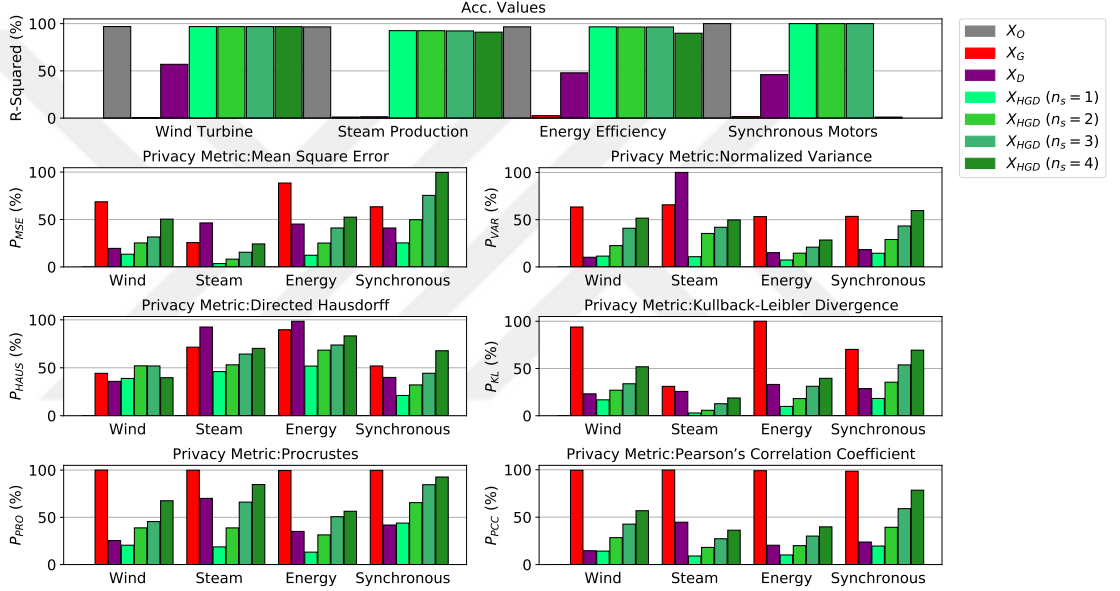


Figure 4.14 Comprehensive Bar Subplot Analysis (X_{HGD})

while the proposed hybrid methods generated using Algorithm 4 exhibited even better performance. The accuracy of the Linear Regression model trained on the X_{HGD} dataset, as shown in Table 4.6, is comparable to that of the original dataset. The minimum *R-squared* score for $n_s = 4$ is 0.96821. Additionally, the proposed approach demonstrated strong privacy preservation capabilities, as indicated by the considered privacy metrics. It is worth noting that while the original dataset (X_O) achieved the highest accuracy, it offered no privacy protection. On the other hand, the proposed method (X_{HGD}) with $n_s = 4$ achieved similar accuracy to X_O while outperforming other privacy preservation approaches such as CTGAN (X_G) and

DP (X_D) in terms of P_{MSE} . These findings suggest that the proposed approach achieves high method performance while providing better privacy protection than its counterparts. This observation is further supported by the results presented in Table 4.6, which highlight the advantages of the proposed approach. It is important to emphasize that a practical privacy preservation approach for an ML system should strike a balance between method accuracy and privacy protection, which is effectively achieved by the proposed method.

Table 4.6 Test Results: Wind Turbine and Steam Production

Wind Turbine								Steam Production						
Tests	R -Squared	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}	R -Squared	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}
Original	0.96890	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.96530	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
CTGAN	0.00459	0.18957	2.51316	0.44230	0.28885	0.99973	1.00002	0.01131	0.07090	2.60373	0.71381	0.09563	0.99821	1.00200
DP	0.56939	0.05402	0.40570	0.35844	0.07128	0.25435	0.14655	0.01430	0.12820	3.96137	0.92479	0.07919	0.70026	0.44918
X_{HGD}		Proposed Method - GAN + DP						X_{HGD}		Proposed Method - GAN + DP				
($n_s=1$)	0.96850	0.03700	0.45330	0.38754	0.05172	0.20485	0.14272	0.92534	0.00973	0.42927	0.45870	0.00901	0.18689	0.09099
($n_s=2$)	0.96852	0.06999	0.89343	0.51935	0.08299	0.38828	0.28543	0.92515	0.02272	1.40318	0.53110	0.01784	0.38795	0.18200
($n_s=3$)	0.96855	0.08744	1.62214	0.51768	0.10401	0.45484	0.42885	0.92283	0.04319	1.66255	0.64312	0.03910	0.66082	0.27391
($n_s=4$)	0.96821	0.13885	2.04442	0.39638	0.15928	0.67519	0.57100	0.90980	0.06771	1.97026	0.70122	0.05783	0.84672	0.36445
X_{HG}		Proposed Method - GAN Only						X_{HG}		Proposed Method - GAN Only				
($n_s=1$)	0.96850	0.02543	0.40157	0.35186	0.04656	0.17848	0.14272	0.92536	0.00378	0.23791	0.39633	0.00603	0.10413	0.09126
($n_s=2$)	0.96851	0.04398	0.73657	0.41390	0.07372	0.32948	0.28529	0.92534	0.00730	0.54773	0.45999	0.01109	0.19526	0.18272
($n_s=3$)	0.96854	0.05371	1.28615	0.44309	0.08943	0.38829	0.42914	0.92298	0.02054	0.75436	0.53867	0.03099	0.52430	0.27381
($n_s=4$)	0.96819	0.08699	1.64156	0.37567	0.13829	0.62102	0.57157	0.90979	0.03481	0.96147	0.58614	0.04790	0.76283	0.36362
X_{HD}		Proposed Method - DP Only						X_{HD}		Proposed Method - DP Only				
($n_s=1$)	0.96873	0.00140	0.01430	0.16782	0.00584	0.00950	0.00584	0.93476	0.00559	0.19000	0.36872	0.00540	0.09552	0.03901
($n_s=2$)	0.96872	0.01000	0.09860	0.34616	0.01584	0.06280	0.03542	0.93466	0.02153	1.03009	0.58430	0.01360	0.28749	0.10162
($n_s=3$)	0.96868	0.01430	0.22430	0.36679	0.02155	0.08514	0.07402	0.93387	0.02793	1.08072	0.62313	0.02000	0.34276	0.11964
($n_s=4$)	0.96866	0.01860	0.24860	0.32111	0.03014	0.10672	0.08542	0.92728	0.04333	1.20162	0.68007	0.02910	0.44685	0.15072

Figure 4.14 allows for comparing the accuracy and privacy metric values. The figure demonstrates that the six privacy metrics yield better results when applied to the GAN dataset. Conversely, the GAN dataset exhibits the lowest R -squared performance among the tested methods for the Wind Turbine dataset.

A similar pattern is observed in the other three datasets. For instance, in the Steam Production dataset, particular features, such as natural gas usage (F_0), gas pressure (F_1), steam pressure (F_2), and tank amount (F_4), are directly linked to production costs and can potentially disclose sensitive information regarding product recipes and the manufacturer's profit margins. These features were identified as sensitive and used to evaluate the system's privacy. Different combinations of sensitive features

and columns were considered, including $n_s = 1$ (F_0), $n_s = 2$ (F_0 and F_4), $n_s = 3$ (F_0 , F_4 , and F_1), and $n_s = 4$ (F_0 , F_4 , F_1 , and F_2). The Steam Production dataset exhibited the worst *R-squared* value for X_D among the tested methods in Figure 4.13 and Figure 4.14, illustrating the impact of different approaches on *R-squared* scores and privacy metrics for the Steam Production dataset. The results indicate that the X_G and X_D approaches offer better privacy metrics but result in a decline in *R-squared* scores. It implies that these methods effectively ensure data privacy but do not contribute to the accuracy of Linear Regression predictions. Conversely, our results demonstrate that the proposed method (X_{HGD}) versions, for $n_s = 1$ to $n_s = 4$, achieve comparable performance in terms of *R-squared* values to those of the original dataset (X_O) across all four datasets, while also providing enhanced privacy preservation. The $n_s = 3$ and $n_s = 4$ methods exhibit privacy preservation values around 50%. Therefore, X_{HGD} effectively preserves the privacy of sensitive features while minimizing the loss in Linear Regression accuracy. Consequently, we can confidently claim that X_{HGD} is a promising candidate for data privacy protection in Linear Regression.

Table 4.7 Test Results: Energy Efficiency and Synchronous Motors

Energy Efficiency								Synchronous Motors							
Tests	R -Squared	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}	R -Squared	P_{MSE}	P_{VAR}	P_{HAUS}	P_{KL}	P_{PRO}	P_{PCC}	
Original	0.96610	0.0000	0.0000	0.00000	0.0000	0.00000	0.0000	1.00000	0.0000	0.0000	0.00000	0.0000	0.00000	0.0000	
CTGAN	0.02566	0.2446	2.1088	0.89649	0.3078	0.99473	0.9952	0.01632	0.1754	2.1196	0.51816	0.2158	0.99679	0.9900	
DP	0.47923	0.1248	0.5980	0.98543	0.1019	0.35066	0.2050	0.46022	0.1134	0.7266	0.39988	0.0884	0.41857	0.2392	
X_{HGD}	Proposed Method - GAN + DP							X_{HGD}	Proposed Method - GAN + DP						
($n_s=1$)	0.96591	0.0345	0.2898	0.51791	0.0303	0.13168	0.1020	1.00000	0.0714	0.5734	0.21131	0.0562	0.43860	0.1964	
($n_s=2$)	0.96424	0.0701	0.5768	0.68265	0.0559	0.31396	0.2007	1.00000	0.1388	1.1500	0.32091	0.1092	0.65536	0.3952	
($n_s=3$)	0.96428	0.1140	0.8289	0.73726	0.0959	0.50649	0.3021	1.00000	0.2110	1.7166	0.44353	0.1654	0.84460	0.5928	
($n_s=4$)	0.89832	0.1456	1.1283	0.83090	0.1218	0.56384	0.3999	0.01075	0.2784	2.3622	0.67580	0.2134	0.92613	0.7886	
X_{HG}	Proposed Method - GAN Only							X_{HG}	Proposed Method - GAN Only						
($n_s=1$)	0.96590	0.0204	0.2241	0.52191	0.0262	0.09196	0.1013	1.00000	0.0396	0.4232	0.19297	0.0476	0.34433	0.1968	
($n_s=2$)	0.96441	0.0412	0.4394	0.58719	0.0491	0.24325	0.2018	1.00000	0.0786	0.8354	0.31848	0.0956	0.54328	0.3932	
($n_s=3$)	0.96430	0.0691	0.6381	0.62507	0.0844	0.42683	0.3021	1.00000	0.1176	1.2546	0.37939	0.1416	0.77629	0.5928	
($n_s=4$)	0.89839	0.0848	0.8555	0.67072	0.1055	0.47743	0.4005	0.00973	0.1472	1.6902	0.51953	0.1762	0.88500	0.7890	
X_{HD}	Proposed Method - DP Only							X_{HD}	Proposed Method - DP Only						
($n_s=1$)	0.96570	0.0081	0.0465	0.41032	0.0084	0.03373	0.0175	1.00000	0.0256	0.1400	0.15038	0.0196	0.13922	0.0472	
($n_s=2$)	0.96509	0.0243	0.1286	0.54980	0.0189	0.09395	0.0428	1.00000	0.0510	0.2824	0.22167	0.0388	0.24479	0.0944	
($n_s=3$)	0.96503	0.0370	0.1736	0.62055	0.0306	0.13666	0.0599	1.00000	0.0764	0.4240	0.25910	0.0580	0.32744	0.1422	
($n_s=4$)	0.90942	0.0437	0.2256	0.64335	0.0376	0.15803	0.0786	0.76831	0.0948	0.5752	0.33463	0.0728	0.37596	0.1902	

However, there is one exception in the case of the Synchronous Motors dataset.

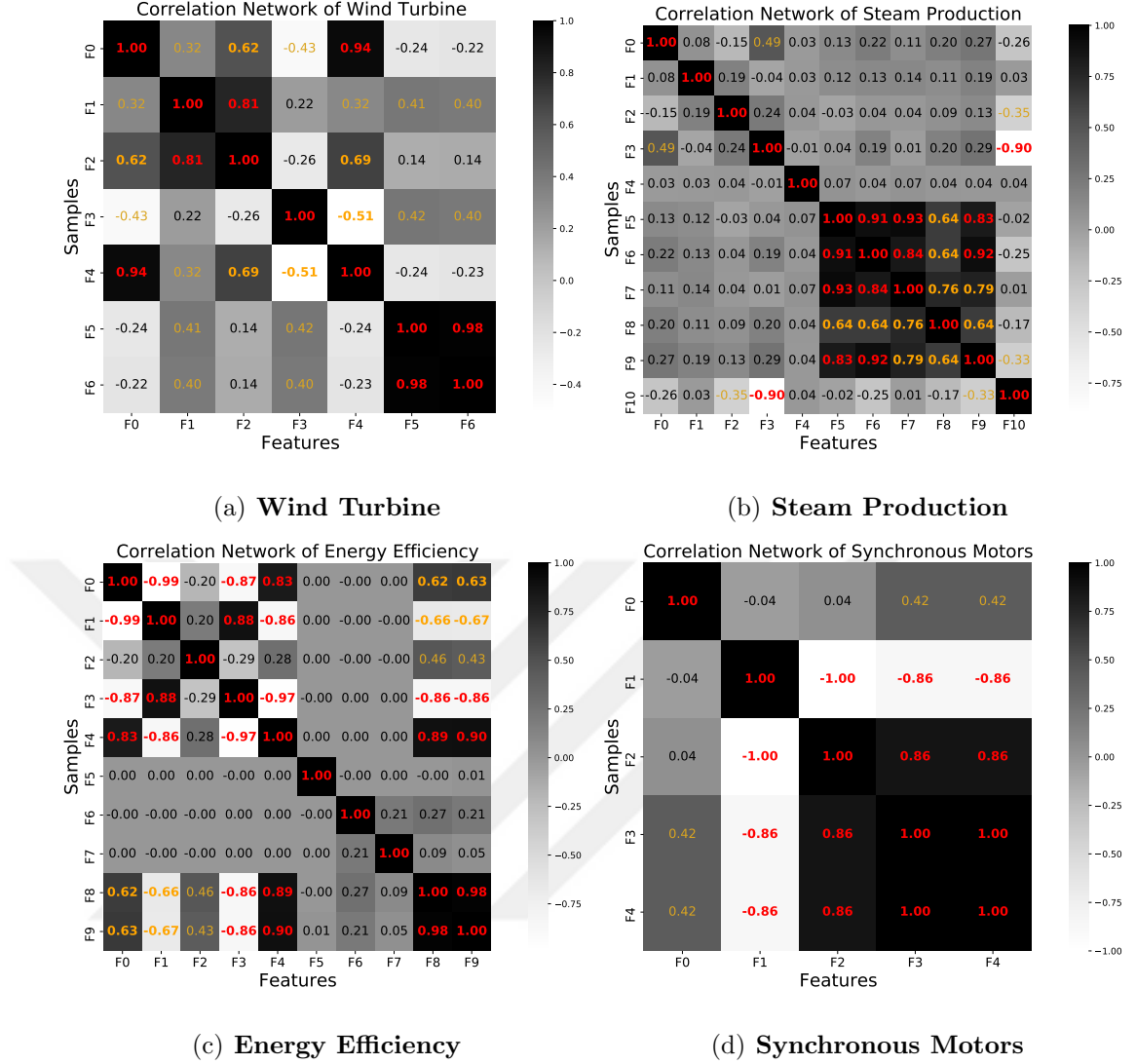


Figure 4.15 Dataset Correlation Heatmaps

The experimental setup revealed an unusual behavior observed in the proposed method with $n_s = 4$ for the Synchronous Motors data. As shown in Table 4.7, the obtained *R-squared* score for this case was 0.01075, which is inferior to that of X_D 0.46022. In contrast, the methods from $n_s = 1$ to $n_s = 3$ within X_{HGD} demonstrated a perfect *R-squared* score of 1.00000. This deviation can be attributed to highly correlated features in the Synchronous Motors data, indicating the existence of dependent features (Figure 4.15). Specifically, feature F_1 exhibits a correlation coefficient of -1.0000 with F_2 , -0.8610 with F_3 , and -0.8610 with F_4 . Additionally, F_3 demonstrates strong correlations with F_1 -0.8610 , F_2 0.8610 , and F_4 1.0000 . The high correlation between sensitive and non-sensitive features can undermine the

effectiveness of our approach. Applying an appropriate dimensionality reduction technique may be necessary before implementing the proposed method.

Our findings demonstrate the effectiveness of the proposed approach (X_{HGD}) in achieving high method performance and privacy preservation in Linear Regression across multiple datasets. The approach balances accuracy and privacy protection and outperforms existing methods such as CTGAN and DP. However, certain dataset characteristics, such as highly correlated features, may affect the performance of the proposed approach, necessitating additional preprocessing techniques. Nonetheless, the results support the notion that X_{HGD} holds promise as a robust solution for data privacy protection in Linear Regression.

We examined alternative methods to the proposed approach, namely GAN Only (X_{HG}) and DP Only (X_{HD}). We followed similar steps in the proposed method in these alternatives, except for the final stages. In the X_{HG} method, we excluded the application of DP methodology after substituting the sensitive features in the synthetic dataset (X_G). In other words, we only replaced the sensitive features in the synthetic dataset without incorporating the DP technique. Similarly, in the X_{HD} method, we replaced the sensitive features in the DP dataset (X_D). Hence, we refer to X_{HG} as the GAN-applied dataset without a random noise method and X_{HD} as the DP-applied dataset without a synthetic method.

Both X_{HG} and X_{HD} utilized $n_s = 1 \dots 4$ features. The experimental results for these methods are presented in Tables 4.6 and 4.7.

As shown in Table 4.6, the X_{HG} and X_{HD} methods exhibit similar R-Squared scores to the proposed method (X_{HGD}), with values close to 0.96850 for cases with $n_s = 1 \dots 3$. However, the proposed method (X_{HGD}) provides enhanced privacy protection. For instance, the privacy metrics (P_{MSE} , P_{VAR} , P_{HAUS} , P_{KL} , P_{PRO} , P_{PCC}) for the X_{HGD} method ($n_s = 3$) outperform those of both X_{HG} ($n_s = 3$) and X_{HD} ($n_s = 3$), with respective values of (0.08744, 1.62214, 0.51768, 0.10401, 0.45484, 0.42885), (0.05371, 1.28615, 0.44309, 0.08943, 0.38829, 0.42914), and (0.01430, 0.22430, 0.36679,

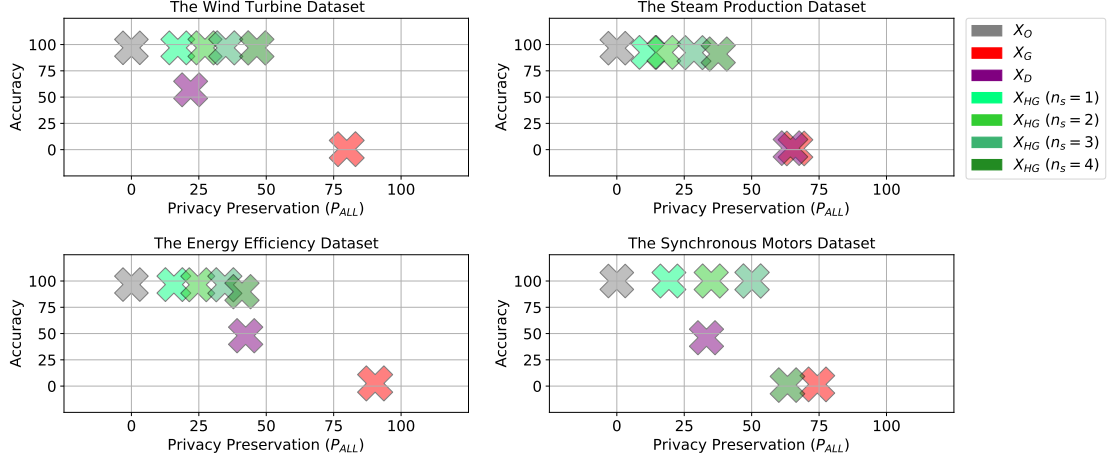


Figure 4.16 Accuracy vs. Privacy Analysis of X_{HG} (Linear Regression)

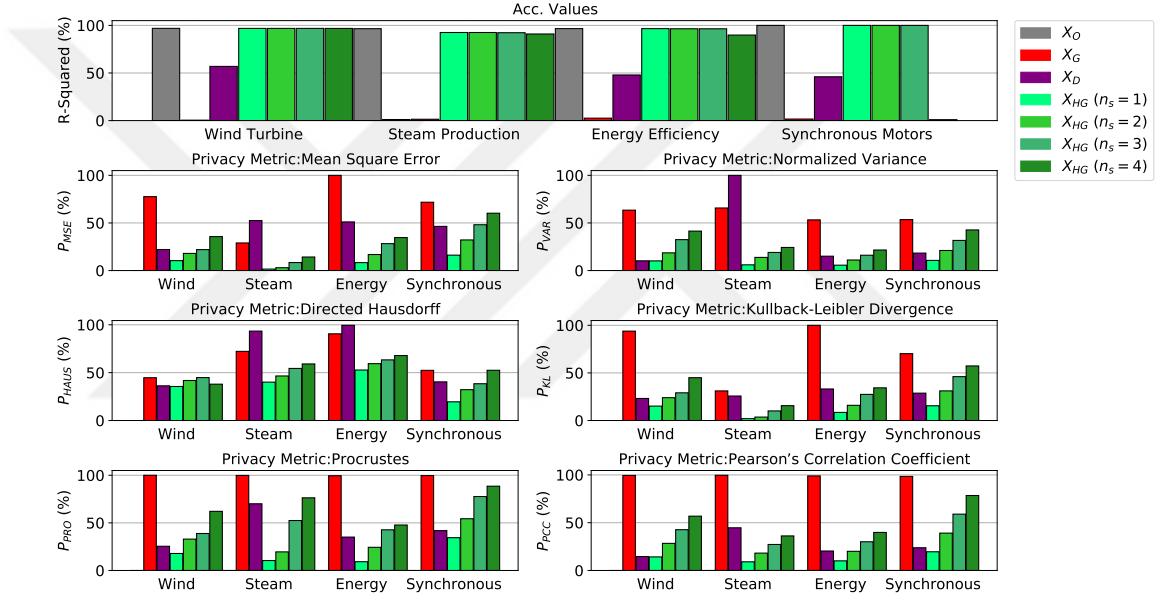


Figure 4.17 Comprehensive Bar Subplot Analysis (X_{HG})

0.02155, 0.08514, 0.07402). Therefore, incorporating the DP approach alongside GAN protection in the X_{HG} method enhances the privacy preservation of the proposed approach (X_{HGD}) for this dataset. Similar trends were observed for the other datasets analyzed in our experiments.

Figure 4.16 and Figure 4.17 depict the test results illustrating the variations among the four datasets and privacy metrics for X_{HG} . The R-Squared values demonstrate comparable performance to X_{HGD} , while visual differences can be observed in the metric privacy values plotted.

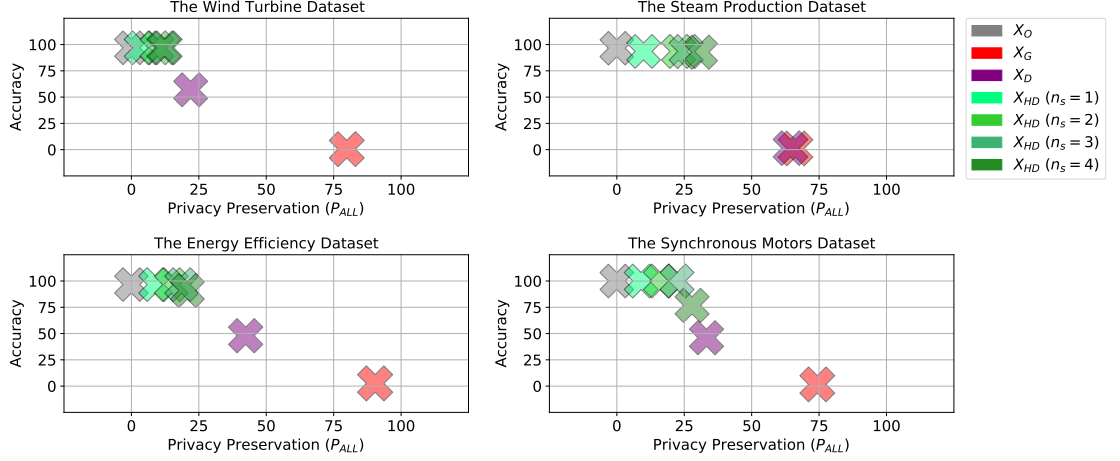


Figure 4.18 Accuracy vs. Privacy Analysis of X_{HD} (Linear Regression)

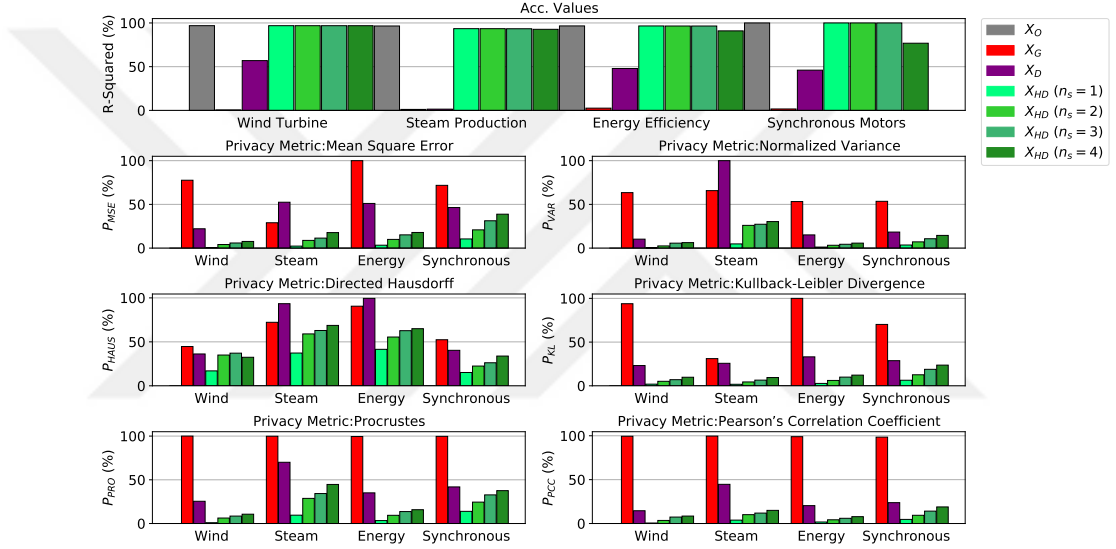


Figure 4.19 Comprehensive Bar Subplot Analysis (X_{HD})

Similarly, Figure 4.18 and Figure 4.19 present the experimental results for X_{HD} . These plots provide insights into the performance of X_{HD} compared to X_{HGD} and X_{HG} methods. While the accuracy performance appears relatively similar among the three methods, noticeable differences emerge in the privacy metrics. Specifically, it can be observed that X_{HD} exhibits lower performance in terms of privacy metrics compared to the other two methods. It implies that the privacy-preserving capabilities of X_{HD} are less effective than those of X_{HGD} and X_{HG} .

Through the experiments conducted on X_{HGD} , X_{HG} , and X_{HD} , we have observed that the proposed method, X_{HGD} , demonstrates the ability to preserve privacy while

maintaining a comparable ML prediction performance to that of the original dataset, X_O . This finding has significant implications for the case study in Chapter 4.9, which aims to extract sensitive information from a subset of X_O . In this case study, the method X_{HGD} distinguishes itself from the alternatives, X_{HG} and X_{HD} . However, an alternative approach involving dimension reduction is worth mentioning before delving into the case study.

4.8 PCA Experiments

We conducted PCA on four datasets: Wind Turbine, Steam Production, Energy Efficiency, and Synchronous Motors.

Figure 4.20 illustrates the PCA test scenarios where we applied the technique for different values of k , specifically $k = 2, 3$, and 4. Subsequently, we employed the DP function to introduce some noise. These experiments can be considered an alternative to generating synthetic datasets using the CTGAN function. Instead, we reduced the datasets to a certain extent using PCA and then reconstructed them to their original feature size. The aim was to explore whether dimensionality reduction and reconstruction could serve as a substitute for GAN in achieving high accuracy in ML predictions while preserving privacy.

For the experiments, we used the reduced dataset $X_{reduced} \in \mathbb{R}^{m \times k}$ for $k = 2, 3$, and 4. Table 4.8 presents the reconstructed errors, which indicate the loss of information incurred during the dimensionality reduction process. In the case of $k = 2$, we observed relative differences in the reconstruction errors for each feature across the datasets. Lower values corresponded to better reconstruction, suggesting that the dataset accurately captured the variability of the feature using only two PCA components. We expected higher errors regarding privacy preservation, as they enable protection through differentiation. Comparing the results, we noticed that the third dataset, Energy Efficiency, exhibited higher reconstruction errors than the other datasets, indicating relatively lower accuracy in representing the features using

Table 4.8 PCA Reconstruction Errors

	Wind Turbine			Steam Production			Energy Efficiency			Synchronous Motors		
Features	($k=2$)	($k=3$)	($k=4$)	($k=2$)	($k=3$)	($k=4$)	($k=2$)	($k=3$)	($k=4$)	($k=2$)	($k=3$)	($k=4$)
F_0	0.0042	0.0034	0.0034	0.0140	0.0139	0.0139	0.0204	0.0174	0.0174	0.0010	0.0	0.0
F_1	0.0210	0.0024	0.0024	0.0339	0.0003	0.0003	0.0178	0.0149	0.0149	0.0011	0.0	0.0
F_2	0.0043	0.0038	0.0038	0.0163	0.0015	0.0015	0.0610	0.0565	0.0565	0.0011	0.0	0.0
F_3	0.0318	0.0110	0.0110	0.0149	0.0126	0.0126	0.0040	0.0036	0.0036	0.0021	0.0	0.0
F_4	0.0138	0.0075	0.0075	0.0096	0.0096	0.0096	0.0053	0.0050	0.0050	0.0021	0.0	0.0
F_5	0.0044	0.0011	0.0011	0.0025	0.0024	0.0024	0.0003	0.0000	0.0000			
F_6	0.0069	0.0008	0.0008	0.0023	0.0020	0.0020	0.1103	0.0290	0.0290			
F_7				0.0030	0.0030	0.0030	0.0960	0.0634	0.0634			
F_8				0.0067	0.0065	0.0065	0.0143	0.0075	0.0075			
F_9				0.0032	0.0029	0.0029	0.0122	0.0086	0.0086			
F_{10}				0.0082	0.0065	0.0065						

only two PCA components. Feature F_6 displayed a significantly higher reconstruction error than the different datasets. On the other hand, Feature F_5 showcased relatively low reconstruction errors across all datasets, suggesting that it was well represented using only two PCA components. We observed low reconstruction errors in all datasets for $k = 3$ and $k = 4$. It can be attributed to the fact that as the number of components increases, the dataset can be reconstructed more successfully.

Now, we can address the question of whether dimensionality reduction can be used as a substitute for GAN operations. We obtained consistent results when repeating the PCA and reconstruction process with the same data and parameters, including the reconstruction error. As a deterministic algorithm, PCA produces the same output when applied repeatedly. Conversely, GAN aims to generate new samples resembling the training data and capture its underlying distribution. GAN has the potential advantage of differentiating a dataset across executions by enhancing the quality and diversity of the generated samples.

Figure 4.20 visually compares the experiments conducted on the Wind Turbine dataset. PCA was performed for different values of k , specifically $k = 2, 3$, and 4 .

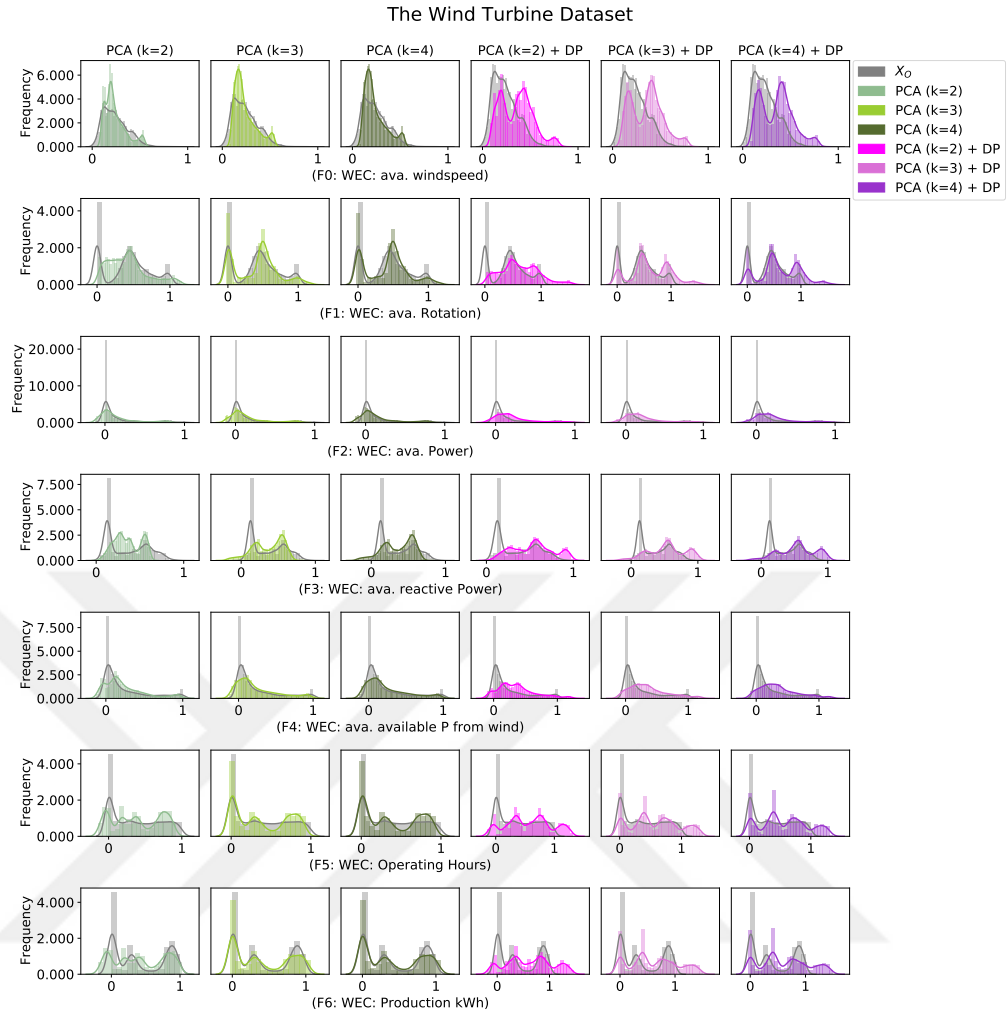


Figure 4.20 PDF Comparison: Original vs. PCA Variations

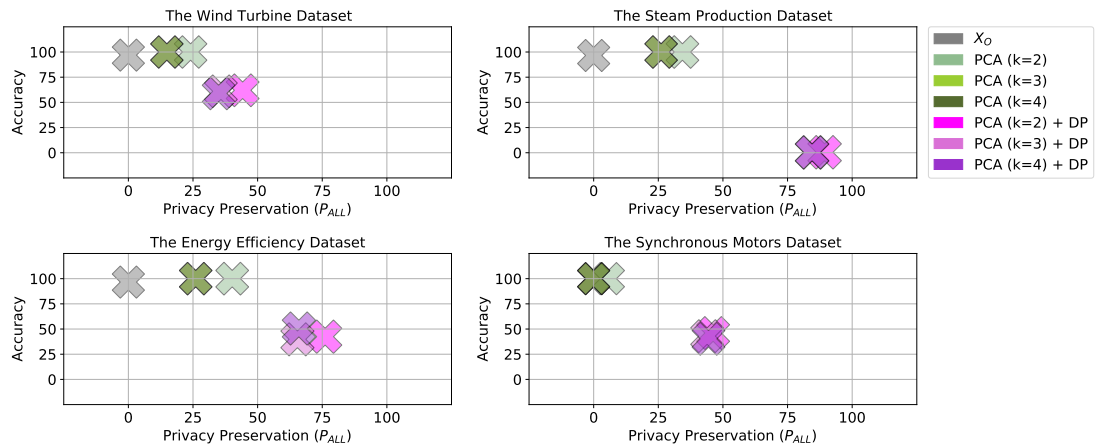


Figure 4.21 Accuracy vs. Privacy Analysis of PCA (Linear Regression)

Subsequently, DP functions were utilized to introduce noise. This visual comparison enables us to assess the degree to which the reconstructed dataset maintains a

compact shape similar to the original dataset. Furthermore, Table 4.8 provides additional insights into the similarity between the datasets.

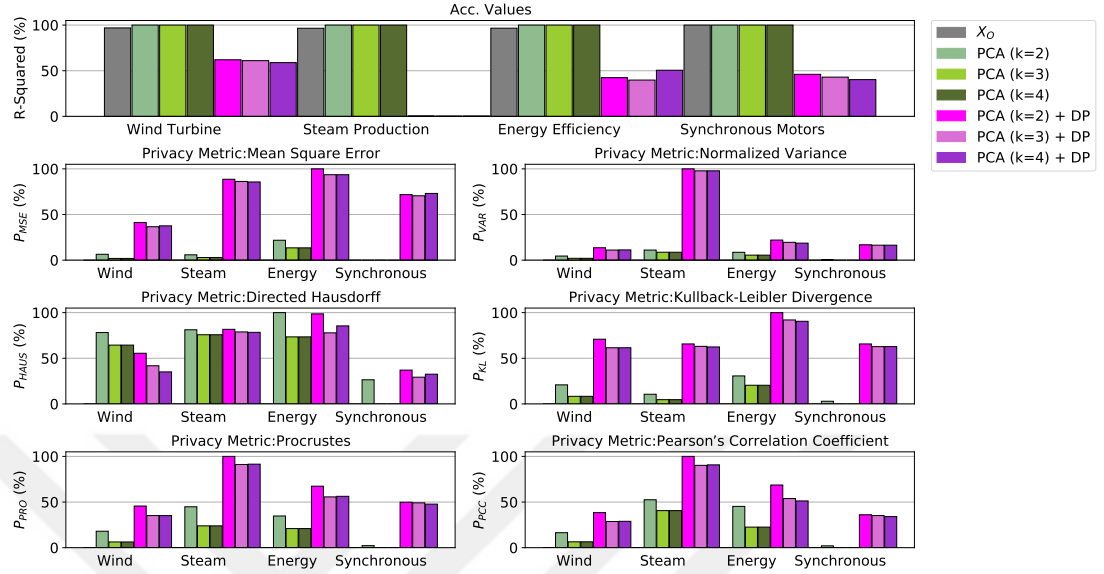


Figure 4.22 Comprehensive Bar Subplot Analysis (PCA)

Figure 4.21 and Figure 4.22 are employed to analyze the accuracy and privacy metrics further. The former presents a graph depicting accuracy versus privacy in percentage values, which was also used in other experiments. For $k = 4$, higher *R-Squared* values were obtained across all four datasets, indicating a closer fit between the original and reconstructed datasets. Moreover, minimal reconstruction errors were observed for $k = 4$. However, the introduction of noise through the DP function had a considerable impact on the *R-Squared* values. Turning our attention to the privacy metrics, the PCA methods achieved approximately 25 % preservation, while the DP-added methods reached around 50 %.

Figure 4.22 provides further evidence of the influence of DP addition on accuracy values and privacy preservation. The proposed method, denoted as X_{HGD} , outperforms the PCA outputs regarding accuracy and consistency. It reinforces the notion that adding DP affects the accuracy values while enhancing privacy preservation.

Finally, an essential aspect of the experimentation involved safeguarding hidden sensitive data. The case study enabled the identification of the most promising

proposed method, which exhibits superior performance compared to other methods.

4.9 Hidden Sensitive Data Experiments

In addition to evaluating the performance of the proposed privacy protection mechanism, our study includes experiments to assess its effectiveness in preserving privacy for hidden sensitive data. Hidden sensitive data refers to a subset of the dataset that can enable an attacker to deduce sensitive information when used in a suitable ML model. Without privacy preservation techniques, an attacker accessing a complete dataset can use ML algorithms to predict sensitive features.

For instance, in the Steam Production dataset, which aims to predict the amount of steam recovery (Y) based on other features, an attacker could potentially manipulate the input/output relationships and predict sensitive information such as the amount of natural gas (F_0) or the tank level (F_4) of raw materials. The proposed privacy protection mechanism offers a potential solution for safeguarding sensitive data from malicious prediction attempts. Our experiments demonstrate that the proposed method effectively preserves privacy even in the presence of hidden sensitive data.

Specifically, we tested the proposed algorithm on datasets containing hidden sensitive features and evaluated its accuracy and privacy metrics performance. The results reveal that the proposed method simultaneously achieves higher accuracy and privacy protection values, even with hidden sensitive data. These findings indicate that the proposed privacy protection mechanism effectively safeguards sensitive data from malicious attacks and preserves privacy in the presence of hidden sensitive features. As a result, the proposed approach can serve as a valuable tool for protecting data privacy in various applications involving sensitive information.

In the final experimental setup, we aimed to demonstrate the proposed approach's ability to preserve hidden sensitive data within a switched data structure. A key indicator of this preservation is the low prediction accuracy values obtained from modified ML models. If the proposed method offers superior privacy preservation

capabilities, as demonstrated in this study, Linear Regression models used for predicting hidden sensitive data should not be successful. To support this claim, we modified the input/output structures of the four datasets as follows:

- (i) In the Wind Turbine dataset, we switched the Linear Regression model from predicting WEC Production (F_6) to predicting WEC Average Power (F_2). We used features (F_0), (F_1), (F_3), (F_4), (F_5), and (F_6) as input features to predict (F_2).
- (ii) In the Steam Production dataset, we modified the Linear Regression model from predicting Steam Recovery (F_{10}) to predicting Natural Gas Amount (F_0). The input features for this new prediction task were (F_1), (F_2), (F_3), (F_4), (F_5), (F_6), (F_7), (F_8), (F_9), and (F_{10}).
- (iii) We changed the Linear Regression model in the Energy Efficiency dataset from predicting Heating Load (F_9) to Relative Compactness (F_0).
- (iv) In the Synchronous Motors dataset, we modified the Linear Regression model from predicting the Excitation Current of a Synchronous Machine (F_4) to predicting Load Current (F_0).

Through this experimental setup, we aimed to highlight the effectiveness of the proposed privacy preservation mechanism in protecting sensitive data from malicious attacks. Furthermore, we sought to demonstrate that the proposed method outperforms regular Linear Regression models regarding privacy preservation.

Figure 4.23 illustrates four distinct scenarios as part of the case study: the proposed method X_{HGD} , versions of the proposed methods X_{HG} and X_{HD} , and an alternative study involving PCA. Upon initial inspection, it is evident that the X_{HGD} and X_{HG} methods yield the lowest, and thus most protective, outputs.

Table 4.9 presents the accuracy of the proposed algorithm for hidden sensitive data. The experimental results for the listed datasets reveal that if an attacker can access the complete dataset without any privacy protection, they can easily predict the hidden sensitive data for malicious purposes. Specifically, the regular Linear

Table 4.9 Hidden Sensitive Data - Case Study *R-Squared* Values

Tests	Wind Turbine	Steam Production	Energy Efficiency	Synchronous Motors
Original	0.9343	0.6585	1.0000	0.6340
GAN	0.0051	0.0082	0.0228	0.0226
DP	0.6509	0.0585	0.5264	0.1041

X_{HGD} Proposed Method - GAN + DP				
$(n_s=1)$	0.0016	0.0093	0.0269	0.0276
$(n_s=2)$	0.0021	0.0098	0.0287	0.0596
$(n_s=3)$	0.0021	0.0048	0.0250	0.1376
$(n_s=4)$	0.0032	0.0068	0.0255	0.0228

X_{HG} Proposed Method - GAN Only				
$(n_s=1)$	0.0016	0.0065	0.0264	0.0230
$(n_s=2)$	0.0019	0.0064	0.0270	0.0365
$(n_s=3)$	0.0024	0.0058	0.0233	0.1482
$(n_s=4)$	0.0027	0.0078	0.0272	0.0241

X_{HD} Proposed Method - DP Only				
$(n_s=1)$	0.8538	0.2118	0.6890	0.3204
$(n_s=2)$	0.8379	0.2125	0.6861	0.3412
$(n_s=3)$	0.8208	0.2128	0.6875	0.2071
$(n_s=4)$	0.7967	0.2148	0.6859	0.2146

PCA Dimension Reduction - Reconstruction				
$(k=2)$	1.0000	1.0000	1.0000	1.0000
$(k=3)$	1.0000	1.0000	1.0000	0.6401
$(k=4)$	1.0000	1.0000	1.0000	0.6401

$PCA + DP$ Dimension Reduction - Reconstruction + DP				
$(k=2)$	0.7719	0.0011	0.4802	0.0996
$(k=3)$	0.7535	0.0124	0.5644	0.0759
$(k=4)$	0.7551	0.0276	0.5814	0.1318

Regression models (denoted as X_O) for the Wind Turbine, Steam Production, Energy Efficiency, and Synchronous Motors datasets achieved relatively high R-Squared scores of 0.9343, 0.6585, 1.0000, and 0.6340, respectively. These scores highlight the significant privacy risk associated with hidden sensitive data in a dataset.

In contrast, the proposed privacy preservation method (denoted as X_{HGD} and X_{HG}

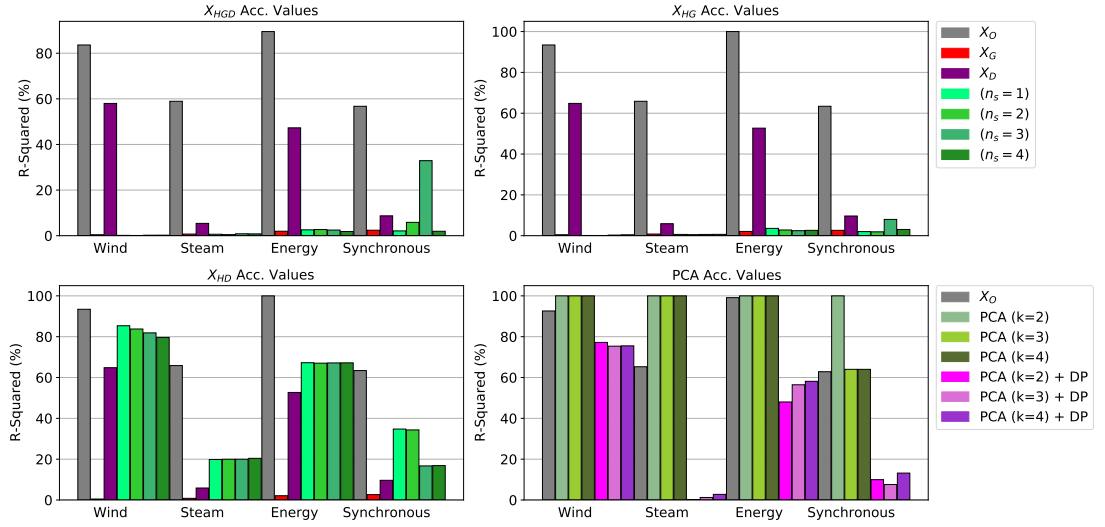


Figure 4.23 Comprehensive Bar Subplot Analysis (Case Study)

with all n_s values) yielded the lowest R-Squared scores in this scenario. The X_{HD} method can be eliminated based on its higher results. When comparing the X_{HGD} and X_{HG} methods, we observe slight differences, with X_{HGD} obtaining lower scores when $n_s = 3$. We need to consider their metric privacy performances to determine a better method. Table 4.9 demonstrates that X_{HGD} exhibits better metric results than X_{HG} .

Table 4.9 also includes the PCA performances for hidden sensitive protection. Neither PCA with values of $k = 2, 3$, and 4, nor DP-added PCA methods, achieved lower R-Squared scores.

These experiments further confirm the potential privacy risk associated with hidden sensitive data in a dataset, as the proposed method significantly limits the prediction performance for such data. It indicates that datasets under the proposed privacy preservation scheme cannot be effectively utilized for further Linear Regression analysis.

Preserving hidden sensitive data is a critical aspect of privacy preservation, and the ability of the proposed algorithm to protect such data demonstrates its potential for real-world applications where sensitive information must be kept confidential.

The experimental results presented in Table 4.9 validate the effectiveness of the proposed method in protecting sensitive data, emphasizing the need for robust privacy-preserving algorithms in data analysis.

4.10 Computational Cost Experiments

We conducted a test to monitor the CPU and RAM usage percentages of the proposed method, denoted as X_{HGD} . The test was performed in the following manner. Initially, we observed the system's idle state to measure the current CPU and RAM usage load of the operating system. The first plot in Figure 4.24 illustrates this idle mode, where the RAM usage remains stable at approximately 60 %, and the mean CPU usage is below 40 %. Next, we monitored the usage of Algorithm 4 for four different datasets: Wind Turbine, Steam Production, Energy Efficiency, and Synchronous Motors.

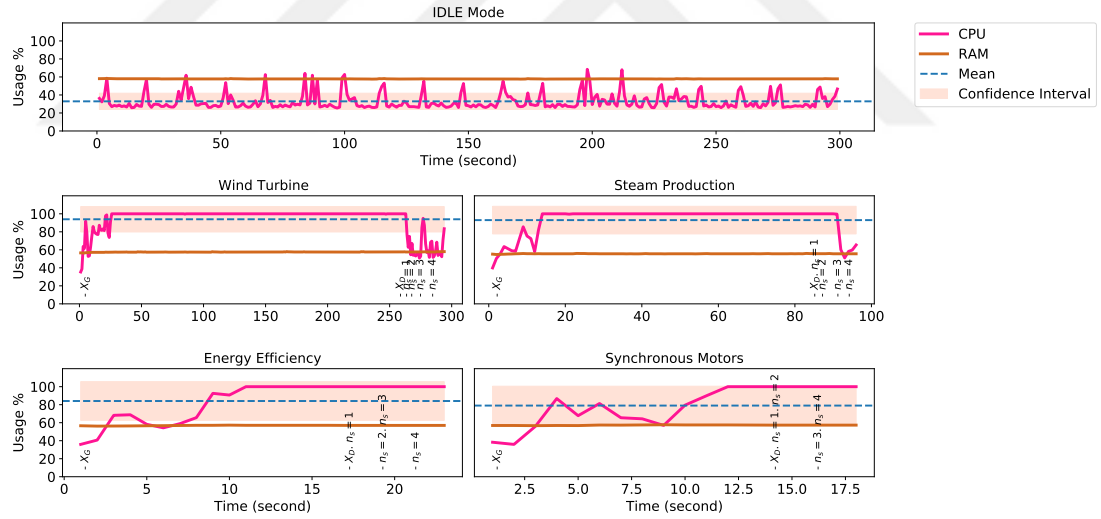


Figure 4.24 Resource Usage Analysis: CPU and RAM

Figure 4.24 displays the plots illustrating the computer resource usage for these four datasets. The plots depict the execution steps of the proposed method, with the GAN function triggering high CPU usage. It is important to note that the Wind Turbine dataset has 8,655 observations. The Steam Production has 2,556, Energy Efficiency has 768, and the Synchronous Motors dataset has 557 observations. By examining the plots, it can be observed that the execution duration of the proposed

method varies based on the number of observations in the datasets. The second plot represents the Wind Turbine dataset, where the CPU usage reaches around 90 % for the first 250 seconds. Similarly, the Steam Production, Energy Efficiency, and Synchronous Motors datasets exhibit comparable CPU usage for durations of 80, 20, and 15 seconds, respectively. Hence, as the number of observations decreases, the execution time of the proposed method also decreases.

In this context, the optimal usage of the proposed method can be understood by referring to Figure 3.7. A pipeline structure allows for the parallel execution of the proposed method while data collection is ongoing.



5. CONCLUSIONS

This thesis has presented a hybrid approach that effectively balances the benefits of AI and ML with preserving privacy. By addressing the challenge of maintaining privacy without compromising ML accuracy, this research contributes to AI and ML and provides practical solutions for privacy preservation. The experimental results have confirmed the efficacy of the proposed approach, demonstrating its ability to achieve satisfactory performance in privacy preservation and Linear Regression accuracy across diverse datasets. This empirical evidence reinforces the validity and effectiveness of the proposed method.

An essential contribution of this study is providing a robust solution to safeguarding sensitive information within data. In today's data-driven world, protecting hidden sensitive data from potential malicious attacks is of paramount importance. Through the proposed method, this study successfully preserves the privacy of sensitive information, ensuring its integrity and security. The experimental evaluations conducted on multiple datasets, including publicly available ones, validate the method's effectiveness in maintaining high accuracy while preserving privacy.

5.1 Scope and Limitations

This study is relevant to businesses in an era where the demand for private data analytics is escalating. The method described here provides a mechanism for enterprises, particularly larger ones, to share data without infringing on privacy standards. This approach is seamlessly compatible with data ecosystems such as IoT by employing a balanced pipeline that encompasses data collection, transmission, and analysis. However, its efficacy is contingent upon low correlations between sensitive and non-sensitive features. Challenges arising from high correlations might be mitigated through dimensionality reduction techniques. In the business realm, the proposed

hybrid approach can find utility in several areas:

- (i) **Data Sharing and Collaboration** Companies collaborating with third-party vendors or partners can benefit from this approach, ensuring data privacy while sharing essential insights.
- (ii) **Customer Data Protection** In sectors where customer data is paramount, such as finance or healthcare, the methodology can be instrumental in maintaining clients' trust while harnessing AI and ML's power.
- (iii) **Supply Chain Optimization** Data privacy becomes essential with the ever-growing need to make supply chains smarter and more efficient. This approach ensures data from various links in the chain remains confidential while drawing actionable insights.
- (iv) **Marketing Analytics** In the age of personalized marketing, ensuring customer privacy while tailoring marketing strategies can be achieved using the described method.

5.2 Future Directions

The versatility of the approach presented is evident in its potential as a preprocessing step, ensuring data protection without imposing excessive computational burdens. One direction worth investigating is identifying and categorizing sensitive features in datasets. While data proprietors bring to bear their business insights and understanding of market dynamics, Algorithm 3 offers a systematic, automated solution. An interesting avenue for future exploration would be aligning this methodology with data protection regulations such as the GDPR.

Potential research trajectories include broadening the application of this method, enhancing its scalability, and developing strategies for secure reversibility. Additionally, integrating other privacy-centric methodologies and conducting evaluations over a broader range of datasets could further propel advancements in the domain of privacy preservation for AI and ML systems.

BIBLIOGRAPHY

- Arachchige, Pathum Chamikara Mahawaga, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. 2020. “A Trustworthy Privacy Preserving Framework for Machine Learning in Industrial IoT Systems.” *IEEE Transactions on Industrial Informatics* 16 (9): 6092–6102.
- Arjovsky, Martin, Soumith Chintala, and Léon Bottou. 2017. “Wasserstein generative adversarial networks.” In *International conference on machine learning*, 214–223. PMLR.
- Atallah, Mike, Elisa Bertino, Ahmed Elmagarmid, Mohamed Ibrahim, and Vassilios Verykios. 1999. “Disclosure limitation of sensitive rules.” In *Proceedings 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX’99)(Cat. No. PR00453)*, 45–52. IEEE.
- Bakhshi, Zeinab, Ali Balador, and Jawad Mustafa. 2018. “Industrial IoT security threats and concerns by considering Cisco and Microsoft IoT reference models.” In *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 173–178. IEEE.
- Bayindir, R., and H. T. Kahraman. 2021. *Synchronous Machine dataset*. <https://archive.ics.uci.edu/ml/datasets/Synchronous+Machine+Data+Set>.
- Beaulieu-Jones, Brett K, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P Bhavnani, James Brian Byrd, and Casey S Greene. 2019. “Privacy-preserving generative deep neural networks support clinical data sharing.” *Circulation: Cardiovascular Quality and Outcomes* 12 (7): e005122.
- Bisio, Igor, Chiara Garibotto, Aldo Grattarola, Fabio Lavagetto, and Andrea Sciarone. 2018. “Exploiting context-aware capabilities over the internet of things for industry 4.0 applications.” *Ieee network* 32 (3): 101–107.
- Brock, Andrew, Jeff Donahue, and Karen Simonyan. 2018. “Large scale GAN training for high fidelity natural image synthesis.” *arXiv preprint arXiv:1809.11096*.

- Cheng, Peng, John F Roddick, Shu-Chuan Chu, and Chun-Wei Lin. 2016. “Privacy preservation through a greedy, distortion-based rule-hiding method.” *Applied Intelligence* 44:295–306.
- Choi, Edward, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. 2017. “Generating multi-label discrete patient records using generative adversarial networks.” In *Machine learning for healthcare conference*, 286–305. PMLR.
- Dalipi, Enri, Floris Van den Abeele, Isam Ishaq, Ingrid Moerman, and Jeroen Hoebeke. 2016. “EC-IoT: An easy configuration framework for constrained IoT devices.” In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, 159–164. IEEE.
- Dash, Manoranjan, Hua Liu, and Jun Yao. 1997. “Dimensionality reduction of unsupervised data.” In *Proceedings ninth ieee international conference on tools with artificial intelligence*, 532–539. IEEE.
- Dwork, Cynthia. 2006. “Differential privacy.” In *International colloquium on automata, languages, and programming*, 1–12. Springer.
- . 2008. “Differential privacy: A survey of results.” In *International conference on theory and applications of models of computation*, 1–19. Springer.
- Dwork, Cynthia, Aaron Roth, et al. 2014. “The algorithmic foundations of differential privacy.” *Foundations and Trends® in Theoretical Computer Science* 9 (3–4): 211–407.
- Fan, Xiaoyu, Guosai Wang, Kun Chen, Xu He, and Wei Xu. 2021. “PPCA: Privacy-preserving principal component analysis using secure multiparty computation (MPC).” *arXiv preprint arXiv:2105.07612*.
- Feng, Jun, Laurence T Yang, Ronghao Zhang, Weizhong Qiang, and Jinjun Chen. 2020. “Privacy preserving high-order bi-lanczos in cloud-fog computing for industrial applications.” *IEEE Transactions on Industrial Informatics* 18 (10): 7009–7018.

- Fernandes, Marta, Juan Manuel Corchado, and Goreti Marreiros. 2022. “Machine learning techniques applied to mechanical fault diagnosis and fault prognosis in the context of real industrial manufacturing use-cases: a systematic literature review.” *Applied Intelligence* 52 (12): 14246–14280.
- Fredrikson, Matthew, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. 2014. “Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing.” In *23rd USENIX security symposium (USENIX Security 14)*, 17–32.
- Galas, David J, Gregory Dewey, James Kunert-Graf, and Nikita A Sakhanenko. 2017. “Expansion of the Kullback-Leibler divergence, and a new class of information metrics.” *Axioms* 6 (2): 8.
- Gilad-Bachrach, Ran, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. “Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy.” In *International conference on machine learning*, 201–210. PMLR.
- Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. “Generative adversarial nets.” *Advances in neural information processing systems* 27.
- Gower, John C. 1975. “Generalized procrustes analysis.” *Psychometrika* 40:33–51.
- Han, Chunling, and Rui Xue. 2021. “Differentially private GANs by adding noise to discriminator’s loss.” *Computers & Security* 107:102322.
- Hindistan, Yavuz Selim. 2023. *GitHub repository*. <https://github.com/yavuzselimhindistan>.
- Hitaj, Briland, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. “Deep models under the GAN: information leakage from collaborative deep learning.” In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 603–618.

- Hu, Pengfei, Huansheng Ning, Tie Qiu, Houbing Song, Yanna Wang, and Xuanxia Yao. 2017. "Security and privacy preservation scheme of face identification and resolution framework using fog computing in internet of things." *IEEE Internet of Things Journal* 4 (5): 1143–1155.
- Huang, Chenxi, Sirui Chen, Yaqing Zhang, Wen Zhou, Joel JPC Rodrigues, and Victor Hugo C de Albuquerque. 2021. "A robust approach for privacy data protection: IoT security assurance using generative adversarial imitation learning." *IEEE Internet of Things Journal* 9 (18): 17089–17097.
- Husnoo, Muhammad Akbar, Adnan Anwar, Ripon K Chakraborty, Robin Doss, and Mike J Ryan. 2021. "Differential privacy for IoT-enabled critical infrastructure: A comprehensive survey." *IEEE Access* 9:153276–153304.
- Islam, Mir Riyanul, Mobyen Uddin Ahmed, Shaibal Barua, and Shahina Begum. 2022. "A Systematic Review of Explainable Artificial Intelligence in Terms of Different Application Domains and Tasks." *Applied Sciences* 12 (3): 1353. <https://doi.org/10.3390/app12031353>.
- Jeong, Hyuk-Jin, Hyeon-Jae Lee, and Soo-Mook Moon. 2017. "Work-in-progress: Cloud-based machine learning for IoT devices with better privacy." In *2017 International Conference on Embedded Software (EMSOFT)*, 1–2. IEEE.
- Jiang, Bin, Jianqiang Li, Guanghui Yue, and Houbing Song. 2021. "Differential privacy for industrial internet of things: Opportunities, applications, and challenges." *IEEE Internet of Things Journal* 8 (13): 10430–10451.
- Kairouz, Peter, Sewoong Oh, and Pramod Viswanath. 2016. "Extremal mechanisms for local differential privacy." *The Journal of Machine Learning Research* 17 (1): 492–542.
- Kasiviswanathan, Shiva Prasad, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. "What can we learn privately?" *SIAM Journal on Computing* 40 (3): 793–826.

- Khosravy, Mahdi, Kazuaki Nakamura, Naoko Nitta, Nilanjan Dey, Rubén González Crespo, Enrique Herrera-Viedma, and Noboru Babaguchi. 2022. "Social IoT Approach to Cyber Defense of a Deep-Learning-Based Recognition System in Front of Media Clones Generated by Model Inversion Attack." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 53 (5): 2694–2704.
- Langarica, Saúl, Christian Ruffelmacher, and Felipe Núñez. 2019. "An industrial internet application for real-time fault diagnosis in industrial motors." *IEEE Transactions on Automation Science and Engineering* 17 (1): 284–295.
- Li, Wei, Pengqiu Meng, Yi Hong, and Xiaohui Cui. 2020. "Using deep learning to preserve data confidentiality." *Applied Intelligence* 50:341–353.
- Liang, Fan, Wei Yu, Xing Liu, David Griffith, and Nada Golmie. 2020. "Toward edge-based deep learning in industrial Internet of Things." *IEEE Internet of Things Journal* 7 (5): 4329–4341.
- Liu, Yi, Jialiang Peng, JQ James, and Yi Wu. 2019. "PPGAN: Privacy-preserving generative adversarial network." In *2019 IEEE 25Th international conference on parallel and distributed systems (ICPADS)*, 985–989. IEEE.
- Long, Guodong, Tao Shen, Yue Tan, Leah Gerrard, Allison Clarke, and Jing Jiang. 2021. "Federated learning for privacy-preserving open innovation future on digital health." In *Humanity Driven AI: Productivity, Well-being, Sustainability and Partnership*, 113–133. Springer.
- Majeed, Abdul, and Sungchang Lee. 2020. "Attribute susceptibility and entropy based data anonymization to improve users community privacy and utility in publishing data." *Applied Intelligence* 50 (8): 2555–2574.
- Malekzadeh, Mohammad, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. 2020. "Privacy and utility preserving sensor-data transformations." *Pervasive and Mobile Computing* 63:101132.

- McMahan, Brendan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agueray Arcas. 2017. "Communication-efficient learning of deep networks from decentralized data." In *Artificial intelligence and statistics*, 1273–1282. PMLR.
- Micro, Trend. 2021. *A Constant State of Flux: Trend Micro 2020 Annual Cybersecurity Report*. <https://www.trendmicro.com/>.
- Narayanan, Arvind, and Vitaly Shmatikov. 2006. "How to break anonymity of the netflix prize dataset." *arXiv preprint cs/0610105*.
- Oliveira, Stanley RM, and Osmar R Zaiane. 2010. "Privacy preserving clustering by data transformation." *Journal of Information and Data Management* 1 (1): 37–37.
- Ping, Haoyue, Julia Stoyanovich, and Bill Howe. 2017. "Datasyntesizer: Privacy-preserving synthetic datasets." In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 1–5.
- Piramuthu, Selwyn. 2005. "Feature selection for reduction of tabular knowledge-based systems." *Information Technology and Management* 6:351–362.
- Qi, Saiyu, Youshui Lu, Wei Wei, and Xiaofeng Chen. 2020. "Efficient data access control with fine-grained data protection in cloud-assisted IIoT." *IEEE Internet of Things Journal* 8 (4): 2886–2899.
- Radford, Alec, Luke Metz, and Soumith Chintala. 2015. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434*.
- Samie, Farzad, Lars Bauer, and Jörg Henkel. 2019. "From cloud down to things: An overview of machine learning in internet of things." *IEEE Internet of Things Journal* 6 (3): 4921–4934.
- Sang, Yingpeng, Hong Shen, and Hui Tian. 2011. "Effective reconstruction of data perturbed by random projections." *IEEE Transactions on Computers* 61 (1): 101–117.

- Sari, Sercan, Onur Demir, and Gurhan Kucuk. 2019. "FairSDP: Fair and secure dynamic cache partitioning." In *2019 4th International Conference on Computer Science and Engineering (UBMK)*, 469–474. IEEE.
- Shahid, Mustafizur R, Gregory Blanc, Houda Jmila, Zonghua Zhang, and Hervé Debar. 2020. "Generative deep learning for Internet of Things network traffic generation." In *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*, 70–79. IEEE.
- Sharma, Parjanay, Siddhant Jain, Shashank Gupta, and Vinay Chamola. 2021. "Role of machine learning and deep learning in securing 5G-driven industrial IoT applications." *Ad Hoc Networks* 123:102685.
- Shokri, Reza, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. "Membership inference attacks against machine learning models." In *2017 IEEE symposium on security and privacy (SP)*, 3–18. IEEE.
- Soontronchai, Waraporn. 2019. *IIoT Data of Wind Turbine*. <https://www.kaggle.com/datasets/wasuratme96/iiot-data-of-wind-turbine>.
- Stout, William MS, and Vincent E Urias. 2016. "Challenges to securing the Internet of Things." In *2016 IEEE International Carnahan Conference on Security Technology (ICCST)*, 1–8. IEEE.
- Sweeney, Latanya. 1997. "Maintaining patient confidentiality when sharing medical data requires a symbiotic relationship between technology and policy." *Retrieved March 9:2011*.
- Tahsien, Syeda Manjia, Hadis Karimipour, and Petros Spachos. 2020. "Machine learning based solutions for security of Internet of Things (IoT): A survey." *Journal of Network and Computer Applications* 161:102630.
- Tsanas, Athanasios, and Angeliki Xifara. 2012. *Energy efficiency*. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C51307>.

- Usman, Muhammad, Alireza Jolfaei, and Mian Ahmad Jan. 2020. "RaSEC: an intelligent framework for reliable and secure multilevel edge computing in industrial environments." *IEEE Transactions on Industry Applications* 56 (4): 4543–4551.
- Wagner, Isabel, and David Eckhoff. 2018. "Technical privacy metrics: a systematic survey." *ACM Computing Surveys (CSUR)* 51 (3): 1–38.
- Wang, Sen, and J Morris Chang. 2018. "Differentially private principal component analysis over horizontally partitioned data." In *2018 IEEE Conference on Dependable and Secure Computing (DSC)*, 1–8. IEEE.
- Xu, Chugui, Ju Ren, Deyu Zhang, Yaoxue Zhang, Zhan Qin, and Kui Ren. 2019. "GANobfuscator: Mitigating information leakage under GAN via differential privacy." *IEEE Transactions on Information Forensics and Security* 14 (9): 2358–2371.
- Xu, Lei, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. "Modeling tabular data using conditional gan." *Advances in neural information processing systems* 32.
- Xu, Lei, and Kalyan Veeramachaneni. 2018. "Synthesizing tabular data using generative adversarial networks." *arXiv preprint arXiv:1811.11264*.
- Xu, Xiaolong, Shucun Fu, Lianyong Qi, Xuyun Zhang, Qingxiang Liu, Qiang He, and Shancang Li. 2018. "An IoT-oriented data placement method with privacy preservation in cloud environment." *Journal of Network and Computer Applications* 124:148–157.
- Xu, Xiaolong, Ruichao Mo, Xiaochun Yin, Mohanmmad R Khosravi, Fahimeh Aghaei, Victor Chang, and Guangshun Li. 2020. "PDM: Privacy-aware deployment of machine-learning applications for industrial cyber-physical cloud systems." *IEEE Transactions on Industrial Informatics* 17 (8): 5819–5828.
- Yang, Xinyu, Teng Wang, Xuebin Ren, and Wei Yu. 2017. "Survey on improving data utility in differentially private sequential data publishing." *IEEE Transactions on Big Data* 7 (4): 729–749.

Zeng, Qingtian, Zhenzhen Lv, Chao Li, Yongkui Shi, Zedong Lin, Cong Liu, and Ge Song. 2023. “FedProLs: federated learning for IoT perception data prediction.” *Applied Intelligence* 53 (3): 3563–3575.



APPENDIX A: Extended Version in Chapter 4.5 - Experimental Exploration of Privacy Preservation Methods

Table A.1 Experiments in Chapter 4.5 - Accuracy Values

Logistic Regression Accuracy												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
Log.Reg. Accuracy	1.0	0.87500	0.86540	0.8365	0.87500	0.84620	0.85580	0.88460	0.86540	0.8558	0.85580	01
Log.Reg. Accuracy	1.0	0.67920	0.88680	0.8491	0.90570	0.89620	0.88680	0.89620	0.86790	0.8868	0.88680	02
Log.Reg. Accuracy	1.0	0.71700	0.81130	0.8113	0.80190	0.82080	0.80190	0.81130	0.81130	0.8113	0.81130	03
Log.Reg. Accuracy	1.0	0.84260	0.85190	0.8241	0.85190	0.85190	0.85190	0.83330	0.84260	0.8611	0.84260	04
Log.Reg. Accuracy	1.0	0.85320	0.87160	0.8349	0.87160	0.86240	0.86240	0.90830	0.86240	0.8716	0.87160	05
Log.Reg. Accuracy	1.0	0.90570	0.90570	0.8302	0.90570	0.91510	0.90570	0.88680	0.88680	0.9057	0.90570	06
Log.Reg. Accuracy	1.0	0.66980	0.84910	0.7642	0.86790	0.84910	0.86790	0.83960	0.84910	0.8396	0.84910	07
Log.Reg. Accuracy	1.0	0.62390	0.90830	0.6881	0.85320	0.90830	0.92660	0.83490	0.90830	0.9174	0.88990	08
Log.Reg. Accuracy	1.0	0.81980	0.84680	0.8468	0.81980	0.85590	0.84680	0.83780	0.83780	0.8468	0.85590	09
Log.Reg. Accuracy	1.0	0.90990	0.90090	0.8378	0.89190	0.91890	0.90990	0.81080	0.90090	0.9099	0.90990	10
Log.Reg. Accuracy	1.0	0.78961	0.86978	0.8123	0.86446	0.87248	0.87157	0.85436	0.86325	0.8706	0.86786	Mean

Support Vector Machine Accuracy												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
SVM Accuracy	1.0	0.46230	0.84910	0.83020	0.84910	0.84910	0.84910	0.8585	0.84910	0.84910	0.84910	01
SVM Accuracy	1.0	0.78500	0.88790	0.87850	0.87850	0.87850	0.87850	0.8785	0.87850	0.87850	0.87850	02
SVM Accuracy	1.0	0.76640	0.81310	0.80370	0.80370	0.80370	0.80370	0.8037	0.80370	0.80370	0.80370	03
SVM Accuracy	1.0	0.65740	0.84260	0.84260	0.84260	0.84260	0.84260	0.8426	0.84260	0.84260	0.84260	04
SVM Accuracy	1.0	0.84820	0.84820	0.84820	0.84820	0.84820	0.84820	0.8482	0.84820	0.84820	0.84820	05
SVM Accuracy	1.0	0.93200	0.94170	0.93200	0.93200	0.93200	0.93200	0.9320	0.93200	0.93200	0.93200	06
SVM Accuracy	1.0	0.85850	0.84910	0.84910	0.84910	0.84910	0.84910	0.8491	0.84910	0.84910	0.84910	07
SVM Accuracy	1.0	0.92520	0.92520	0.92520	0.92520	0.92520	0.92520	0.9252	0.92520	0.92520	0.92520	08
SVM Accuracy	1.0	0.42340	0.84680	0.84680	0.84680	0.84680	0.84680	0.8468	0.84680	0.84680	0.84680	09
SVM Accuracy	1.0	0.35450	0.90910	0.91820	0.91820	0.91820	0.91820	0.9364	0.91820	0.91820	0.91820	10
SVM Accuracy	1.0	0.70129	0.87128	0.86745	0.86934	0.86934	0.86934	0.8721	0.86934	0.86934	0.86934	Mean

Random Forest Accuracy												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
RF Accuracy	1.00000	0.44860	0.85980	0.73830	0.8131	0.75700	0.72900	0.8411	0.85050	0.80370	0.62620	01
RF Accuracy	1.00000	0.51460	0.96120	0.84470	0.8447	0.91260	0.91260	0.8738	0.83500	0.90290	0.88350	02
RF Accuracy	1.00000	0.55450	0.92080	0.83170	0.7624	0.87130	0.82180	0.8020	0.89110	0.72280	0.85150	03
RF Accuracy	1.00000	0.81080	0.90990	0.67570	0.7207	0.74770	0.81080	0.8018	0.77480	0.73870	0.78380	04
RF Accuracy	1.00000	0.45000	0.99000	0.90000	0.9200	0.84000	0.86000	0.9400	0.88000	0.80000	0.96000	05
RF Accuracy	0.99050	0.91430	1.00000	0.88570	0.7714	0.88570	0.77140	0.8667	0.87620	0.84760	0.80000	06
RF Accuracy	1.00000	0.68270	0.88460	0.82690	0.9038	0.80770	0.84620	0.8558	0.81730	0.85580	0.83650	07
RF Accuracy	1.00000	0.53700	0.97220	0.81480	0.7778	0.84260	0.89810	0.9352	0.90740	0.85190	0.65740	08
RF Accuracy	1.00000	0.64550	0.87270	0.77270	0.6182	0.70000	0.69090	0.8636	0.63640	0.80000	0.89090	09
RF Accuracy	1.00000	0.48650	0.95500	0.82880	0.8559	0.84680	0.80180	0.9640	0.87390	0.68470	0.86490	10
RF Accuracy	0.99905	0.60445	0.93262	0.81193	0.7988	0.82114	0.81426	0.8744	0.83426	0.80081	0.81547	Mean

Table A.2 Experiments in Chapter 4.5 - P_{MSE} , P_{VAR} , and P_{HAUS} Metrics

Privacy Metric : Mean Square Error P_{MSE}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{MSE}	0.0	1.0000	0.00100	0.8246	0.58050	0.4156	0.20790	0.77500	0.62350	0.38370	0.1948	01
P_{MSE}	0.0	1.0000	0.00490	0.8094	0.59780	0.3667	0.19630	0.78660	0.57310	0.37960	0.2133	02
P_{MSE}	0.0	0.6802	0.33230	0.6863	0.73590	0.6830	0.75110	0.80560	0.82300	0.99090	1.0000	03
P_{MSE}	0.0	0.9709	0.26920	0.8189	0.81660	0.8050	0.71620	0.89110	0.87140	1.00000	0.8914	04
P_{MSE}	0.0	0.8699	0.34630	0.8310	0.80560	0.7921	0.74610	0.88780	0.93920	0.97620	1.0000	05
P_{MSE}	0.0	0.7365	0.32960	0.7565	0.68080	0.6543	0.75700	0.80850	0.84030	0.99290	1.0000	06
P_{MSE}	0.0	0.5050	0.34550	0.5231	0.61280	0.6105	0.65510	0.59690	0.69820	0.81810	1.0000	07
P_{MSE}	0.0	1.0000	0.11120	0.8699	0.68880	0.5989	0.38380	0.89880	0.76850	0.59220	0.4723	08
P_{MSE}	0.0	0.6605	0.37390	0.6786	0.73120	0.6563	0.75490	0.73910	0.88580	0.95470	1.0000	09
P_{MSE}	0.0	1.0000	0.00200	0.7527	0.58760	0.3996	0.21500	0.77740	0.61810	0.41090	0.1712	10
P_{MSE}	0.0	0.8423	0.21159	0.7551	0.68376	0.5982	0.53834	0.79668	0.76411	0.74992	0.6943	Mean

Privacy Metric : Normalized Variance P_{VAR}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{VAR}	0.0	0.74320	0.00010	0.95870	0.92920	0.84660	0.46360	0.95990	1.00000	0.7376	0.43550	01
P_{VAR}	0.0	0.84890	0.00050	1.00000	0.89260	0.66340	0.41030	0.96820	0.90020	0.7101	0.46920	02
P_{VAR}	0.0	0.59810	0.19250	0.60430	0.64130	0.68480	0.53400	0.84900	1.00000	0.9270	0.79100	03
P_{VAR}	0.0	1.00000	0.01320	0.84130	0.82400	0.64370	0.31520	0.97490	0.76000	0.7030	0.38510	04
P_{VAR}	0.0	0.84750	0.07520	0.94960	0.78060	0.75720	0.58100	0.97450	1.00000	0.8942	0.63890	05
P_{VAR}	0.0	0.10430	0.17840	0.70360	0.97220	1.00000	0.72570	0.58760	0.90850	0.9398	0.76620	06
P_{VAR}	0.0	0.23350	0.05930	0.75270	0.98040	0.95320	0.62110	0.78070	1.00000	0.9530	0.66390	07
P_{VAR}	0.0	0.53020	0.00640	0.87210	0.95550	1.00000	0.47950	0.86290	0.98370	0.8355	0.52840	08
P_{VAR}	0.0	0.24430	0.33450	0.38380	0.48740	0.49610	0.45460	0.63390	0.89820	1.0000	0.89100	09
P_{VAR}	0.0	0.90450	0.00350	0.89010	0.96640	0.82520	0.47850	0.88910	1.00000	0.8828	0.38900	10
P_{VAR}	0.0	0.60545	0.08636	0.79562	0.84296	0.78702	0.50635	0.84807	0.94506	0.8583	0.59582	Mean

Privacy Metric : Directed Hausdorff P_{HAUS}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{HAUS}	0.0	1.0	0.1345	0.3298	0.1006	0.0861	0.0683	0.2593	0.1924	0.183	0.1345	01
P_{HAUS}	0.0	1.0	0.3607	0.5606	0.5692	0.2623	0.2726	0.4541	0.3702	0.3823	0.3607	02
P_{HAUS}	0.0	1.0	0.1567	0.5266	0.4556	0.4556	0.1149	0.1817	0.2332	0.1567	0.1567	03
P_{HAUS}	0.0	1.0	0.2415	0.1995	0.4714	0.1995	0.0774	0.8257	0.376	0.2415	0.2415	04
P_{HAUS}	0.0	1.0	0.1749	0.4754	0.2967	0.1256	0.0758	0.2353	0.1844	0.1761	0.1761	05
P_{HAUS}	0.0	1.0	0.113	0.2073	0.3007	0.1273	0.0768	0.2144	0.1932	0.1235	0.113	06
P_{HAUS}	0.0	1.0	0.1279	0.3711	0.0946	0.3113	0.0625	0.2801	0.1666	0.1766	0.1279	07
P_{HAUS}	0.0	1.0	0.1727	0.4833	0.1812	0.1427	0.0559	0.3763	0.3582	0.2294	0.1922	08
P_{HAUS}	0.0	1.0	0.1910	0.2403	0.1048	0.4803	0.1476	0.3415	0.2107	0.1910	0.2201	09
P_{HAUS}	0.0	1.0	0.4271	0.7102	0.7102	0.1913	0.6630	0.7102	0.4970	0.4271	0.4271	10
P_{HAUS}	0.0	1.0	0.2100	0.41041	0.3285	0.2382	0.16148	0.38786	0.27819	0.22872	0.21498	Mean

Table A.3 Experiments in Chapter 4.5 - P_{KL} , P_{PRO} , and P_{PCC} Metrics

Privacy Metric : Kullback-Leibler Divergence P_{KL}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{KL}	0.0	0.0	0.01730	1.00000	0.77030	0.51270	0.26320	0.97570	0.75780	0.48760	0.28200	01
P_{KL}	0.0	1.0	0.03150	0.87940	0.65950	0.44300	0.25380	0.85860	0.67460	0.46260	0.26790	02
P_{KL}	0.0	1.0	0.26620	0.93040	0.80910	0.64420	0.57580	0.95150	0.90890	0.88870	0.72240	03
P_{KL}	0.0	1.0	0.27740	0.87740	0.76200	0.68810	0.60730	0.98240	0.83130	0.80980	0.78550	04
P_{KL}	0.0	0.0	0.11200	1.00000	0.74890	0.54670	0.39240	0.93680	0.77340	0.57370	0.39840	05
P_{KL}	0.0	1.0	0.15990	0.87280	0.73990	0.59150	0.44000	0.87860	0.77260	0.67580	0.53290	06
P_{KL}	0.0	1.0	0.33580	0.88610	0.73580	0.74300	0.60590	0.94910	0.91520	0.83340	0.88230	07
P_{KL}	0.0	0.0	0.17970	0.95590	0.80330	0.68320	0.43210	1.00000	0.86010	0.71350	0.58520	08
P_{KL}	0.0	0.0	0.30640	0.92890	0.89340	0.68160	0.56470	1.00000	0.96120	0.88940	0.77910	09
P_{KL}	0.0	0.0	0.03030	1.00000	0.83440	0.57740	0.33740	0.98550	0.82810	0.62570	0.32040	10
P_{KL}	0.0	0.5	0.17165	0.93309	0.77566	0.61114	0.44726	0.95182	0.82832	0.69602	0.55561	Mean

Privacy Metric : Procrustes P_{PRO}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{PRO}	0.0	0.99680	0.32070	1.00000	0.99990	1.00000	0.99930	0.99990	0.99610	0.99650	0.99950	01
P_{PRO}	0.0	0.99750	0.33410	0.99890	0.99840	0.99750	0.99990	0.99900	0.99960	0.99910	1.00000	02
P_{PRO}	0.0	0.99850	0.30780	0.99840	0.99740	0.99880	1.00000	0.99530	0.99750	0.99950	0.99870	03
P_{PRO}	0.0	0.99970	0.29800	0.99750	0.99680	0.99930	0.99940	1.00000	0.99830	0.99860	0.99930	04
P_{PRO}	0.0	0.99850	0.32030	0.99820	0.99950	0.99700	1.00000	0.99950	0.99800	0.99900	0.99940	05
P_{PRO}	0.0	1.00000	0.31900	0.99680	0.98810	0.98970	0.99800	0.99940	0.99840	0.99710	0.99860	06
P_{PRO}	0.0	0.99910	0.31870	0.99910	0.99940	1.00000	0.99990	0.99930	0.99670	0.99900	0.99830	07
P_{PRO}	0.0	1.00000	0.31220	0.99980	0.99910	0.99690	0.99690	0.99970	0.99840	0.99640	0.99980	08
P_{PRO}	0.0	0.99620	0.31350	0.99900	0.99990	0.99760	1.00000	0.99690	0.99800	0.99900	0.99880	09
P_{PRO}	0.0	0.99820	0.30850	0.99600	0.99630	1.00000	0.99950	0.99880	0.99920	0.99970	0.99990	10
P_{PRO}	0.0	0.99845	0.31528	0.99837	0.99748	0.99768	0.99929	0.99878	0.99802	0.99839	0.99923	Mean

Privacy Metric : Pearson's Correlation Coefficient P_{PCC}												
Test	Original	V-GAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{PCC}	0.0	0.99760	0.32420	0.99260	1.00000	0.98640	0.98010	0.99380	0.96780	0.96400	0.99630	01
P_{PCC}	0.0	0.98530	0.30680	1.00000	0.96090	0.98660	0.98530	0.96940	0.94980	0.98290	0.97920	02
P_{PCC}	0.0	0.99140	0.31740	0.97920	1.00000	0.96330	0.98900	0.98280	0.96090	0.99260	0.96940	03
P_{PCC}	0.0	0.99530	0.29900	0.98570	0.96520	0.96280	0.96640	0.93280	0.95800	1.00000	0.94600	04
P_{PCC}	0.0	0.96610	0.30430	0.97340	1.00000	0.98790	0.98670	0.96610	0.93340	0.96000	0.99160	05
P_{PCC}	0.0	0.89640	0.30820	0.95980	0.96460	0.95000	0.98900	0.99630	0.99140	1.00000	0.97440	06
P_{PCC}	0.0	0.95060	0.29730	0.95780	1.00000	0.97230	0.93010	0.93260	0.96020	0.96750	0.97230	07
P_{PCC}	0.0	0.95100	0.31250	0.97300	1.00000	0.98900	0.96570	0.96570	0.98280	0.97670	0.98410	08
P_{PCC}	0.0	0.94870	0.30320	0.97800	0.97920	0.95350	1.00000	0.97310	0.95600	0.98410	0.98170	09
P_{PCC}	0.0	0.97900	0.31980	0.97280	0.98520	0.98650	0.95920	0.95310	1.00000	0.97780	0.97280	10
P_{PCC}	0.0	0.96614	0.30927	0.97723	0.98551	0.97383	0.97515	0.96657	0.96603	0.98056	0.97678	Mean

APPENDIX B: Extended Version in Chapter 4.6 - Vanilla Gan versus CTGAN Experiments

Table B.1 Experiments in Chapter 4.6 - Accuracy Values

Logistic Regression Accuracy												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
Log.Reg. Accuracy	1.00000	0.89320	0.90290	0.91260	0.90290	0.89320	0.93200	0.90290	0.90290	0.90290	0.91260	01
Log.Reg. Accuracy	1.00000	0.88070	0.87160	0.88070	0.87160	0.86240	0.87160	0.87160	0.87160	0.88070	0.87160	02
Log.Reg. Accuracy	1.00000	0.83810	0.86670	0.79050	0.82860	0.82860	0.85710	0.82860	0.82860	0.82860	0.83810	03
Log.Reg. Accuracy	0.97980	0.98990	0.98990	0.98990	0.98990	0.97980	1.00000	1.00000	0.98990	0.98990	1.00000	04
Log.Reg. Accuracy	1.00000	0.90820	0.90820	0.90820	0.89800	0.90820	0.90820	0.90820	0.90820	0.90820	0.90820	05
Log.Reg. Accuracy	1.00000	0.82910	0.82910	0.79490	0.82910	0.82910	0.82910	0.82910	0.82910	0.82910	0.82910	06
Log.Reg. Accuracy	1.00000	0.81650	0.82570	0.81650	0.81650	0.81650	0.81650	0.81650	0.81650	0.81650	0.81650	07
Log.Reg. Accuracy	1.00000	0.85190	0.85190	0.85190	0.85190	0.85190	0.85190	0.85190	0.85190	0.84260	0.85190	08
Log.Reg. Accuracy	1.00000	0.94120	0.95100	0.94120	0.94120	0.93140	0.94120	0.94120	0.94120	0.94120	0.94120	09
Log.Reg. Accuracy	1.00000	0.91590	0.87850	0.87850	0.87850	0.89720	0.87850	0.87850	0.87850	0.88790	0.87850	10
Log.Reg. Accuracy	0.99798	0.88647	0.88755	0.87649	0.88082	0.87983	0.88861	0.88285	0.88184	0.88276	0.88477	Mean

Support Vector Machine Accuracy												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
SVM Accuracy	1.00000	0.89420	0.90380	0.89420	0.89420	0.89420	0.89420	0.89420	0.89420	0.89420	0.89420	01
SVM Accuracy	1.00000	0.88790	0.88790	0.88790	0.88790	0.88790	0.88790	0.88790	0.88790	0.88790	0.88790	02
SVM Accuracy	1.00000	0.80560	0.82410	0.80560	0.80560	0.80560	0.80560	0.80560	0.80560	0.80560	0.80560	03
SVM Accuracy	0.98980	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	04
SVM Accuracy	1.00000	0.84760	0.86670	0.84760	0.84760	0.84760	0.84760	0.84760	0.84760	0.84760	0.84760	05
SVM Accuracy	1.00000	0.84350	0.85220	0.84350	0.84350	0.84350	0.84350	0.84350	0.84350	0.84350	0.84350	06
SVM Accuracy	1.00000	0.83180	0.84110	0.83180	0.83180	0.83180	0.83180	0.83180	0.83180	0.83180	0.83180	07
SVM Accuracy	1.00000	0.84400	0.84400	0.84400	0.84400	0.84400	0.84400	0.84400	0.84400	0.84400	0.84400	08
SVM Accuracy	1.00000	0.96970	0.97980	0.96970	0.96970	0.96970	0.96970	0.96970	0.96970	0.96970	0.96970	09
SVM Accuracy	1.00000	0.88680	0.88680	0.88680	0.88680	0.88680	0.88680	0.88680	0.88680	0.88680	0.88680	10
SVM Accuracy	0.99898	0.88111	0.88864	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111	0.88111	Mean

Random Forest Accuracy												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
RF Accuracy	1.00000	0.94060	0.9109	0.92080	0.86140	0.90100	0.87130	0.91090	0.89110	0.91090	0.98020	01
RF Accuracy	1.00000	0.91180	0.9510	0.72550	0.89220	0.80390	0.92160	0.78430	0.97060	0.98040	0.86270	02
RF Accuracy	1.00000	0.86270	0.9118	0.84310	0.77450	0.82350	0.84310	0.83330	0.90200	0.83330	0.75490	03
RF Accuracy	0.98060	0.96120	1.0000	0.89320	0.93200	0.89320	0.90290	0.93200	0.96120	0.82520	0.96120	04
RF Accuracy	1.00000	0.84110	0.9252	0.81310	0.76640	0.85980	0.79440	0.77570	0.72900	0.84110	0.82240	05
RF Accuracy	1.00000	0.83050	0.8729	0.76270	0.68640	0.69490	0.69490	0.75420	0.80510	0.85590	0.82200	06
RF Accuracy	1.00000	0.87250	0.8824	0.87250	0.80390	0.85290	0.88240	0.88240	0.88240	0.88240	0.86270	07
RF Accuracy	1.00000	0.82080	0.8962	0.88680	0.84910	0.84910	0.80190	0.89620	0.83020	0.86790	0.86790	08
RF Accuracy	0.94060	0.70300	1.0000	0.90100	0.96040	0.88120	0.88120	0.95050	0.91090	0.95050	0.94060	09
RF Accuracy	1.00000	0.83930	0.9196	0.85710	0.79460	0.83040	0.76790	0.84820	0.83930	0.85710	0.83930	10
RF Accuracy	0.99212	0.85835	0.9270	0.84758	0.83209	0.83899	0.83616	0.85677	0.87218	0.88047	0.87139	Mean

Table B.2 Experiments in Chapter 4.6 - P_{MSE} , P_{VAR} , and P_{HAUS} Metrics

Privacy Metric : Mean Square Error P_{MSE}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{MSE}	0.0	1.00000	0.10760	0.82950	0.67250	0.52800	0.37710	0.8314	0.80130	0.49960	0.41170	01
P_{MSE}	0.0	0.80520	0.33710	0.78430	0.76660	0.71690	0.71390	0.8351	0.95240	0.91960	1.00000	02
P_{MSE}	0.0	0.86620	0.31160	0.79670	0.79260	0.79660	0.70800	0.9825	0.93810	1.00000	0.99330	03
P_{MSE}	0.0	0.87640	0.29950	0.84250	0.83530	0.71790	0.71770	0.8951	0.86000	1.00000	0.90090	04
P_{MSE}	0.0	0.86310	0.29680	0.86080	0.76870	0.75590	0.71610	0.9337	0.90320	1.00000	0.86880	05
P_{MSE}	0.0	0.99330	0.25970	1.00000	0.90860	0.78370	0.68410	0.9755	0.83120	0.93100	0.81630	06
P_{MSE}	0.0	1.00000	0.29240	0.92910	0.84600	0.75130	0.75710	0.9599	0.92480	0.99250	0.94020	07
P_{MSE}	0.0	0.97840	0.29170	0.90420	0.83980	0.83420	0.72270	0.9208	0.99460	1.00000	0.97900	08
P_{MSE}	0.0	1.00000	0.13650	0.86220	0.70810	0.58210	0.46440	0.9010	0.79040	0.68570	0.52670	09
P_{MSE}	0.0	0.93900	0.29940	0.88820	0.82410	0.75020	0.75920	1.0000	0.91510	0.94410	0.97030	10
P_{MSE}	0.0	0.93216	0.26323	0.86975	0.79623	0.72168	0.66203	0.9235	0.89111	0.89725	0.84072	Mean

Privacy Metric : Normalized Variance P_{VAR}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{VAR}	0.0	0.7745	0.41400	0.75360	0.70140	0.65010	0.55870	0.92350	0.99270	1.00000	0.95870	01
P_{VAR}	0.0	0.7156	0.46560	0.68710	0.65320	0.63960	0.57960	0.82020	0.90940	1.00000	0.98340	02
P_{VAR}	0.0	0.7470	0.43610	0.66860	0.65890	0.59320	0.54760	0.84560	0.88090	1.00000	0.97170	03
P_{VAR}	0.0	0.8174	0.41820	0.74200	0.72510	0.65540	0.55580	0.94210	0.96380	1.00000	0.99190	04
P_{VAR}	0.0	0.7141	0.45580	0.67940	0.61440	0.57050	0.54760	0.83470	0.96130	1.00000	0.97470	05
P_{VAR}	0.0	0.7690	0.45550	0.67100	0.67720	0.60800	0.56230	0.85110	0.90170	0.98510	1.00000	06
P_{VAR}	0.0	0.7533	0.39810	0.73680	0.65550	0.59820	0.53140	0.88430	0.95110	1.00000	0.92400	07
P_{VAR}	0.0	0.6969	0.41440	0.67190	0.61520	0.57300	0.51570	0.82270	0.86940	1.00000	0.98130	08
P_{VAR}	0.0	0.7585	0.37160	0.70980	0.65360	0.62160	0.53800	0.92980	1.00000	0.99450	0.93330	09
P_{VAR}	0.0	0.8147	0.43830	0.74860	0.68200	0.65580	0.59680	0.90560	0.96080	1.00000	0.98880	10
P_{VAR}	0.0	0.7561	0.42676	0.70688	0.66365	0.61654	0.55335	0.87596	0.93911	0.99796	0.97078	Mean

Privacy Metric : Directed Hausdorff P_{HAUS}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{HAUS}	0.0	0.4624	1.00000	0.40990	0.4450	0.60050	0.61110	0.46240	0.41790	0.59550	0.79650	01
P_{HAUS}	0.0	0.6300	0.41610	0.63000	0.6788	0.83150	0.26600	0.83150	0.63000	1.00000	1.00000	02
P_{HAUS}	0.0	0.7855	1.00000	0.83260	0.8673	0.83260	0.89740	0.78550	0.83260	0.74410	0.74410	03
P_{HAUS}	0.0	0.7270	1.00000	0.47630	0.7270	0.72700	0.25000	0.87620	0.72700	0.72700	1.00000	04
P_{HAUS}	0.0	0.5589	1.00000	0.75770	0.7217	0.65980	0.92910	0.61400	0.55890	0.73210	0.64620	05
P_{HAUS}	0.0	0.6388	1.00000	0.70700	0.6853	0.66190	0.68530	0.63880	0.69140	0.72460	1.00000	06
P_{HAUS}	0.0	0.7499	0.57490	0.74990	1.0000	0.37150	0.72500	0.74990	0.74990	0.57490	0.57490	07
P_{HAUS}	0.0	0.7829	1.00000	0.62210	0.9185	0.80410	0.28140	0.97230	0.84550	0.91850	0.98650	08
P_{HAUS}	0.0	0.8198	1.00000	0.78760	0.6407	0.83290	0.41970	0.68170	0.79650	0.83290	0.88510	09
P_{HAUS}	0.0	0.7168	0.79250	0.71680	0.3147	1.00000	0.27040	0.71680	0.79250	0.79250	0.79250	10
P_{HAUS}	0.0	0.6872	0.87835	0.66899	0.6999	0.73218	0.53354	0.73291	0.70422	0.76421	0.84258	Mean

Table B.3 Experiments in Chapter 4.6 - P_{KL} , P_{PRO} , and P_{PCC} Metrics

Privacy Metric : Kullback-Leibler Divergence P_{KL}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{KL}	0.0	1.00000	0.27680	0.97400	0.82780	0.7331	0.55880	0.97530	0.96060	0.78540	0.71020	01
P_{KL}	0.0	0.97740	0.39790	0.87990	0.71730	0.7171	0.66610	0.99240	0.92430	1.00000	0.89770	02
P_{KL}	0.0	0.96400	0.39650	0.91490	0.80800	0.7502	0.62690	1.00000	0.94560	0.94860	0.93100	03
P_{KL}	0.0	1.00000	0.34740	0.91980	0.74480	0.7305	0.57870	0.89840	0.82050	0.91610	0.91650	04
P_{KL}	0.0	0.95380	0.36400	0.84950	0.75660	0.7248	0.63150	1.00000	0.88710	0.95700	0.87560	05
P_{KL}	0.0	1.00000	0.32400	0.85950	0.89280	0.7215	0.57450	0.93210	0.90610	0.79420	0.80240	06
P_{KL}	0.0	1.00000	0.31910	0.91250	0.81820	0.7635	0.64780	0.87610	0.80430	0.84530	0.80270	07
P_{KL}	0.0	0.97690	0.33090	0.90900	0.72720	0.6972	0.53310	1.00000	0.82780	0.85840	0.85220	08
P_{KL}	0.0	0.95850	0.26620	0.93010	0.72240	0.6994	0.58280	1.00000	0.84530	0.83740	0.71370	09
P_{KL}	0.0	0.99580	0.32580	0.82200	0.77140	0.6327	0.65490	1.00000	0.80560	0.81190	0.78520	10
P_{KL}	0.0	0.98264	0.33486	0.89712	0.77865	0.7170	0.60551	0.96743	0.87272	0.87543	0.82872	Mean

Privacy Metric : Procrustes P_{PRO}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{PRO}	0.0	0.99570	0.32530	0.99960	0.9987	0.99860	1.00000	0.99840	1.00000	0.98880	0.9962	01
P_{PRO}	0.0	0.99990	0.32330	0.99950	0.9965	1.00000	0.99110	0.99940	0.99970	1.00000	0.9979	02
P_{PRO}	0.0	1.00000	0.32560	0.99770	0.9994	0.99540	0.99950	0.99540	0.99970	0.99810	0.9919	03
P_{PRO}	0.0	1.00000	0.30680	0.99900	0.9945	0.99980	0.99990	0.99850	0.99990	0.99870	0.9950	04
P_{PRO}	0.0	0.99840	0.31250	0.99750	1.0000	0.99610	0.99930	0.99950	0.99640	0.99890	0.9956	05
P_{PRO}	0.0	0.98570	0.31510	0.99680	0.9982	1.00000	0.99690	0.99460	0.99540	0.99900	0.9987	06
P_{PRO}	0.0	0.99950	0.30440	0.99920	1.0000	0.99980	0.99930	0.99800	0.99970	0.99980	0.9980	07
P_{PRO}	0.0	0.99730	0.29690	0.99210	0.9995	0.99450	0.99790	0.99870	0.99390	0.99610	1.0000	08
P_{PRO}	0.0	1.00000	0.30550	0.99820	0.9979	0.99950	0.99600	0.99890	0.99800	0.99810	0.9949	09
P_{PRO}	0.0	0.99920	0.31910	0.99930	0.9993	0.99990	1.00000	0.99790	0.99850	0.99810	0.9978	10
P_{PRO}	0.0	0.99757	0.31345	0.99789	0.9984	0.99836	0.99799	0.99793	0.99812	0.99756	0.9966	Mean

Privacy Metric : Pearson's Correlation Coefficient P_{PCC}												
Test	Original	CTGAN	DP	20% Ori-GAN	40% Ori-GAN	60% Ori-GAN	80% Ori-GAN	20% DP-GAN	40% DP-GAN	60% DP-GAN	80% DP-GAN	Test Nb.
P_{PCC}	0.0	0.96500	0.31290	0.97830	0.98070	0.94690	0.97100	1.00000	0.96860	0.95290	0.98190	01
P_{PCC}	0.0	0.95960	0.31420	0.97310	0.97920	0.97680	0.96220	0.99380	1.00000	0.96090	0.97680	02
P_{PCC}	0.0	0.95030	0.30880	0.94920	0.97450	0.97580	0.95030	0.97940	0.96370	0.97940	1.00000	03
P_{PCC}	0.0	0.98790	0.32160	0.98790	1.00000	0.97330	0.93790	0.96230	0.97080	0.97560	0.96470	04
P_{PCC}	0.0	0.97940	0.32490	0.96000	0.95890	1.00000	0.95760	0.97940	0.94550	0.96970	0.95390	05
P_{PCC}	0.0	0.97910	0.31200	1.00000	0.97910	0.96810	0.96310	0.96310	0.96920	0.99750	0.96920	06
P_{PCC}	0.0	0.98520	0.32020	0.97900	1.00000	0.97780	0.95550	0.97660	0.97530	0.97410	0.99390	07
P_{PCC}	0.0	0.98520	0.32020	0.97900	1.00000	0.97780	0.95550	0.97660	0.97530	0.97410	0.99390	07
P_{PCC}	0.0	0.96380	0.29950	0.96500	0.93600	0.99030	1.00000	0.96620	0.97580	0.98680	0.93240	08
P_{PCC}	0.0	1.00000	0.29760	0.96590	0.94630	0.97560	0.98650	0.98650	0.96340	0.97320	0.98780	09
P_{PCC}	0.0	0.97900	0.31560	0.99390	0.98140	0.98760	0.96530	1.00000	0.97030	0.99260	0.98270	10
P_{PCC}	0.0	0.97493	0.31273	0.97523	0.97361	0.97722	0.96494	0.98073	0.97026	0.97627	0.97433	Mean

APPENDIX C: Extended Version in Chapter 4.7 - Proposed Method Experiments

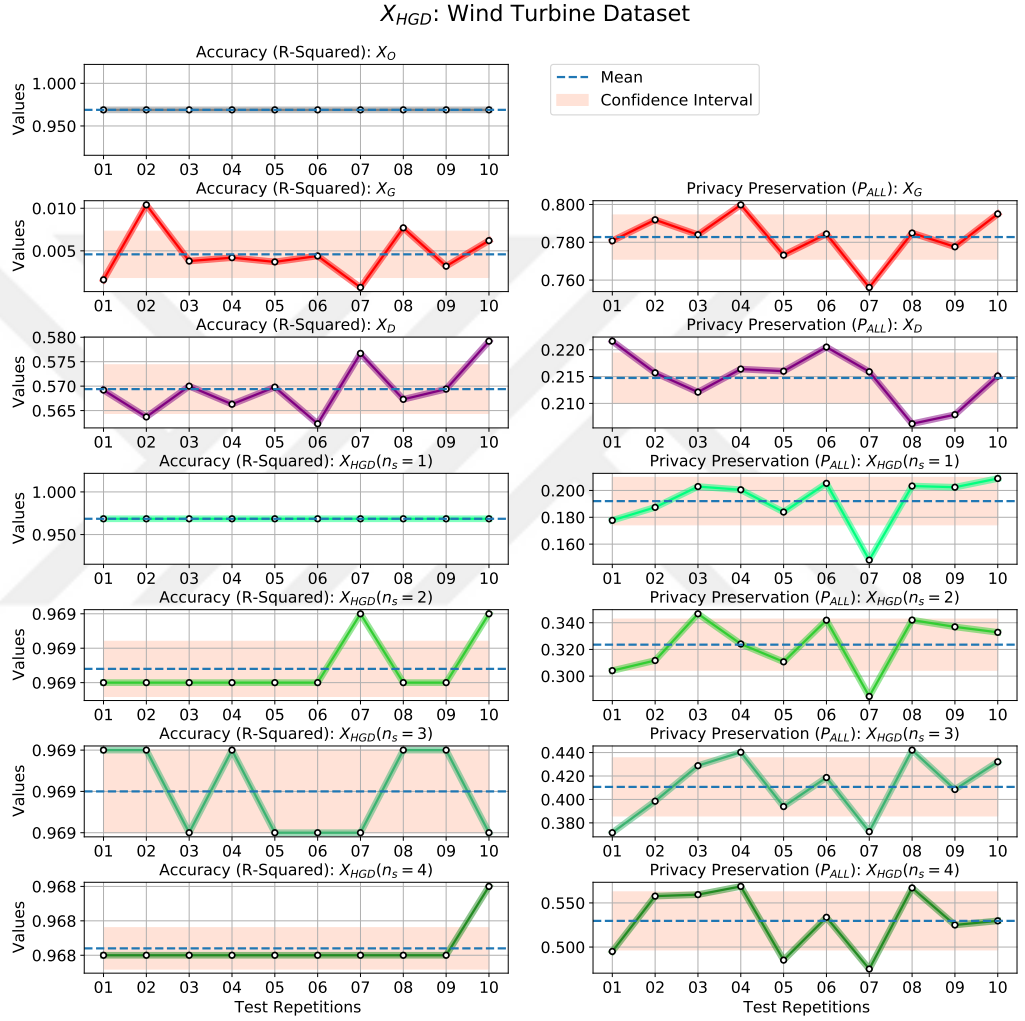


Figure C.1 Variations Among 10 Tests (X_{HGD}) - Wind Turbine Dataset

Table C.1 Experiments in Chapter 4.7 - GAN + DP - Wind Turbine

Proposed Method GAN + DP on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
$R\text{-Squared}$	0.9689	0.00160	0.56920	0.9685	0.96850	0.96860	0.96820	01
$R\text{-Squared}$	0.9689	0.01040	0.56370	0.9685	0.96850	0.96860	0.96820	02
$R\text{-Squared}$	0.9689	0.00380	0.57000	0.9685	0.96850	0.96850	0.96820	03
$R\text{-Squared}$	0.9689	0.00420	0.56630	0.9685	0.96850	0.96860	0.96820	04
$R\text{-Squared}$	0.9689	0.00370	0.56980	0.9685	0.96850	0.96850	0.96820	05
$R\text{-Squared}$	0.9689	0.00440	0.56230	0.9685	0.96850	0.96850	0.96820	06
$R\text{-Squared}$	0.9689	0.00070	0.57670	0.9685	0.96860	0.96850	0.96820	07
$R\text{-Squared}$	0.9689	0.00770	0.56730	0.9685	0.96850	0.96860	0.96820	08
$R\text{-Squared}$	0.9689	0.00320	0.56940	0.9685	0.96850	0.96860	0.96820	09
$R\text{-Squared}$	0.9689	0.00620	0.57920	0.9685	0.96860	0.96850	0.96830	10
$R\text{-Squared}$	0.9689	0.00459	0.56939	0.9685	0.96852	0.96855	0.96821	Mean
Proposed Method GAN + DP on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.67160	0.20260	0.09590	0.20260	0.24510	0.43690	01
P_{MSE}	0.0	0.82570	0.22910	0.14470	0.26540	0.34350	0.63880	02
P_{MSE}	0.0	0.63450	0.16860	0.15960	0.28390	0.35060	0.53230	03
P_{MSE}	0.0	0.69490	0.18120	0.14180	0.24450	0.34250	0.53320	04
P_{MSE}	0.0	0.61870	0.19120	0.11060	0.23630	0.28170	0.41230	05
P_{MSE}	0.0	0.71960	0.20880	0.15380	0.30770	0.36270	0.54960	06
P_{MSE}	0.0	0.51680	0.15970	0.06740	0.15120	0.19320	0.30680	07
P_{MSE}	0.0	0.74130	0.20040	0.16250	0.29200	0.38980	0.60080	08
P_{MSE}	0.0	0.68830	0.19820	0.17190	0.30260	0.33910	0.49530	09
P_{MSE}	0.0	0.75400	0.21550	0.13060	0.24370	0.31190	0.53290	10
P_{MSE}	0.0	0.68654	0.19553	0.13388	0.25299	0.31601	0.50389	Mean
Proposed Method GAN + DP on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10250	0.09460	0.20140	0.3465	0.45300	01
P_{VAR}	0.0	0.63820	0.10240	0.10960	0.21350	0.4028	0.52390	02
P_{VAR}	0.0	0.66980	0.10250	0.13720	0.25730	0.4649	0.57960	03
P_{VAR}	0.0	0.70830	0.10230	0.12430	0.23490	0.4889	0.60160	04
P_{VAR}	0.0	0.58470	0.10250	0.10470	0.22200	0.3663	0.45870	05
P_{VAR}	0.0	0.60790	0.10230	0.11890	0.23600	0.3953	0.49730	06
P_{VAR}	0.0	0.59730	0.10230	0.09010	0.19490	0.3538	0.45070	07
P_{VAR}	0.0	0.66230	0.10240	0.12450	0.23560	0.4693	0.57360	08
P_{VAR}	0.0	0.62930	0.10240	0.13230	0.24400	0.3911	0.49270	09
P_{VAR}	0.0	0.65040	0.10250	0.10830	0.21580	0.4161	0.52980	10
P_{VAR}	0.0	0.63441	0.10241	0.11445	0.22554	0.4095	0.51609	Mean
Proposed Method GAN + DP on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.43200	0.55740	0.51720	0.40270	01
P_{HAUS}	0.0	0.36360	0.32540	0.37060	0.47260	0.44330	0.39580	02
P_{HAUS}	0.0	0.43430	0.36050	0.32550	0.50500	0.43140	0.36930	03
P_{HAUS}	0.0	0.41860	0.38010	0.38830	0.49510	0.52160	0.44880	04
P_{HAUS}	0.0	0.50480	0.36700	0.39920	0.46510	0.52190	0.33910	05
P_{HAUS}	0.0	0.46330	0.37260	0.42520	0.52900	0.51730	0.38000	06
P_{HAUS}	0.0	0.52240	0.40400	0.30100	0.50890	0.56960	0.43780	07
P_{HAUS}	0.0	0.37270	0.30360	0.38590	0.56220	0.52200	0.42220	08
P_{HAUS}	0.0	0.37460	0.31100	0.33110	0.47390	0.47100	0.38630	09
P_{HAUS}	0.0	0.46910	0.35760	0.52850	0.63550	0.67290	0.38370	10
P_{HAUS}	0.0	0.44229	0.35855	0.38873	0.52047	0.51882	0.39657	Mean

Table C.2 Experiments in Chapter 4.7 - GAN + DP - Wind Turbine

Proposed Method GAN + DP on Wind Turbine Dataset: P_{KL} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.92060	0.22220	0.13620	0.2267	0.28130	0.45810	01
P_{KL}	0.0	0.93250	0.23440	0.16080	0.2617	0.33090	0.53280	02
P_{KL}	0.0	0.96520	0.23880	0.20540	0.3201	0.40130	0.59730	03
P_{KL}	0.0	0.97910	0.23640	0.17980	0.2838	0.37380	0.56290	04
P_{KL}	0.0	0.94430	0.23490	0.16310	0.2732	0.33090	0.48890	05
P_{KL}	0.0	0.92260	0.23900	0.17800	0.2905	0.35110	0.53410	06
P_{KL}	0.0	0.90030	0.22960	0.13310	0.2251	0.28010	0.44080	07
P_{KL}	0.0	0.94940	0.23030	0.17970	0.2765	0.36420	0.54840	08
P_{KL}	0.0	0.97910	0.23640	0.19870	0.3026	0.35960	0.53440	09
P_{KL}	0.0	0.89660	0.21540	0.14950	0.2418	0.31200	0.48340	10
P_{KL}	0.0	0.93897	0.23174	0.16843	0.2702	0.33852	0.51811	Mean

Proposed Method GAN + DP on Wind Turbine Dataset: P_{KL} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.1653	0.35130	0.41130	0.65080	01
P_{PRO}	0.0	1.0	0.25540	0.1983	0.37570	0.44750	0.68660	02
P_{PRO}	0.0	1.0	0.25470	0.2441	0.42940	0.49410	0.70410	03
P_{PRO}	0.0	1.0	0.25300	0.2278	0.40360	0.49130	0.69760	04
P_{PRO}	0.0	1.0	0.25500	0.1846	0.38100	0.43480	0.64770	05
P_{PRO}	0.0	1.0	0.25460	0.2146	0.40650	0.46020	0.67520	06
P_{PRO}	0.0	1.0	0.25440	0.1547	0.34260	0.40790	0.64160	07
P_{PRO}	0.0	1.0	0.25450	0.2269	0.40550	0.48520	0.69490	08
P_{PRO}	0.0	1.0	0.25460	0.2372	0.41310	0.46630	0.67830	09
P_{PRO}	0.0	1.0	0.25430	0.1955	0.37520	0.45090	0.67690	10
P_{PRO}	0.0	1.0	0.25444	0.2049	0.38839	0.45495	0.67537	Mean

Proposed Method GAN + DP on Wind Turbine Dataset: P_{PCC} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.99700	0.14460	0.14190	0.28500	0.42830	0.56870	01
P_{PCC}	0.0	0.99160	0.14780	0.14060	0.28140	0.42340	0.56830	02
P_{PCC}	0.0	1.00000	0.14730	0.14590	0.28470	0.43060	0.57370	03
P_{PCC}	0.0	0.99770	0.14560	0.14130	0.28260	0.42390	0.56810	04
P_{PCC}	0.0	0.98680	0.14540	0.14120	0.28660	0.42780	0.56460	05
P_{PCC}	0.0	0.99300	0.14560	0.14150	0.28150	0.42570	0.56570	06
P_{PCC}	0.0	1.00000	0.14530	0.14250	0.28630	0.43020	0.57260	07
P_{PCC}	0.0	0.98390	0.14580	0.14010	0.28030	0.42190	0.56200	08
P_{PCC}	0.0	0.99440	0.14470	0.14330	0.28510	0.42410	0.56320	09
P_{PCC}	0.0	1.00000	0.14530	0.14100	0.28490	0.42880	0.57120	10
P_{PCC}	0.0	0.99444	0.14574	0.14193	0.28384	0.42647	0.56781	Mean

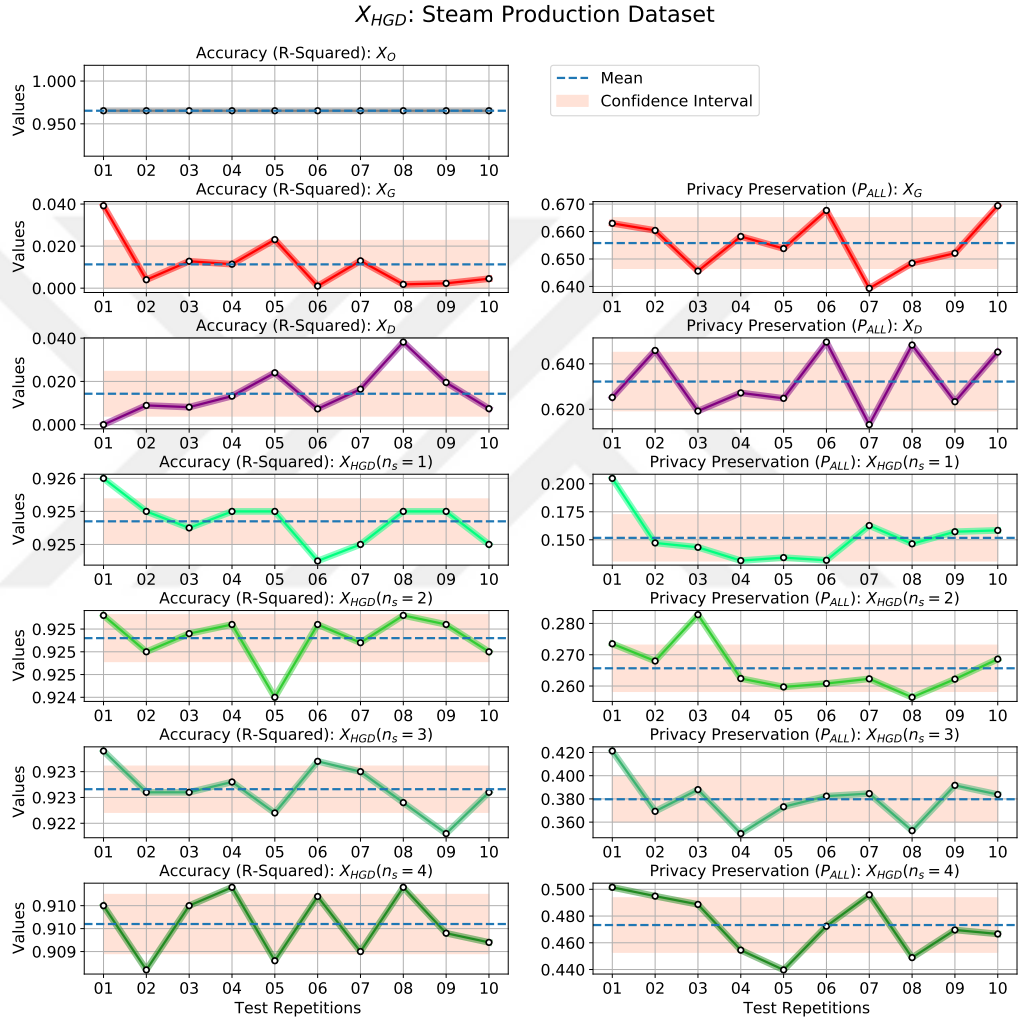


Figure C.2 Variations Among 10 Tests (X_{HGD}) - Steam Production Dataset

Table C.3 Experiments in Chapter 4.7 - GAN + DP - Steam Production

Proposed Method GAN + DP on Steam Production Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	0.9653	0.03920	0.0000	0.92560	0.92540	0.92320	0.9100	1
R -Squared	0.9653	0.00400	0.0089	0.92540	0.92500	0.92280	0.9093	2
R -Squared	0.9653	0.01280	0.0081	0.92530	0.92520	0.92280	0.9100	3
R -Squared	0.9653	0.01140	0.0132	0.92540	0.92530	0.92290	0.9102	4
R -Squared	0.9653	0.02310	0.0240	0.92540	0.92450	0.92260	0.9094	5
R -Squared	0.9653	0.00100	0.0073	0.92510	0.92530	0.92310	0.9101	6
R -Squared	0.9653	0.01300	0.0164	0.92520	0.92510	0.92300	0.9095	7
R -Squared	0.9653	0.00180	0.0382	0.92540	0.92540	0.92270	0.9102	8
R -Squared	0.9653	0.00230	0.0195	0.92540	0.92530	0.92240	0.9097	9
R -Squared	0.9653	0.00450	0.0074	0.92520	0.92500	0.92280	0.9096	10
R -Squared	0.9653	0.01131	0.0143	0.92534	0.92515	0.92283	0.9098	Mean
Proposed Method GAN + DP on Steam Production Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.2679	0.48840	0.05410	0.10860	0.19660	0.27460	1
P_{MSE}	0.0	0.2992	0.54090	0.03840	0.09200	0.17640	0.27220	2
P_{MSE}	0.0	0.2118	0.40090	0.02830	0.07330	0.17800	0.27110	3
P_{MSE}	0.0	0.2366	0.43600	0.02190	0.07160	0.13080	0.22090	4
P_{MSE}	0.0	0.2623	0.44510	0.02570	0.06090	0.13130	0.20490	5
P_{MSE}	0.0	0.2831	0.49650	0.02460	0.07350	0.13650	0.20960	6
P_{MSE}	0.0	0.2032	0.37970	0.04260	0.08030	0.14440	0.24590	7
P_{MSE}	0.0	0.2481	0.48560	0.03110	0.07920	0.13790	0.26520	8
P_{MSE}	0.0	0.2719	0.46460	0.04960	0.08940	0.15910	0.22880	9
P_{MSE}	0.0	0.2849	0.50160	0.03250	0.09010	0.15870	0.23100	10
P_{MSE}	0.0	0.2569	0.46393	0.03488	0.08189	0.15497	0.24242	Mean
Proposed Method GAN + DP on Steam Production Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.13340	0.36450	0.45410	0.52620	1
P_{VAR}	0.0	0.65740	1.0	0.10050	0.34760	0.41190	0.49010	2
P_{VAR}	0.0	0.63460	1.0	0.10340	0.36220	0.43130	0.51610	3
P_{VAR}	0.0	0.66250	1.0	0.08920	0.34250	0.40780	0.48100	4
P_{VAR}	0.0	0.68900	1.0	0.10060	0.34150	0.40380	0.48600	5
P_{VAR}	0.0	0.64830	1.0	0.08970	0.33180	0.39300	0.47400	6
P_{VAR}	0.0	0.66160	1.0	0.13390	0.37620	0.44890	0.52700	7
P_{VAR}	0.0	0.63930	1.0	0.10420	0.33910	0.39760	0.48230	8
P_{VAR}	0.0	0.67320	1.0	0.12680	0.37910	0.43070	0.50530	9
P_{VAR}	0.0	0.64810	1.0	0.10200	0.35760	0.41780	0.48580	10
P_{VAR}	0.0	0.65727	1.0	0.10837	0.35421	0.41969	0.49738	Mean
Proposed Method GAN + DP on Steam Production Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.6884	0.5214	0.78310	0.80120	1
P_{HAUS}	0.0	0.70670	0.92660	0.4516	0.5483	0.57990	0.81280	2
P_{HAUS}	0.0	0.73660	0.89930	0.4287	0.6313	0.62150	0.70810	3
P_{HAUS}	0.0	0.73450	0.92090	0.4044	0.5495	0.51510	0.63310	4
P_{HAUS}	0.0	0.64090	0.88860	0.3878	0.5416	0.65530	0.56000	5
P_{HAUS}	0.0	0.75500	0.98880	0.4048	0.5596	0.73130	0.77330	6
P_{HAUS}	0.0	0.67750	0.90110	0.4458	0.4583	0.63330	0.78630	7
P_{HAUS}	0.0	0.71480	1.00000	0.4477	0.5107	0.54520	0.55390	8
P_{HAUS}	0.0	0.63940	0.87210	0.4229	0.4469	0.68850	0.68800	9
P_{HAUS}	0.0	0.78590	0.98030	0.5219	0.5414	0.67720	0.70750	10
P_{HAUS}	0.0	0.71475	0.92473	0.4604	0.5309	0.64304	0.70242	Mean

Table C.4 Experiments in Chapter 4.7 - GAN + DP - Steam Production

Proposed Method GAN + DP on Steam Production Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.03750	0.06350	0.13560	0.19050	1
P_{KL}	0.0	0.30100	0.25430	0.02640	0.05560	0.11990	0.18420	2
P_{KL}	0.0	0.29800	0.26450	0.02740	0.05790	0.14880	0.21270	3
P_{KL}	0.0	0.31620	0.26190	0.02420	0.05430	0.12350	0.18680	4
P_{KL}	0.0	0.34160	0.26850	0.02750	0.05810	0.12520	0.18620	5
P_{KL}	0.0	0.33080	0.26230	0.02390	0.05080	0.11640	0.17570	6
P_{KL}	0.0	0.29520	0.25720	0.03790	0.06720	0.13440	0.19870	7
P_{KL}	0.0	0.29030	0.25810	0.02650	0.05290	0.11740	0.18480	8
P_{KL}	0.0	0.33710	0.25890	0.03610	0.06620	0.13540	0.18970	9
P_{KL}	0.0	0.30220	0.23780	0.02520	0.05320	0.11480	0.17080	10
P_{KL}	0.0	0.31095	0.25746	0.02926	0.05797	0.12714	0.18801	Mean
Proposed Method GAN + DP on Steam Production Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.22320	0.40520	0.6869	0.85660	1
P_{PRO}	0.0	0.99820	0.70340	0.17690	0.38440	0.6530	0.84370	2
P_{PRO}	0.0	0.99830	0.70060	0.18270	0.39290	0.6782	0.85910	3
P_{PRO}	0.0	0.99900	0.69780	0.16050	0.37560	0.6525	0.84740	4
P_{PRO}	0.0	0.99870	0.70310	0.17070	0.37650	0.6535	0.83920	5
P_{PRO}	0.0	0.99900	0.70190	0.15910	0.37080	0.6490	0.83800	6
P_{PRO}	0.0	0.99840	0.69810	0.22340	0.41080	0.6717	0.85700	7
P_{PRO}	0.0	0.99850	0.70110	0.17650	0.37200	0.6430	0.84040	8
P_{PRO}	0.0	0.99830	0.70100	0.21840	0.40820	0.6637	0.84630	9
P_{PRO}	0.0	0.99830	0.70120	0.17800	0.38410	0.6585	0.84200	10
P_{PRO}	0.0	0.99848	0.70044	0.18694	0.38805	0.6610	0.84697	Mean
Proposed Method GAN + DP on Steam Production Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	10000	0.44580	0.09200	0.17780	0.27160	0.36010	1
P_{PCC}	0.0	10000	0.45020	0.08950	0.17990	0.27490	0.36620	2
P_{PCC}	0.0	0.9941	0.44970	0.08920	0.17940	0.27040	0.36500	3
P_{PCC}	0.0	10000	0.44690	0.08810	0.18070	0.27070	0.35790	4
P_{PCC}	0.0	0.9899	0.44380	0.09250	0.17960	0.27040	0.36210	5
P_{PCC}	0.0	0.9901	0.44820	0.08820	0.17820	0.26820	0.36370	6
P_{PCC}	0.0	0.9998	0.44320	0.09240	0.18130	0.27470	0.36080	7
P_{PCC}	0.0	10000	0.44510	0.09280	0.18470	0.27480	0.36670	8
P_{PCC}	0.0	0.9930	0.44330	0.08940	0.18320	0.27260	0.35930	9
P_{PCC}	0.0	0.9971	0.45050	0.09060	0.18500	0.27560	0.36250	10
P_{PCC}	0.0	0.9964	0.44667	0.09047	0.18098	0.27239	0.36243	Mean

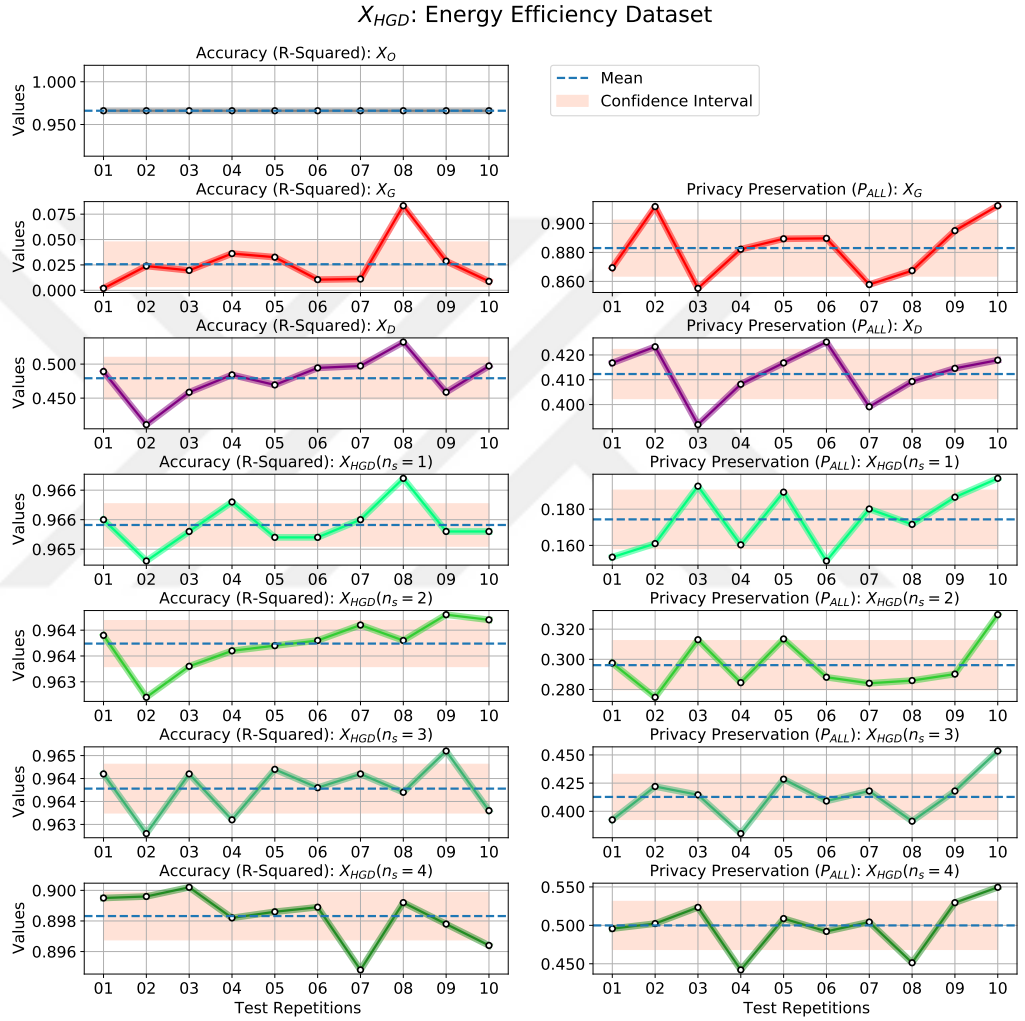


Figure C.3 Variations Among 10 Tests (X_{HGD}) - Energy Efficiency Dataset

Table C.5 Experiments in Chapter 4.7 - GAN + DP - Energy Efficiency

Proposed Method GAN + DP on Energy Efficiency Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	0.9661	0.00190	0.48910	0.96600	0.96440	0.96460	0.89950	01
R -Squared	0.9661	0.02390	0.41100	0.96530	0.96320	0.96330	0.89960	02
R -Squared	0.9661	0.01960	0.45860	0.96580	0.96380	0.96460	0.90020	03
R -Squared	0.9661	0.03620	0.48420	0.96630	0.96410	0.96360	0.89820	04
R -Squared	0.9661	0.03260	0.46940	0.96570	0.96420	0.96470	0.89860	05
R -Squared	0.9661	0.01050	0.49440	0.96570	0.96430	0.96430	0.89890	06
R -Squared	0.9661	0.01110	0.49730	0.96600	0.96460	0.96460	0.89480	07
R -Squared	0.9661	0.08330	0.53240	0.96670	0.96430	0.96420	0.89920	08
R -Squared	0.9661	0.02870	0.45870	0.96580	0.96480	0.96510	0.89780	09
R -Squared	0.9661	0.00880	0.49720	0.96580	0.96470	0.96380	0.89640	10
R -Squared	0.9661	0.02566	0.47923	0.96591	0.96424	0.96428	0.89832	Mean
Proposed Method GAN + DP on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.89180	0.47390	0.08580	0.24250	0.39180	0.52610	01
P_{MSE}	0.0	1.00000	0.51900	0.10970	0.24890	0.44730	0.57380	02
P_{MSE}	0.0	0.74220	0.38820	0.14600	0.25470	0.38820	0.46270	03
P_{MSE}	0.0	0.84590	0.41780	0.08220	0.18490	0.34590	0.39730	04
P_{MSE}	0.0	0.89440	0.44720	0.17250	0.30990	0.42960	0.58100	05
P_{MSE}	0.0	0.91150	0.49230	0.07690	0.22310	0.38080	0.47690	06
P_{MSE}	0.0	0.73530	0.35880	0.11760	0.22650	0.37350	0.50000	07
P_{MSE}	0.0	0.89390	0.47350	0.10980	0.22730	0.37500	0.44320	08
P_{MSE}	0.0	0.93800	0.45260	0.16060	0.29200	0.45620	0.63870	09
P_{MSE}	0.0	0.99210	0.49600	0.17060	0.30950	0.51980	0.64680	10
P_{MSE}	0.0	0.88451	0.45193	0.12317	0.25193	0.41081	0.52465	Mean
Proposed Method GAN + DP on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.06340	0.14100	0.20140	0.28400	01
P_{VAR}	0.0	0.52300	0.15090	0.06180	0.13200	0.19810	0.27060	02
P_{VAR}	0.0	0.52600	0.15130	0.08740	0.15510	0.21980	0.28570	03
P_{VAR}	0.0	0.53810	0.15030	0.06710	0.13690	0.20350	0.26250	04
P_{VAR}	0.0	0.55510	0.15130	0.08640	0.16400	0.22280	0.31200	05
P_{VAR}	0.0	0.52160	0.15060	0.05850	0.13070	0.19400	0.26840	06
P_{VAR}	0.0	0.53130	0.15080	0.07490	0.14850	0.21360	0.29200	07
P_{VAR}	0.0	0.51010	0.15130	0.07170	0.13790	0.20130	0.25960	08
P_{VAR}	0.0	0.56190	0.15090	0.08120	0.15820	0.21900	0.31590	09
P_{VAR}	0.0	0.54090	0.15110	0.07910	0.15180	0.21900	0.29760	10
P_{VAR}	0.0	0.53234	0.15096	0.07315	0.14561	0.20925	0.28483	Mean
Proposed Method GAN + DP on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.47720	0.72370	0.67900	0.8319	01
P_{HAUS}	0.0	0.97250	1.00000	0.50150	0.61500	0.80250	0.8487	02
P_{HAUS}	0.0	0.87580	0.91380	0.54120	0.73460	0.71340	1.0000	03
P_{HAUS}	0.0	0.91530	1.00000	0.50530	0.71720	0.62870	0.6653	04
P_{HAUS}	0.0	0.89050	1.00000	0.49700	0.66340	0.76300	0.7352	05
P_{HAUS}	0.0	0.92740	1.00000	0.49480	0.71490	0.79800	0.8823	06
P_{HAUS}	0.0	0.89190	1.00000	0.54290	0.62600	0.78080	0.8456	07
P_{HAUS}	0.0	0.83290	0.94430	0.52960	0.68500	0.68670	0.7250	08
P_{HAUS}	0.0	0.87520	1.00000	0.51140	0.55140	0.67980	0.7880	09
P_{HAUS}	0.0	0.94490	0.99170	0.57970	0.80640	0.84380	1.0000	10
P_{HAUS}	0.0	0.89647	0.98498	0.51806	0.68376	0.73757	0.8322	Mean

Table C.6 Experiments in Chapter 4.7 - GAN + DP - Energy Efficiency

Proposed Method GAN + DP on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.0825	0.17140	0.29210	0.37780	01
P_{KL}	0.0	1.0	0.32150	0.0836	0.16080	0.29580	0.37300	02
P_{KL}	0.0	1.0	0.34450	0.1171	0.20400	0.34110	0.41810	03
P_{KL}	0.0	1.0	0.33110	0.0927	0.16560	0.31130	0.37420	04
P_{KL}	0.0	1.0	0.34560	0.1208	0.21140	0.33220	0.43960	05
P_{KL}	0.0	1.0	0.34100	0.0820	0.17050	0.28850	0.37380	06
P_{KL}	0.0	1.0	0.32800	0.1029	0.18650	0.32480	0.42120	07
P_{KL}	0.0	1.0	0.33230	0.0903	0.16770	0.28710	0.35160	08
P_{KL}	0.0	1.0	0.33110	0.1126	0.19870	0.33110	0.44040	09
P_{KL}	0.0	1.0	0.31690	0.1015	0.18150	0.31380	0.39080	10
P_{KL}	0.0	1.0	0.33126	0.0986	0.18181	0.31178	0.39605	Mean
Proposed Method GAN + DP on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.11460	0.31260	0.50020	0.5595	01
P_{PRO}	0.0	0.9946	0.34650	0.11060	0.29540	0.49600	0.5541	02
P_{PRO}	0.0	0.9948	0.34910	0.15960	0.32140	0.52070	0.5652	03
P_{PRO}	0.0	0.9949	0.34690	0.10370	0.29190	0.47290	0.5421	04
P_{PRO}	0.0	0.9966	0.35290	0.15680	0.32890	0.52320	0.5846	05
P_{PRO}	0.0	0.9944	0.35920	0.09440	0.28960	0.49240	0.5505	06
P_{PRO}	0.0	0.9944	0.35240	0.14460	0.32350	0.51720	0.5800	07
P_{PRO}	0.0	0.9941	0.35050	0.12490	0.30130	0.49360	0.5360	08
P_{PRO}	0.0	0.9947	0.34950	0.15450	0.34010	0.52580	0.5941	09
P_{PRO}	0.0	0.9972	0.34910	0.15320	0.33580	0.52430	0.5739	10
P_{PRO}	0.0	0.9950	0.35075	0.13169	0.31405	0.50663	0.5640	Mean
Proposed Method GAN + DP on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.09730	0.19360	0.28990	0.39510	01
P_{PCC}	0.0	0.97960	0.20190	0.09850	0.19690	0.29240	0.39380	02
P_{PCC}	0.0	0.99240	0.20430	0.10510	0.20830	0.30440	0.40860	03
P_{PCC}	0.0	0.99910	0.20280	0.11090	0.21080	0.31870	0.41060	04
P_{PCC}	0.0	1.00000	0.20360	0.10280	0.20360	0.30040	0.40020	05
P_{PCC}	0.0	0.98320	0.20790	0.10200	0.20000	0.30100	0.40100	06
P_{PCC}	0.0	0.99410	0.20440	0.09770	0.19340	0.29810	0.38690	07
P_{PCC}	0.0	0.97310	0.20410	0.10310	0.19620	0.30320	0.39240	08
P_{PCC}	0.0	1.00000	0.20360	0.09930	0.20060	0.29590	0.40020	09
P_{PCC}	0.0	0.99810	0.20240	0.09770	0.19240	0.30010	0.38790	10
P_{PCC}	0.0	0.98965	0.20385	0.10144	0.19958	0.30041	0.39767	Mean

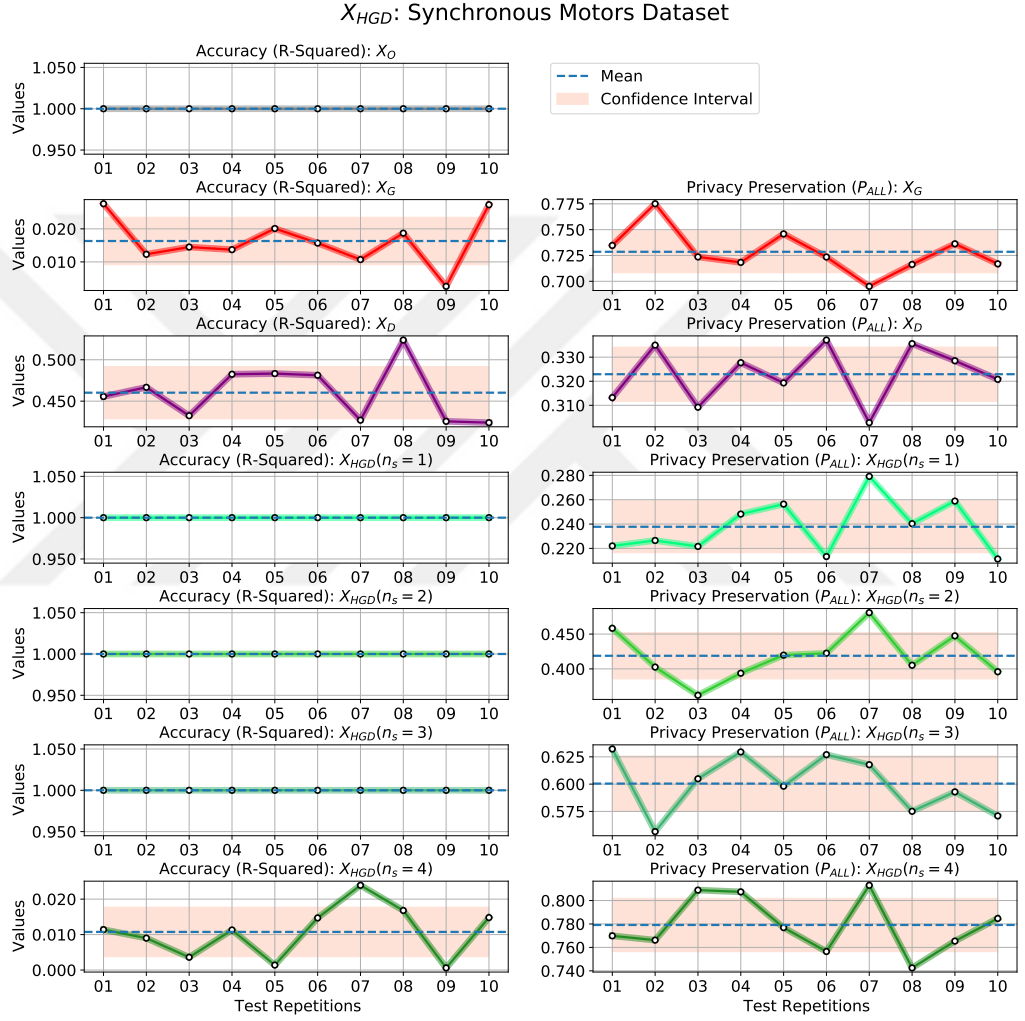


Figure C.4 Variations Among 10 Tests (X_{HGD}) - Synchronous Motors Dataset

Table C.7 Experiments in Chapter 4.7 - GAN + DP - Synchronous Motors

Proposed Method GAN + DP on Synchronous Motors Dataset: R-Squared values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	1.0	0.02760	0.45560	1.0	1.0	1.0	0.01140	1
<i>R-Squared</i>	1.0	0.01230	0.46670	1.0	1.0	1.0	0.00900	2
<i>R-Squared</i>	1.0	0.01450	0.43240	1.0	1.0	1.0	0.00360	3
<i>R-Squared</i>	1.0	0.01370	0.48250	1.0	1.0	1.0	0.01130	4
<i>R-Squared</i>	1.0	0.02010	0.48330	1.0	1.0	1.0	0.00140	5
<i>R-Squared</i>	1.0	0.01570	0.48130	1.0	1.0	1.0	0.01470	6
<i>R-Squared</i>	1.0	0.01070	0.42700	1.0	1.0	1.0	0.02390	7
<i>R-Squared</i>	1.0	0.01870	0.52380	1.0	1.0	1.0	0.01680	8
<i>R-Squared</i>	1.0	0.00260	0.42560	1.0	1.0	1.0	0.00060	9
<i>R-Squared</i>	1.0	0.02730	0.42400	1.0	1.0	1.0	0.01480	10
<i>R-Squared</i>	1.0	0.01632	0.46022	1.0	1.0	1.0	0.01075	Mean

Proposed Method GAN + DP on Synchronous Motors Dataset: P_{MSE} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.64180	0.42540	0.21640	0.65670	0.91790	10000	1
P_{MSE}	0.0	0.76790	0.46410	0.23630	0.47260	0.62450	0.9620	2
P_{MSE}	0.0	0.57140	0.34160	0.17390	0.29810	0.70190	10000	3
P_{MSE}	0.0	0.61640	0.38360	0.27400	0.41780	0.78770	10000	4
P_{MSE}	0.0	0.64790	0.40850	0.31690	0.52110	0.71830	10000	5
P_{MSE}	0.0	0.67690	0.43850	0.17690	0.50770	0.80770	10000	6
P_{MSE}	0.0	0.51180	0.33530	0.34710	0.61760	0.77650	10000	7
P_{MSE}	0.0	0.62880	0.43940	0.31060	0.50000	0.74240	10000	8
P_{MSE}	0.0	0.63500	0.41610	0.29930	0.53280	0.73720	10000	9
P_{MSE}	0.0	0.64290	0.45240	0.18250	0.45240	0.73020	10000	10
P_{MSE}	0.0	0.63408	0.41049	0.25339	0.49768	0.75443	0.9962	Mean

Proposed Method GAN + DP on Synchronous Motors Dataset: P_{VAR} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.13950	0.30620	0.44670	0.5604	1
P_{VAR}	0.0	0.55430	0.18330	0.14540	0.29430	0.40840	0.5810	2
P_{VAR}	0.0	0.53210	0.18390	0.14000	0.25210	0.43450	0.6200	3
P_{VAR}	0.0	0.52760	0.18260	0.15330	0.27390	0.43580	0.5921	4
P_{VAR}	0.0	0.55830	0.18440	0.16220	0.30670	0.43350	0.6240	5
P_{VAR}	0.0	0.55590	0.18360	0.12610	0.29060	0.45810	0.5912	6
P_{VAR}	0.0	0.51540	0.18260	0.16190	0.32180	0.46100	0.6198	7
P_{VAR}	0.0	0.52320	0.18380	0.13230	0.27530	0.39550	0.5727	8
P_{VAR}	0.0	0.55810	0.18320	0.16300	0.31640	0.44860	0.6116	9
P_{VAR}	0.0	0.51130	0.18340	0.12380	0.26580	0.41130	0.5902	10
P_{VAR}	0.0	0.53506	0.18342	0.14475	0.29031	0.43334	0.5963	Mean

Proposed Method GAN + DP on Synchronous Motors Dataset: P_{HAUS} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.6353	0.35320	0.19940	0.35140	0.44670	0.70300	1
P_{HAUS}	0.0	0.5595	0.43530	0.17960	0.25780	0.38090	0.66820	2
P_{HAUS}	0.0	0.5060	0.40010	0.20000	0.28210	0.43950	0.72090	3
P_{HAUS}	0.0	0.4219	0.42220	0.20410	0.28750	0.52980	0.83970	4
P_{HAUS}	0.0	0.5526	0.35430	0.18500	0.26020	0.48840	0.62000	5
P_{HAUS}	0.0	0.4249	0.43450	0.21730	0.33260	0.48420	0.56220	6
P_{HAUS}	0.0	0.4733	0.39010	0.26610	0.43620	0.43060	0.77450	7
P_{HAUS}	0.0	0.5093	0.43720	0.21680	0.28990	0.44210	0.54820	8
P_{HAUS}	0.0	0.5196	0.39280	0.21370	0.36830	0.37990	0.57020	9
P_{HAUS}	0.0	0.5846	0.37570	0.23230	0.34640	0.40660	0.76850	10
P_{HAUS}	0.0	0.5187	0.39954	0.21143	0.32124	0.44287	0.67754	Mean

Table C.8 Experiments in Chapter 4.7 - GAN + DP - Synchronous Motors

Proposed Method GAN + DP on Synchronous Motors Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.15240	0.37460	0.55240	0.65400	1
P_{KL}	0.0	0.77810	0.27010	0.16720	0.33440	0.48230	0.66880	2
P_{KL}	0.0	0.74250	0.28760	0.18730	0.32780	0.59530	0.78260	3
P_{KL}	0.0	0.76160	0.30460	0.19870	0.35760	0.58940	0.72190	4
P_{KL}	0.0	0.72480	0.30200	0.20130	0.36910	0.52350	0.71140	5
P_{KL}	0.0	0.68850	0.30160	0.16390	0.35410	0.55080	0.67540	6
P_{KL}	0.0	0.67520	0.27650	0.22510	0.42440	0.57880	0.75240	7
P_{KL}	0.0	0.67740	0.29030	0.18060	0.33550	0.48390	0.64520	8
P_{KL}	0.0	0.71520	0.29800	0.21190	0.38410	0.55630	0.70200	9
P_{KL}	0.0	0.59690	0.27080	0.14150	0.28920	0.46770	0.62770	10
P_{KL}	0.0	0.70205	0.28745	0.18299	0.35508	0.53804	0.69414	Mean
Proposed Method GAN + DP on Synchronous Motors Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.43180	0.67980	0.85610	0.9237	1
P_{PRO}	0.0	0.99820	0.41830	0.42350	0.64690	0.83220	0.9234	2
P_{PRO}	0.0	0.99850	0.40750	0.43020	0.61200	0.84920	0.9350	3
P_{PRO}	0.0	0.99750	0.42730	0.46500	0.65030	0.84720	0.9232	4
P_{PRO}	0.0	0.99790	0.42730	0.48140	0.67750	0.84190	0.9291	5
P_{PRO}	0.0	0.99480	0.42480	0.39570	0.64790	0.86150	0.9241	6
P_{PRO}	0.0	0.99750	0.40320	0.47340	0.68900	0.85770	0.9293	7
P_{PRO}	0.0	0.99550	0.42320	0.42890	0.64390	0.82560	0.9259	8
P_{PRO}	0.0	0.99700	0.43450	0.46290	0.68160	0.84560	0.9322	9
P_{PRO}	0.0	0.99620	0.40890	0.39430	0.62650	0.83140	0.9181	10
P_{PRO}	0.0	0.99706	0.41868	0.43871	0.65554	0.84484	0.9264	Mean
Proposed Method GAN + DP on Synchronous Motors Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.19260	0.38120	0.57380	0.77830	1
P_{PCC}	0.0	0.99250	0.23870	0.20690	0.40970	0.61060	0.79360	2
P_{PCC}	0.0	0.99140	0.23430	0.19830	0.40060	0.60890	0.79510	3
P_{PCC}	0.0	0.98510	0.24580	0.19380	0.37570	0.58750	0.76730	4
P_{PCC}	0.0	0.99410	0.23910	0.19170	0.38340	0.58300	0.77670	5
P_{PCC}	0.0	100000	0.23960	0.20000	0.40200	0.60000	0.78610	6
P_{PCC}	0.0	0.99710	0.22930	0.20140	0.39490	0.60230	0.80170	7
P_{PCC}	0.0	0.96320	0.23980	0.17240	0.38650	0.56090	0.76310	8
P_{PCC}	0.0	0.99300	0.24630	0.20260	0.40120	0.58990	0.77660	9
P_{PCC}	0.0	0.96920	0.23330	0.19340	0.39490	0.57830	0.80370	10
P_{PCC}	0.0	0.98446	0.23785	0.19531	0.39301	0.58952	0.78422	Mean

X_{HG} : Wind Turbine Dataset

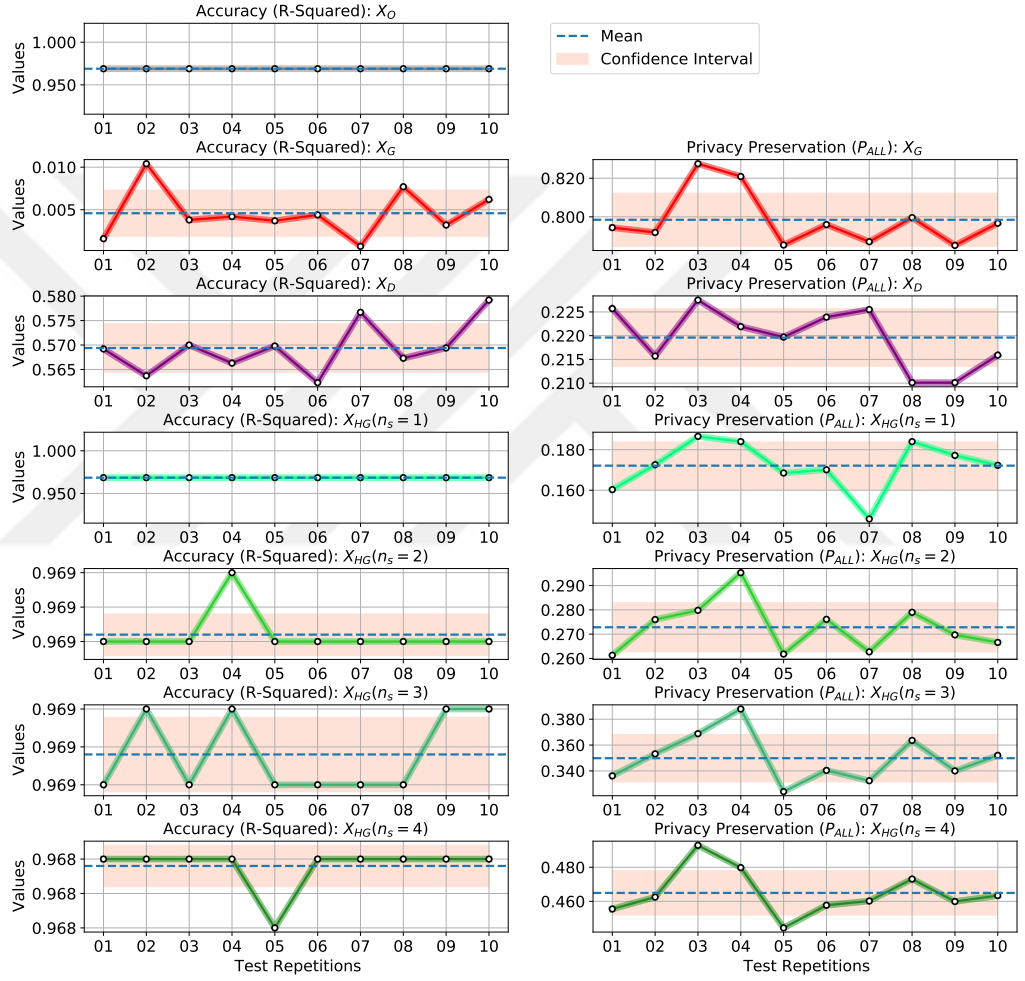


Figure C.5 Variations Among 10 Tests (X_{HG}) - Wind Turbine Dataset

Table C.9 Experiments in Chapter 4.7 - GAN Only - Wind Turbine

Proposed Method GAN Only on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.9689	0.00160	0.56920	0.9685	0.96850	0.96850	0.96820	01
<i>R-Squared</i>	0.9689	0.01040	0.56370	0.9685	0.96850	0.96860	0.96820	02
<i>R-Squared</i>	0.9689	0.00380	0.57000	0.9685	0.96850	0.96850	0.96820	03
<i>R-Squared</i>	0.9689	0.00420	0.56630	0.9685	0.96860	0.96860	0.96820	04
<i>R-Squared</i>	0.9689	0.00370	0.56980	0.9685	0.96850	0.96850	0.96810	05
<i>R-Squared</i>	0.9689	0.00440	0.56230	0.9685	0.96850	0.96850	0.96820	06
<i>R-Squared</i>	0.9689	0.00070	0.57670	0.9685	0.96850	0.96850	0.96820	07
<i>R-Squared</i>	0.9689	0.00770	0.56730	0.9685	0.96850	0.96850	0.96820	08
<i>R-Squared</i>	0.9689	0.00320	0.56940	0.9685	0.96850	0.96860	0.96820	09
<i>R-Squared</i>	0.9689	0.00620	0.57920	0.9685	0.96850	0.96860	0.96820	10
<i>R-Squared</i>	0.9689	0.00459	0.56939	0.9685	0.96851	0.96854	0.96819	Mean
Proposed Method GAN Only on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.75310	0.22720	0.08370	0.16150	0.19120	0.32890	01
P_{MSE}	0.0	0.82570	0.22910	0.10250	0.17470	0.21690	0.38570	02
P_{MSE}	0.0	0.85480	0.22720	0.13140	0.21510	0.26320	0.41260	03
P_{MSE}	0.0	0.82150	0.21420	0.10970	0.19070	0.24860	0.38740	04
P_{MSE}	0.0	0.69170	0.21380	0.09020	0.16300	0.19130	0.30350	05
P_{MSE}	0.0	0.78950	0.22910	0.11430	0.19870	0.23500	0.36750	06
P_{MSE}	0.0	0.70280	0.21720	0.07440	0.14280	0.17720	0.29720	07
P_{MSE}	0.0	0.82920	0.22420	0.12120	0.19960	0.25420	0.39960	08
P_{MSE}	0.0	0.73390	0.21130	0.11670	0.18910	0.21670	0.33890	09
P_{MSE}	0.0	0.76000	0.21720	0.09720	0.16560	0.20560	0.34280	10
P_{MSE}	0.0	0.77622	0.22105	0.10413	0.18008	0.21999	0.35641	Mean
Proposed Method GAN Only on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10250	0.08810	0.17290	0.28770	0.37640	01
P_{VAR}	0.0	0.63820	0.10240	0.09880	0.17740	0.31980	0.41650	02
P_{VAR}	0.0	0.66980	0.10250	0.11590	0.20540	0.35510	0.44860	03
P_{VAR}	0.0	0.70830	0.10230	0.10840	0.19780	0.38760	0.48020	04
P_{VAR}	0.0	0.58470	0.10250	0.09460	0.18050	0.29130	0.37320	05
P_{VAR}	0.0	0.60790	0.10230	0.10310	0.18840	0.30850	0.39460	06
P_{VAR}	0.0	0.59730	0.10230	0.08360	0.16570	0.29250	0.37830	07
P_{VAR}	0.0	0.66230	0.10240	0.10890	0.19050	0.35210	0.44230	08
P_{VAR}	0.0	0.62930	0.10240	0.11390	0.19970	0.31790	0.40650	09
P_{VAR}	0.0	0.65040	0.10250	0.09850	0.18120	0.33420	0.42730	10
P_{VAR}	0.0	0.63441	0.10241	0.10138	0.18595	0.32467	0.41439	Mean
Proposed Method GAN Only on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.36710	0.42210	0.4830	0.43030	01
P_{HAUS}	0.0	0.36360	0.32540	0.37060	0.47260	0.4956	0.33350	02
P_{HAUS}	0.0	0.47530	0.39450	0.35530	0.35520	0.4341	0.40710	03
P_{HAUS}	0.0	0.41860	0.38010	0.38830	0.49510	0.5211	0.31830	04
P_{HAUS}	0.0	0.50480	0.36700	0.36830	0.38050	0.3753	0.37520	05
P_{HAUS}	0.0	0.46330	0.37260	0.31920	0.40810	0.3982	0.35290	06
P_{HAUS}	0.0	0.52240	0.40400	0.30100	0.46810	0.4719	0.49060	07
P_{HAUS}	0.0	0.37270	0.30360	0.38590	0.42860	0.4602	0.35920	08
P_{HAUS}	0.0	0.37460	0.31100	0.31670	0.33260	0.3666	0.34670	09
P_{HAUS}	0.0	0.47310	0.36060	0.38170	0.41690	0.4780	0.38210	10
P_{HAUS}	0.0	0.44679	0.36225	0.35541	0.41798	0.4484	0.37959	Mean

Table C.10 Experiments in Chapter 4.7 - GAN Only - Wind Turbine

Proposed Method GAN Only on Wind Turbine Dataset: P_{KL} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.92060	0.22220	0.13140	0.22220	0.27210	0.42630	01
P_{KL}	0.0	0.93250	0.23440	0.14690	0.22960	0.28010	0.44570	02
P_{KL}	0.0	0.96520	0.23880	0.16720	0.25790	0.31070	0.47290	03
P_{KL}	0.0	0.97910	0.23640	0.16090	0.25530	0.31690	0.47780	04
P_{KL}	0.0	0.94430	0.23490	0.15340	0.24460	0.29230	0.45070	05
P_{KL}	0.0	0.92260	0.23900	0.15930	0.24360	0.29050	0.44950	06
P_{KL}	0.0	0.90030	0.22960	0.13310	0.22060	0.27110	0.42730	07
P_{KL}	0.0	0.94940	0.23030	0.15190	0.23520	0.28580	0.44230	08
P_{KL}	0.0	0.97910	0.23640	0.17020	0.26030	0.31230	0.47320	09
P_{KL}	0.0	0.89660	0.21540	0.14060	0.22860	0.27690	0.43080	10
P_{KL}	0.0	0.93897	0.23174	0.15149	0.23979	0.29087	0.44965	Mean

Proposed Method GAN Only on Wind Turbine Dataset: P_{PRO} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.14960	0.30270	0.35320	0.59820	01
P_{PRO}	0.0	1.0	0.25540	0.17610	0.32010	0.38200	0.62650	02
P_{PRO}	0.0	1.0	0.25470	0.20420	0.35920	0.41910	0.64420	03
P_{PRO}	0.0	1.0	0.25300	0.19680	0.35030	0.42650	0.64690	04
P_{PRO}	0.0	1.0	0.25500	0.16230	0.31870	0.36730	0.59610	05
P_{PRO}	0.0	1.0	0.25460	0.18290	0.33490	0.38460	0.61430	06
P_{PRO}	0.0	1.0	0.25440	0.14040	0.29440	0.35290	0.59660	07
P_{PRO}	0.0	1.0	0.25450	0.19610	0.34120	0.40880	0.63310	08
P_{PRO}	0.0	1.0	0.25460	0.20230	0.35020	0.39930	0.62830	09
P_{PRO}	0.0	1.0	0.25430	0.17460	0.32400	0.39020	0.62760	10
P_{PRO}	0.0	1.0	0.25444	0.17853	0.32957	0.38839	0.62118	Mean

Proposed Method GAN Only on Wind Turbine Dataset: P_{PCC} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.99700	0.14460	0.14190	0.28650	0.42980	0.57290	01
P_{PCC}	0.0	0.99160	0.14780	0.14060	0.28140	0.42480	0.56690	02
P_{PCC}	0.0	1.00000	0.14730	0.14590	0.28610	0.43060	0.57220	03
P_{PCC}	0.0	0.99770	0.14560	0.13990	0.28260	0.42670	0.56810	04
P_{PCC}	0.0	0.98680	0.14540	0.14260	0.28370	0.42630	0.56750	05
P_{PCC}	0.0	0.99300	0.14560	0.14150	0.28290	0.42570	0.56720	06
P_{PCC}	0.0	1.00000	0.14530	0.14250	0.28490	0.42880	0.57120	07
P_{PCC}	0.0	0.98390	0.14580	0.14010	0.27890	0.42050	0.56200	08
P_{PCC}	0.0	0.99440	0.14470	0.14330	0.28660	0.42700	0.56600	09
P_{PCC}	0.0	1.00000	0.14530	0.14100	0.28350	0.42740	0.56970	10
P_{PCC}	0.0	0.99444	0.14574	0.14193	0.28371	0.42676	0.56837	Mean

X_{HG} : Steam Production Dataset

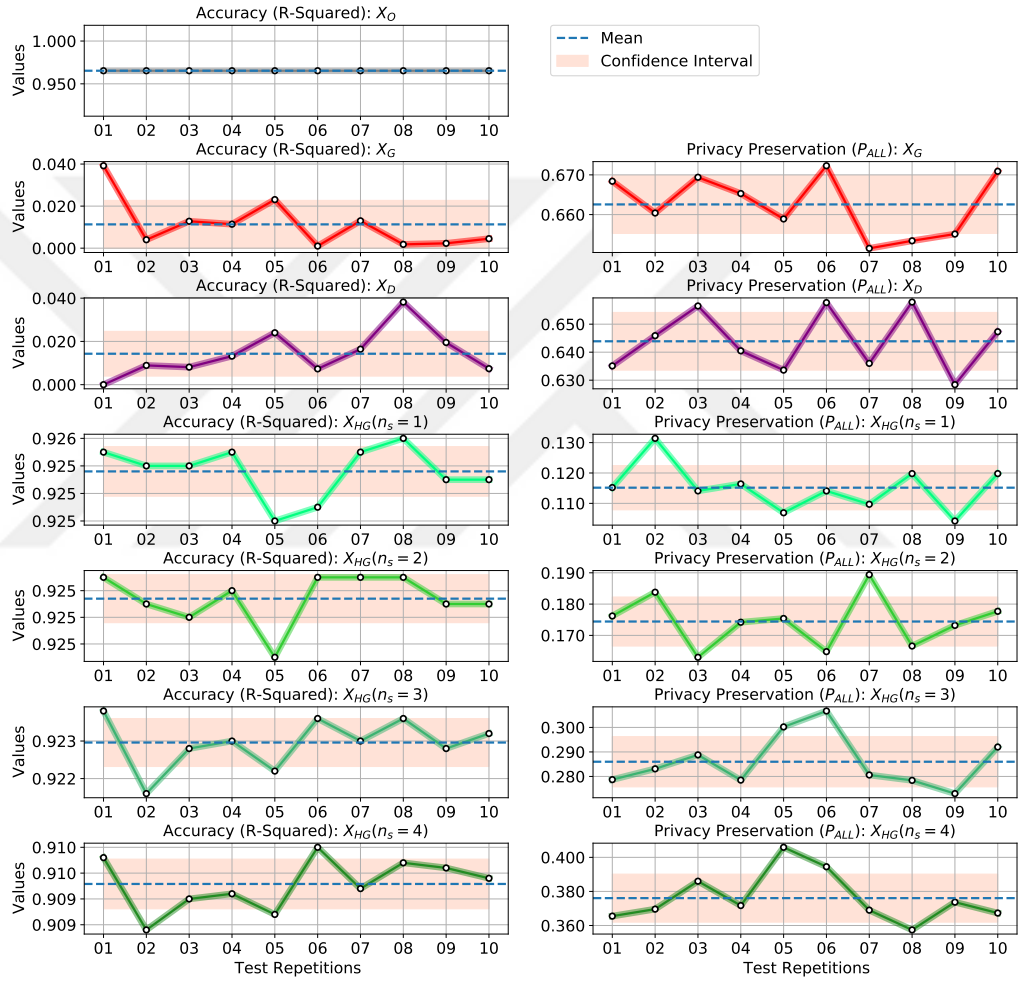


Figure C.6 Variations Among 10 Tests (X_{HG}) - Steam Production Dataset

Table C.11 Experiments in Chapter 4.7 - GAN Only - Steam Production

Proposed Method GAN Only on Steam Production Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
$R\text{-Squared}$	0.9653	0.03920	0.0000	0.92550	0.92550	0.92340	0.91030	01
$R\text{-Squared}$	0.9653	0.00400	0.0089	0.92540	0.92530	0.92230	0.90890	02
$R\text{-Squared}$	0.9653	0.01280	0.0081	0.92540	0.92520	0.92290	0.90950	03
$R\text{-Squared}$	0.9653	0.01140	0.0132	0.92550	0.92540	0.92300	0.90960	04
$R\text{-Squared}$	0.9653	0.02310	0.0240	0.92500	0.92490	0.92260	0.90920	05
$R\text{-Squared}$	0.9653	0.00100	0.0073	0.92510	0.92550	0.92330	0.91050	06
$R\text{-Squared}$	0.9653	0.01300	0.0164	0.92550	0.92550	0.92300	0.90970	07
$R\text{-Squared}$	0.9653	0.00180	0.0382	0.92560	0.92550	0.92330	0.91020	08
$R\text{-Squared}$	0.9653	0.00230	0.0195	0.92530	0.92530	0.92290	0.91010	09
$R\text{-Squared}$	0.9653	0.00450	0.0074	0.92530	0.92530	0.92310	0.90990	10
$R\text{-Squared}$	0.9653	0.01131	0.0143	0.92536	0.92534	0.92298	0.90979	Mean
Proposed Method GAN Only on Steam Production Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.30040	0.5477	0.01880	0.0305	0.08740	0.14440	01
P_{MSE}	0.0	0.29920	0.5409	0.01520	0.0308	0.08820	0.14560	02
P_{MSE}	0.0	0.28540	0.5402	0.01510	0.0268	0.09120	0.14440	03
P_{MSE}	0.0	0.27980	0.5154	0.01460	0.0259	0.07370	0.13240	04
P_{MSE}	0.0	0.29330	0.4976	0.01420	0.0358	0.08580	0.14690	05
P_{MSE}	0.0	0.31050	0.5447	0.01520	0.0308	0.08820	0.15740	06
P_{MSE}	0.0	0.27640	0.5164	0.01800	0.0328	0.08360	0.13440	07
P_{MSE}	0.0	0.27750	0.5432	0.01530	0.0271	0.08090	0.13090	08
P_{MSE}	0.0	0.28990	0.4953	0.01400	0.0319	0.08130	0.14510	09
P_{MSE}	0.0	0.28720	0.5056	0.01440	0.0256	0.08000	0.14200	10
P_{MSE}	0.0	0.28996	0.5247	0.01548	0.0298	0.08403	0.14235	Mean
Proposed Method GAN Only on Steam Production Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.06340	0.13000	0.18580	0.23890	01
P_{VAR}	0.0	0.65740	1.0	0.05350	0.13520	0.18790	0.23870	02
P_{VAR}	0.0	0.63460	1.0	0.05790	0.12540	0.17910	0.22900	03
P_{VAR}	0.0	0.66250	1.0	0.05800	0.12840	0.17720	0.23220	04
P_{VAR}	0.0	0.68900	1.0	0.06130	0.17130	0.22280	0.27770	05
P_{VAR}	0.0	0.64830	1.0	0.05480	0.12980	0.18250	0.23710	06
P_{VAR}	0.0	0.66160	1.0	0.06700	0.14600	0.19850	0.24690	07
P_{VAR}	0.0	0.63930	1.0	0.06170	0.13240	0.18250	0.22930	08
P_{VAR}	0.0	0.67320	1.0	0.06330	0.15280	0.20320	0.26050	09
P_{VAR}	0.0	0.64810	1.0	0.05970	0.13140	0.18470	0.23680	10
P_{VAR}	0.0	0.65727	1.0	0.06006	0.13827	0.19042	0.24271	Mean
Proposed Method GAN Only on Steam Production Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.38970	0.49590	0.49770	0.53420	01
P_{HAUS}	0.0	0.70670	0.92660	0.52300	0.53600	0.54090	0.56630	02
P_{HAUS}	0.0	0.80610	0.98410	0.40210	0.43220	0.57970	0.69080	03
P_{HAUS}	0.0	0.73450	0.92090	0.41000	0.48590	0.53650	0.57850	04
P_{HAUS}	0.0	0.64090	0.88860	0.34610	0.38600	0.56580	0.69720	05
P_{HAUS}	0.0	0.75500	0.98880	0.40960	0.42720	0.66530	0.66540	06
P_{HAUS}	0.0	0.67750	0.90110	0.34710	0.53240	0.50010	0.57250	07
P_{HAUS}	0.0	0.71480	1.00000	0.42500	0.43410	0.51990	0.53140	08
P_{HAUS}	0.0	0.63940	0.87210	0.32710	0.42090	0.44510	0.53420	09
P_{HAUS}	0.0	0.79240	0.98850	0.42950	0.50270	0.58970	0.53840	10
P_{HAUS}	0.0	0.72235	0.93403	0.40092	0.46533	0.54407	0.59089	Mean

Table C.12 Experiments in Chapter 4.7 - GAN Only - Steam Production

Proposed Method GAN Only on Steam Production Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.01750	0.02890	0.08950	0.14440	01
P_{KL}	0.0	0.30100	0.25430	0.01450	0.03220	0.09650	0.14920	02
P_{KL}	0.0	0.29800	0.26450	0.01840	0.03040	0.08830	0.13380	03
P_{KL}	0.0	0.31620	0.26190	0.02420	0.03910	0.10230	0.15960	04
P_{KL}	0.0	0.34160	0.26850	0.02450	0.05200	0.11580	0.17990	05
P_{KL}	0.0	0.33080	0.26230	0.02100	0.03570	0.11020	0.17870	06
P_{KL}	0.0	0.29520	0.25720	0.01770	0.03500	0.09650	0.13730	07
P_{KL}	0.0	0.29030	0.25810	0.02060	0.03520	0.10260	0.14350	08
P_{KL}	0.0	0.33710	0.25890	0.01820	0.03910	0.10530	0.16850	09
P_{KL}	0.0	0.30220	0.23780	0.01970	0.03350	0.10060	0.16220	10
P_{KL}	0.0	0.31095	0.25746	0.01963	0.03611	0.10076	0.15571	Mean
Proposed Method GAN Only on Steam Production Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.11140	0.18960	0.53950	0.77080	01
P_{PRO}	0.0	0.99820	0.70340	0.09100	0.19020	0.51320	0.75540	02
P_{PRO}	0.0	0.99830	0.70060	0.10010	0.18200	0.52220	0.75490	03
P_{PRO}	0.0	0.99900	0.69780	0.10200	0.18410	0.50960	0.76480	04
P_{PRO}	0.0	0.99870	0.70310	0.10380	0.22330	0.53790	0.77150	05
P_{PRO}	0.0	0.99900	0.70190	0.09510	0.18620	0.52500	0.77010	06
P_{PRO}	0.0	0.99840	0.69810	0.11850	0.20990	0.53330	0.76280	07
P_{PRO}	0.0	0.99850	0.70110	0.10230	0.18590	0.50860	0.74170	08
P_{PRO}	0.0	0.99830	0.70100	0.11330	0.21230	0.53300	0.77570	09
P_{PRO}	0.0	0.99830	0.70120	0.10380	0.18970	0.52230	0.76270	10
P_{PRO}	0.0	0.99848	0.70044	0.10413	0.19532	0.52446	0.76304	Mean
Proposed Method GAN Only on Steam Production Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	1.0000	0.44580	0.09020	0.1823	0.27250	0.36010	01
P_{PCC}	0.0	1.0000	0.45020	0.09130	0.1781	0.27210	0.36250	02
P_{PCC}	0.0	0.9941	0.44970	0.09100	0.1812	0.27220	0.36320	03
P_{PCC}	0.0	1.0000	0.44690	0.08990	0.1816	0.27160	0.36240	04
P_{PCC}	0.0	0.9899	0.44380	0.09160	0.1842	0.27310	0.36210	05
P_{PCC}	0.0	0.9901	0.44820	0.08910	0.1791	0.26910	0.35820	06
P_{PCC}	0.0	0.9998	0.44320	0.08970	0.1804	0.27190	0.35990	07
P_{PCC}	0.0	1.0000	0.44510	0.09360	0.1847	0.27570	0.36760	08
P_{PCC}	0.0	0.9930	0.44330	0.08940	0.1823	0.26990	0.35840	09
P_{PCC}	0.0	0.9971	0.45050	0.09150	0.1831	0.27470	0.36170	10
P_{PCC}	0.0	0.9964	0.44667	0.09073	0.1817	0.27228	0.36161	Mean

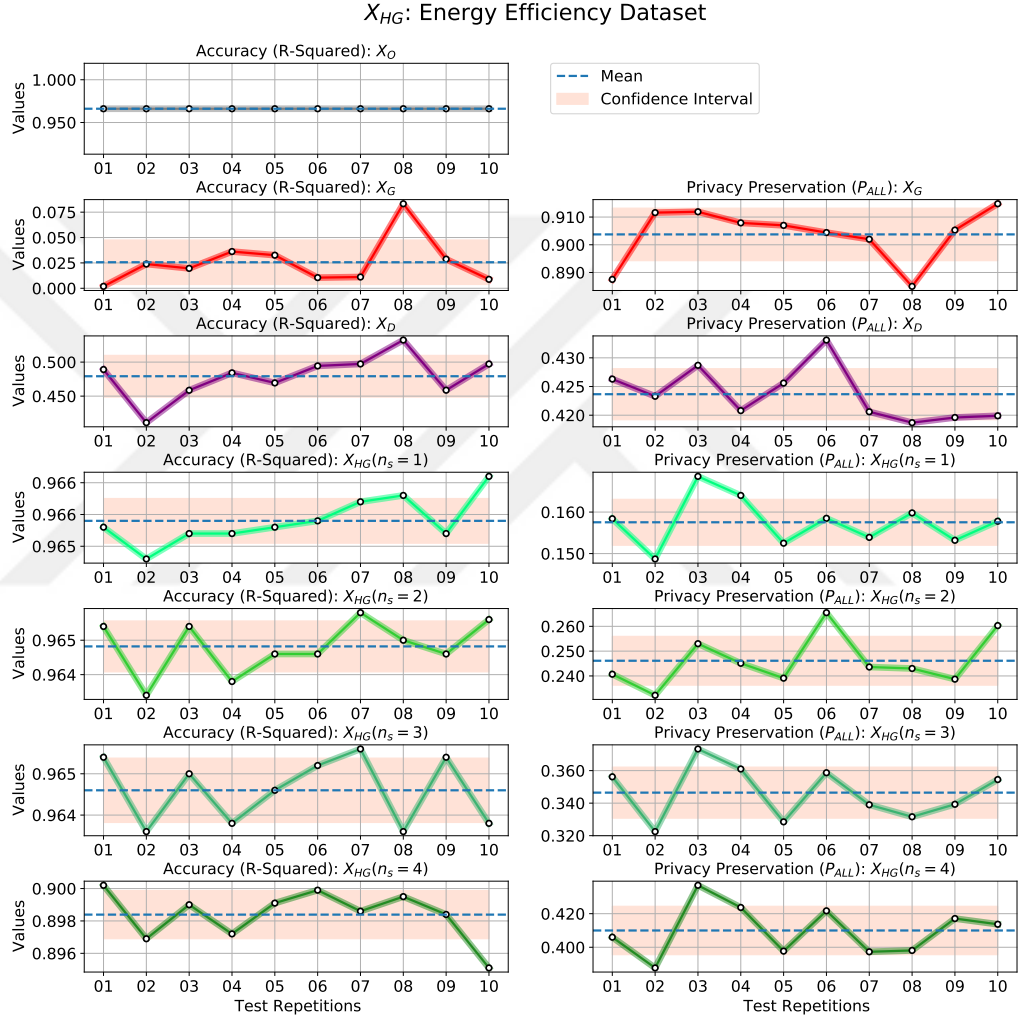


Figure C.7 Variations Among 10 Tests (X_{HG}) - Energy Efficiency Dataset

Table C.13 Experiments in Chapter 4.7 - GAN Only - Energy Efficiency

Proposed Method GAN Only on Energy Efficiency Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.9661	0.00190	0.48910	0.9658	0.96470	0.9647	0.90020	01
<i>R-Squared</i>	0.9661	0.02390	0.41100	0.9653	0.96370	0.9638	0.89690	02
<i>R-Squared</i>	0.9661	0.01960	0.45860	0.9657	0.96470	0.9645	0.89900	03
<i>R-Squared</i>	0.9661	0.03620	0.48420	0.9657	0.96390	0.9639	0.89720	04
<i>R-Squared</i>	0.9661	0.03260	0.46940	0.9658	0.96430	0.9643	0.89910	05
<i>R-Squared</i>	0.9661	0.01050	0.49440	0.9659	0.96430	0.9646	0.89990	06
<i>R-Squared</i>	0.9661	0.01110	0.49730	0.9662	0.96490	0.9648	0.89860	07
<i>R-Squared</i>	0.9661	0.08330	0.53240	0.9663	0.96450	0.9638	0.89950	08
<i>R-Squared</i>	0.9661	0.02870	0.45870	0.9657	0.96430	0.9647	0.89840	09
<i>R-Squared</i>	0.9661	0.00880	0.49720	0.9666	0.96480	0.9639	0.89510	10
<i>R-Squared</i>	0.9661	0.02566	0.47923	0.9659	0.96441	0.9643	0.89839	Mean
Proposed Method GAN Only on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	1.0	0.53140	0.07950	0.16740	0.27620	0.33890	01
P_{MSE}	0.0	1.0	0.51900	0.07170	0.15190	0.27000	0.33760	02
P_{MSE}	0.0	1.0	0.52300	0.09210	0.17990	0.30130	0.35560	03
P_{MSE}	0.0	1.0	0.49390	0.07690	0.17000	0.28340	0.34410	04
P_{MSE}	0.0	1.0	0.50000	0.08660	0.17720	0.29530	0.36610	05
P_{MSE}	0.0	1.0	0.54010	0.08020	0.16030	0.28270	0.34180	06
P_{MSE}	0.0	1.0	0.48800	0.08400	0.16000	0.26800	0.33600	07
P_{MSE}	0.0	1.0	0.52970	0.08470	0.17370	0.28810	0.33900	08
P_{MSE}	0.0	1.0	0.48250	0.08950	0.17120	0.27240	0.35020	09
P_{MSE}	0.0	1.0	0.50000	0.08800	0.17200	0.28800	0.35600	10
P_{MSE}	0.0	1.0	0.51076	0.08332	0.16836	0.28254	0.34653	Mean
Proposed Method GAN Only on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.05380	0.10940	0.15640	0.21020	01
P_{VAR}	0.0	0.52300	0.15090	0.04920	0.09970	0.15020	0.20770	02
P_{VAR}	0.0	0.52600	0.15130	0.05910	0.11390	0.16570	0.21630	03
P_{VAR}	0.0	0.53810	0.15030	0.05450	0.11270	0.16270	0.21610	04
P_{VAR}	0.0	0.55510	0.15130	0.05860	0.11770	0.17180	0.23270	05
P_{VAR}	0.0	0.52160	0.15060	0.05220	0.10190	0.15410	0.20830	06
P_{VAR}	0.0	0.53130	0.15080	0.05850	0.10770	0.15530	0.20960	07
P_{VAR}	0.0	0.51010	0.15130	0.05860	0.11290	0.16240	0.20760	08
P_{VAR}	0.0	0.56190	0.15090	0.06260	0.11880	0.16630	0.22580	09
P_{VAR}	0.0	0.54090	0.15110	0.05860	0.11440	0.16600	0.22540	10
P_{VAR}	0.0	0.53234	0.15096	0.05657	0.11091	0.16109	0.21597	Mean
Proposed Method GAN Only on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.53700	0.56350	0.72780	0.70070	01
P_{HAUS}	0.0	0.97250	1.00000	0.52250	0.58100	0.55590	0.60010	02
P_{HAUS}	0.0	0.95840	1.00000	0.57690	0.61270	0.74350	0.80190	03
P_{HAUS}	0.0	0.91530	1.00000	0.57780	0.56780	0.69120	0.72320	04
P_{HAUS}	0.0	0.89050	1.00000	0.48840	0.51730	0.48210	0.53550	05
P_{HAUS}	0.0	0.92740	1.00000	0.53920	0.74300	0.71730	0.76210	06
P_{HAUS}	0.0	0.89190	1.00000	0.50210	0.60910	0.62740	0.64600	07
P_{HAUS}	0.0	0.83290	0.94430	0.53350	0.55770	0.52820	0.63340	08
P_{HAUS}	0.0	0.87520	1.00000	0.47230	0.51360	0.58190	0.68950	09
P_{HAUS}	0.0	0.95280	1.00000	0.52490	0.67220	0.67810	0.69280	10
P_{HAUS}	0.0	0.90552	0.99443	0.52746	0.59379	0.63334	0.67852	Mean

Table C.14 Experiments in Chapter 4.7 - GAN Only - Energy Efficiency

Proposed Method GAN Only on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.09210	0.16510	0.26670	0.32700	01
P_{KL}	0.0	1.0	0.32150	0.07400	0.14150	0.25080	0.31830	02
P_{KL}	0.0	1.0	0.34450	0.08360	0.16050	0.29100	0.36450	03
P_{KL}	0.0	1.0	0.33110	0.08940	0.17220	0.29470	0.37750	04
P_{KL}	0.0	1.0	0.34560	0.08390	0.16780	0.27850	0.35230	05
P_{KL}	0.0	1.0	0.34100	0.09840	0.16390	0.28200	0.35410	06
P_{KL}	0.0	1.0	0.32800	0.08040	0.14790	0.26370	0.32480	07
P_{KL}	0.0	1.0	0.33230	0.08710	0.16450	0.28060	0.34520	08
P_{KL}	0.0	1.0	0.33110	0.08610	0.16560	0.27810	0.34770	09
P_{KL}	0.0	1.0	0.31690	0.07690	0.14770	0.25850	0.32000	10
P_{KL}	0.0	1.0	0.33126	0.08519	0.15967	0.27446	0.34314	Mean
Proposed Method GAN Only on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.09150	0.24310	0.41740	0.46880	01
P_{PRO}	0.0	0.9946	0.34650	0.07710	0.22330	0.41200	0.46890	02
P_{PRO}	0.0	0.9948	0.34910	0.09800	0.24480	0.43080	0.47720	03
P_{PRO}	0.0	0.9949	0.34690	0.07720	0.23610	0.41920	0.46580	04
P_{PRO}	0.0	0.9966	0.35290	0.09760	0.25290	0.44030	0.49440	05
P_{PRO}	0.0	0.9944	0.35920	0.07900	0.22590	0.41500	0.46610	06
P_{PRO}	0.0	0.9944	0.35240	0.09950	0.24170	0.42660	0.47840	07
P_{PRO}	0.0	0.9941	0.35050	0.09110	0.24710	0.42810	0.47200	08
P_{PRO}	0.0	0.9947	0.34950	0.10870	0.26110	0.43990	0.49230	09
P_{PRO}	0.0	0.9972	0.34910	0.09990	0.25720	0.44010	0.49150	10
P_{PRO}	0.0	0.9950	0.35075	0.09196	0.24332	0.42694	0.47754	Mean
Proposed Method GAN Only on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.09630	0.19560	0.29290	0.39020	01
P_{PCC}	0.0	0.97960	0.20190	0.09750	0.19590	0.29640	0.39380	02
P_{PCC}	0.0	0.99240	0.20430	0.10210	0.20630	0.30740	0.40560	03
P_{PCC}	0.0	0.99910	0.20280	0.10790	0.21180	0.31470	0.41560	04
P_{PCC}	0.0	1.00000	0.20360	0.09980	0.20160	0.30340	0.40510	05
P_{PCC}	0.0	0.98320	0.20790	0.10200	0.19800	0.30100	0.39800	06
P_{PCC}	0.0	0.99410	0.20440	0.09870	0.19540	0.29310	0.38890	07
P_{PCC}	0.0	0.97310	0.20410	0.10410	0.20220	0.30220	0.39140	08
P_{PCC}	0.0	1.00000	0.20360	0.10030	0.20160	0.29690	0.39720	09
P_{PCC}	0.0	0.99810	0.20240	0.09870	0.19840	0.29610	0.39680	10
P_{PCC}	0.0	0.98965	0.20385	0.10074	0.20068	0.30041	0.39826	Mean

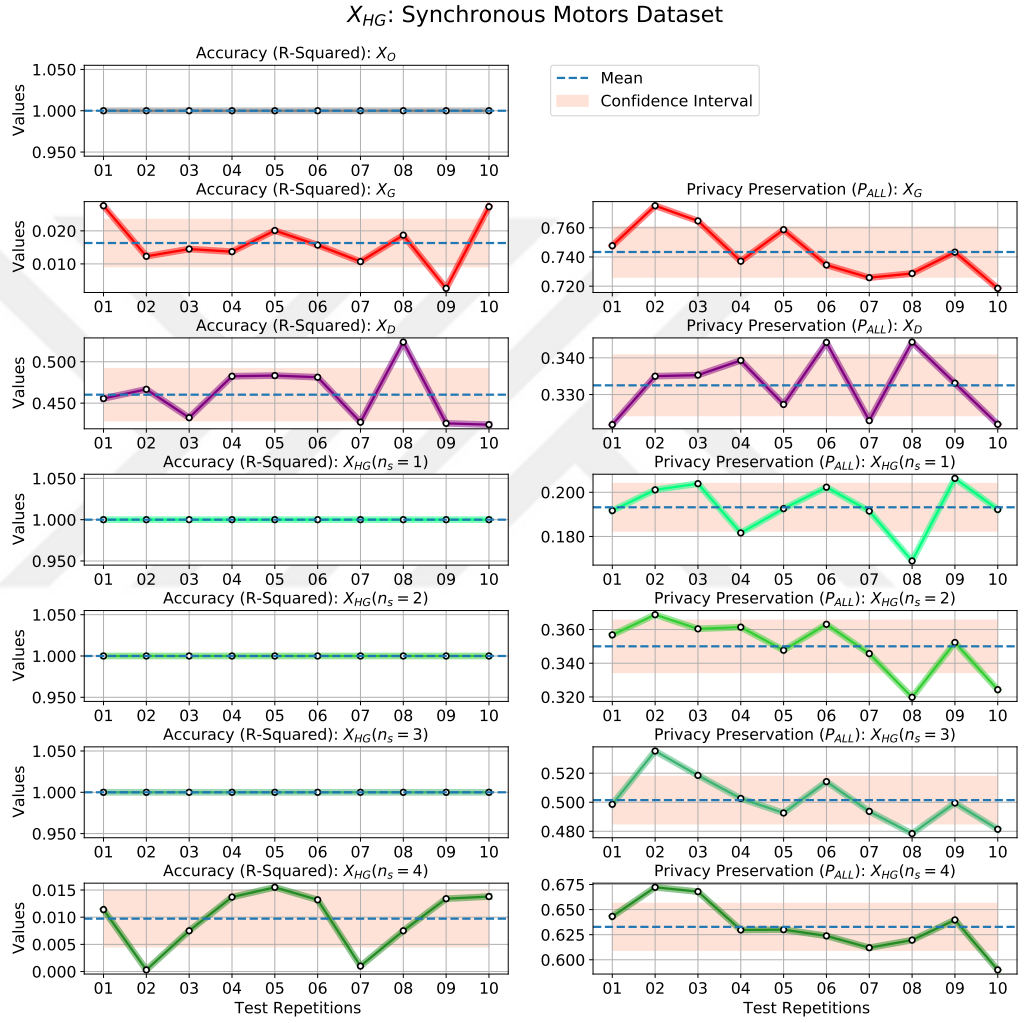


Figure C.8 Variations Among 10 Tests (X_{HG}) - Synchronous Motors Dataset

Table C.15 Experiments in Chapter 4.7 - GAN Only - Synchronous Motors

Proposed Method GAN Only on Synchronous Motors Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	1.0	0.02760	0.45560	1.0	1.0	1.0	0.01140	01
R -Squared	1.0	0.01230	0.46670	1.0	1.0	1.0	0.00030	02
R -Squared	1.0	0.01450	0.43240	1.0	1.0	1.0	0.00750	03
R -Squared	1.0	0.01370	0.48250	1.0	1.0	1.0	0.01370	04
R -Squared	1.0	0.02010	0.48330	1.0	1.0	1.0	0.01550	05
R -Squared	1.0	0.01570	0.48130	1.0	1.0	1.0	0.01320	06
R -Squared	1.0	0.01070	0.42700	1.0	1.0	1.0	0.00100	07
R -Squared	1.0	0.01870	0.52380	1.0	1.0	1.0	0.00750	08
R -Squared	1.0	0.00260	0.42560	1.0	1.0	1.0	0.01340	09
R -Squared	1.0	0.02730	0.42400	1.0	1.0	1.0	0.01380	10
R -Squared	1.0	0.01632	0.46022	1.0	1.0	1.0	0.00973	Mean
Proposed Method GAN Only on Synchronous Motors Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.71970	0.47700	0.15900	0.33470	0.47700	0.59410	01
P_{MSE}	0.0	0.76790	0.46410	0.17720	0.34600	0.52320	0.65820	02
P_{MSE}	0.0	0.76990	0.46030	0.16740	0.32640	0.51050	0.66110	03
P_{MSE}	0.0	0.72870	0.45340	0.15380	0.34010	0.51010	0.61540	04
P_{MSE}	0.0	0.72440	0.45670	0.16540	0.32280	0.48030	0.59840	05
P_{MSE}	0.0	0.74260	0.48100	0.16880	0.33760	0.50630	0.62450	06
P_{MSE}	0.0	0.69600	0.45600	0.16800	0.31200	0.45600	0.57600	07
P_{MSE}	0.0	0.70340	0.49150	0.14410	0.29660	0.45760	0.58470	08
P_{MSE}	0.0	0.67700	0.44360	0.17120	0.31910	0.45910	0.56810	09
P_{MSE}	0.0	0.64800	0.45600	0.14400	0.28000	0.43200	0.54400	10
P_{MSE}	0.0	0.71776	0.46396	0.16189	0.32153	0.48121	0.60245	Mean
Proposed Method GAN Only on Synchronous Motors Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.10760	0.20510	0.3032	0.40220	01
P_{VAR}	0.0	0.55430	0.18330	0.11810	0.22770	0.3357	0.45130	02
P_{VAR}	0.0	0.53210	0.18390	0.11370	0.20670	0.3102	0.42240	03
P_{VAR}	0.0	0.52760	0.18260	0.10240	0.21490	0.3182	0.42260	04
P_{VAR}	0.0	0.55830	0.18440	0.11270	0.22480	0.3365	0.45020	05
P_{VAR}	0.0	0.55590	0.18360	0.10640	0.21540	0.3309	0.44440	06
P_{VAR}	0.0	0.51540	0.18260	0.09530	0.19320	0.2966	0.40450	07
P_{VAR}	0.0	0.52320	0.18380	0.09190	0.19390	0.3020	0.41820	08
P_{VAR}	0.0	0.55810	0.18320	0.12210	0.23060	0.3310	0.43850	09
P_{VAR}	0.0	0.51130	0.18340	0.09800	0.19660	0.3027	0.41230	10
P_{VAR}	0.0	0.53506	0.18342	0.10682	0.21089	0.3167	0.42666	Mean
Proposed Method GAN Only on Synchronous Motors Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.63530	0.35320	0.18820	0.39970	0.46760	0.65080	01
P_{HAUS}	0.0	0.55950	0.43530	0.16650	0.32470	0.41120	0.57670	02
P_{HAUS}	0.0	0.55380	0.43790	0.20010	0.32090	0.39970	0.62570	03
P_{HAUS}	0.0	0.42190	0.42220	0.14830	0.30190	0.32220	0.47490	04
P_{HAUS}	0.0	0.55260	0.35430	0.16540	0.26940	0.29300	0.48180	05
P_{HAUS}	0.0	0.42490	0.43450	0.22660	0.36660	0.38090	0.40320	06
P_{HAUS}	0.0	0.47330	0.39010	0.25490	0.40600	0.43780	0.49690	07
P_{HAUS}	0.0	0.50930	0.43720	0.16190	0.26380	0.37030	0.54820	08
P_{HAUS}	0.0	0.51960	0.39280	0.18430	0.26620	0.34840	0.57020	09
P_{HAUS}	0.0	0.58950	0.37890	0.25840	0.30110	0.41100	0.41840	10
P_{HAUS}	0.0	0.52397	0.40364	0.19546	0.32203	0.38421	0.52468	Mean

Table C.16 Experiments in Chapter 4.7 - GAN Only - Synchronous Motors

Proposed Method GAN Only on Synchronous Motors Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.14600	0.27300	0.40000	0.55870	01
P_{KL}	0.0	0.77810	0.27010	0.18010	0.34730	0.54660	0.65590	02
P_{KL}	0.0	0.74250	0.28760	0.18060	0.37460	0.51510	0.62210	03
P_{KL}	0.0	0.76160	0.30460	0.14570	0.36420	0.49670	0.60260	04
P_{KL}	0.0	0.72480	0.30200	0.14770	0.30870	0.46310	0.57050	05
P_{KL}	0.0	0.68850	0.30160	0.17700	0.31480	0.47210	0.58360	06
P_{KL}	0.0	0.67520	0.27650	0.12860	0.26370	0.41800	0.52090	07
P_{KL}	0.0	0.67740	0.29030	0.12260	0.27740	0.41940	0.52900	08
P_{KL}	0.0	0.71520	0.29800	0.17880	0.32450	0.47680	0.58280	09
P_{KL}	0.0	0.59690	0.27080	0.14150	0.26460	0.40000	0.50460	10
P_{KL}	0.0	0.70205	0.28745	0.15486	0.31128	0.46078	0.57307	Mean
Proposed Method GAN Only on Synchronous Motors Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.35650	0.54910	0.7742	0.88090	01
P_{PRO}	0.0	0.99820	0.41830	0.35580	0.56160	0.7884	0.89330	02
P_{PRO}	0.0	0.99850	0.40750	0.35750	0.53110	0.7767	0.88900	03
P_{PRO}	0.0	0.99750	0.42730	0.35330	0.56180	0.7828	0.88930	04
P_{PRO}	0.0	0.99790	0.42730	0.37680	0.56910	0.7918	0.89460	05
P_{PRO}	0.0	0.99480	0.42480	0.32870	0.54790	0.7832	0.88510	06
P_{PRO}	0.0	0.99750	0.40320	0.30310	0.50440	0.7577	0.87580	07
P_{PRO}	0.0	0.99550	0.42320	0.32030	0.51850	0.7599	0.87610	08
P_{PRO}	0.0	0.99700	0.43450	0.37870	0.57410	0.7899	0.89220	09
P_{PRO}	0.0	0.99620	0.40890	0.31360	0.51680	0.7604	0.87640	10
P_{PRO}	0.0	0.99706	0.41868	0.34443	0.54344	0.7765	0.88527	Mean
Proposed Method GAN Only on Synchronous Motors Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.1926	0.37920	0.5698	0.7724	01
P_{PCC}	0.0	0.99250	0.23870	0.2089	0.40580	0.6067	0.7976	02
P_{PCC}	0.0	0.99140	0.23430	0.2043	0.40260	0.5988	0.7871	03
P_{PCC}	0.0	0.98510	0.24580	0.1858	0.38570	0.5855	0.7733	04
P_{PCC}	0.0	0.99410	0.23910	0.1877	0.39130	0.5909	0.7846	05
P_{PCC}	0.0	1.00000	0.23960	0.2059	0.39600	0.6119	0.8020	06
P_{PCC}	0.0	0.99710	0.22930	0.1994	0.39490	0.5963	0.7977	07
P_{PCC}	0.0	0.96320	0.23980	0.1724	0.36860	0.5609	0.7611	08
P_{PCC}	0.0	0.99300	0.24630	0.2026	0.39920	0.5919	0.7865	09
P_{PCC}	0.0	0.96920	0.23330	0.1974	0.38690	0.5823	0.7837	10
P_{PCC}	0.0	0.98446	0.23785	0.1957	0.39102	0.5895	0.7846	Mean

X_{HD} : Wind Turbine Dataset

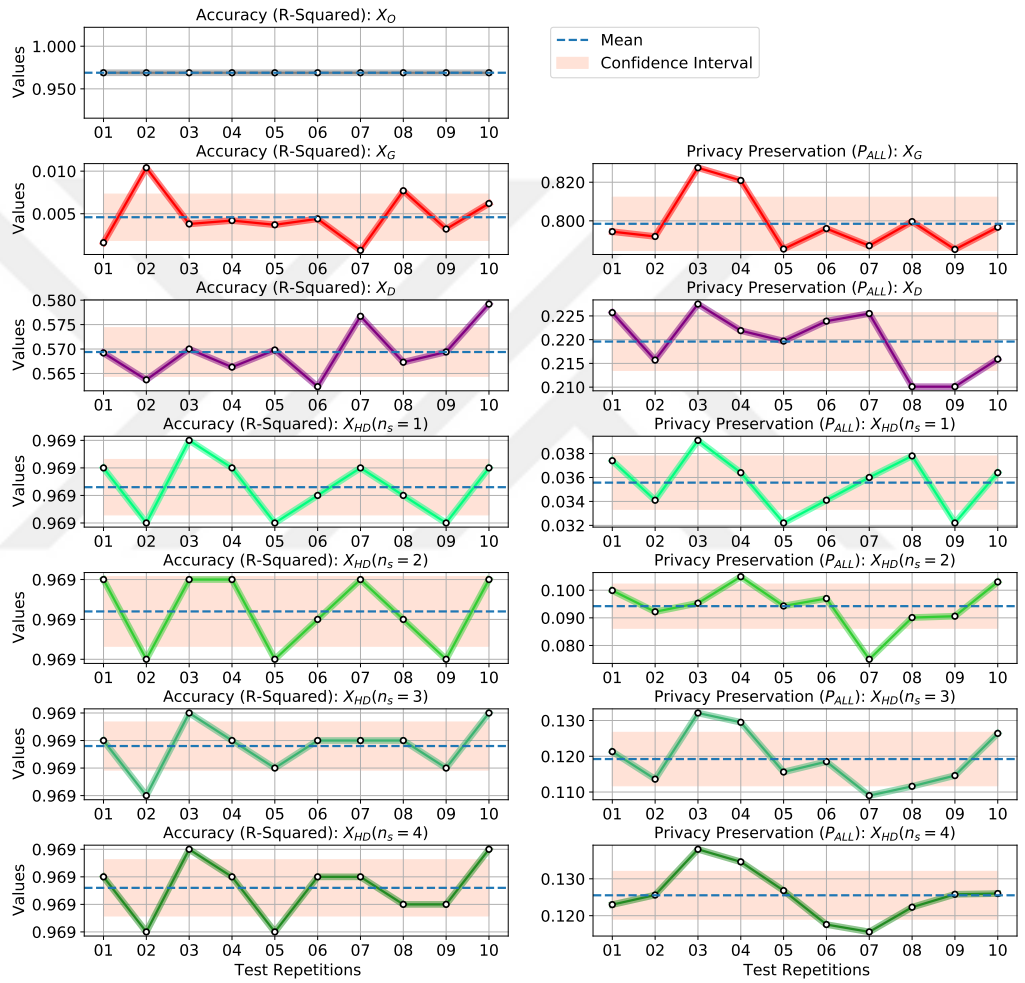


Figure C.9 Variations Among 10 Tests (X_{HD}) - Wind Turbine Dataset

Table C.17 Experiments in Chapter 4.7 - DP Only - Wind Turbine

Proposed Method DP Only on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.9689	0.00160	0.56920	0.96880	0.96880	0.96870	0.96870	01
<i>R-Squared</i>	0.9689	0.01040	0.56370	0.96860	0.96860	0.96850	0.96850	02
<i>R-Squared</i>	0.9689	0.00380	0.57000	0.96890	0.96880	0.96880	0.96880	03
<i>R-Squared</i>	0.9689	0.00420	0.56630	0.96880	0.96880	0.96870	0.96870	04
<i>R-Squared</i>	0.9689	0.00370	0.56980	0.96860	0.96860	0.96860	0.96850	05
<i>R-Squared</i>	0.9689	0.00440	0.56230	0.96870	0.96870	0.96870	0.96870	06
<i>R-Squared</i>	0.9689	0.00070	0.57670	0.96880	0.96880	0.96870	0.96870	07
<i>R-Squared</i>	0.9689	0.00770	0.56730	0.96870	0.96870	0.96870	0.96860	08
<i>R-Squared</i>	0.9689	0.00320	0.56940	0.96860	0.96860	0.96860	0.96860	09
<i>R-Squared</i>	0.9689	0.00620	0.57920	0.96880	0.96880	0.96880	0.96880	10
<i>R-Squared</i>	0.9689	0.00459	0.56939	0.96873	0.96872	0.96868	0.96866	Mean
Proposed Method DP Only on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.75310	0.22720	0.00590	0.04180	0.0598	0.07780	01
P_{MSE}	0.0	0.82570	0.22910	0.00590	0.04220	0.0603	0.07850	02
P_{MSE}	0.0	0.85480	0.22720	0.00590	0.04180	0.0598	0.07780	03
P_{MSE}	0.0	0.82150	0.21420	0.00570	0.04050	0.0579	0.07530	04
P_{MSE}	0.0	0.69170	0.21380	0.00550	0.03940	0.0563	0.07320	05
P_{MSE}	0.0	0.78950	0.22910	0.00590	0.04220	0.0603	0.07850	06
P_{MSE}	0.0	0.70280	0.21720	0.00560	0.04000	0.0572	0.07440	07
P_{MSE}	0.0	0.82920	0.22420	0.00590	0.04240	0.0606	0.07880	08
P_{MSE}	0.0	0.73390	0.21130	0.00540	0.03890	0.0556	0.07240	09
P_{MSE}	0.0	0.76000	0.21720	0.00560	0.04000	0.0572	0.07440	10
P_{MSE}	0.0	0.77622	0.22105	0.00573	0.04092	0.0585	0.07611	Mean
Proposed Method DP Only on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10250	0.0036	0.0249	0.05670	0.06280	01
P_{VAR}	0.0	0.63820	0.10240	0.0036	0.0249	0.05660	0.06270	02
P_{VAR}	0.0	0.66980	0.10250	0.0036	0.0249	0.05670	0.06280	03
P_{VAR}	0.0	0.70830	0.10230	0.0036	0.0249	0.05660	0.06270	04
P_{VAR}	0.0	0.58470	0.10250	0.0036	0.0249	0.05670	0.06280	05
P_{VAR}	0.0	0.60790	0.10230	0.0036	0.0249	0.05660	0.06270	06
P_{VAR}	0.0	0.59730	0.10230	0.0036	0.0249	0.05660	0.06270	07
P_{VAR}	0.0	0.66230	0.10240	0.0036	0.0249	0.05660	0.06280	08
P_{VAR}	0.0	0.62930	0.10240	0.0036	0.0249	0.05660	0.06270	09
P_{VAR}	0.0	0.65040	0.10250	0.0036	0.0249	0.05670	0.06280	10
P_{VAR}	0.0	0.63441	0.10241	0.0036	0.0249	0.05664	0.06275	Mean
Proposed Method DP Only on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.18190	0.38460	0.38460	0.31010	01
P_{HAUS}	0.0	0.36360	0.32540	0.16030	0.33550	0.33550	0.32210	02
P_{HAUS}	0.0	0.47530	0.39450	0.19060	0.35420	0.44510	0.39450	03
P_{HAUS}	0.0	0.41860	0.38010	0.17060	0.40880	0.42710	0.37340	04
P_{HAUS}	0.0	0.50480	0.36700	0.14970	0.35040	0.35040	0.33240	05
P_{HAUS}	0.0	0.46330	0.37260	0.16100	0.36550	0.36550	0.27450	06
P_{HAUS}	0.0	0.52240	0.40400	0.17350	0.23800	0.31380	0.26950	07
P_{HAUS}	0.0	0.37270	0.30360	0.18360	0.32420	0.32420	0.30400	08
P_{HAUS}	0.0	0.37460	0.31100	0.15020	0.33110	0.34710	0.32970	09
P_{HAUS}	0.0	0.47310	0.36060	0.17640	0.40790	0.42060	0.33570	10
P_{HAUS}	0.0	0.44679	0.36225	0.16978	0.35002	0.37139	0.32459	Mean

Table C.18 Experiments in Chapter 4.7 - DP Only - Wind Turbine

Proposed Method DP Only on Wind Turbine Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.92060	0.22220	0.01810	0.0498	0.06790	0.0952	01
P_{KL}	0.0	0.93250	0.23440	0.01830	0.0505	0.06880	0.0965	02
P_{KL}	0.0	0.96520	0.23880	0.01910	0.0525	0.07160	0.1003	03
P_{KL}	0.0	0.97910	0.23640	0.02350	0.0566	0.07580	0.1040	04
P_{KL}	0.0	0.94430	0.23490	0.01910	0.0527	0.07180	0.1007	05
P_{KL}	0.0	0.92260	0.23900	0.01870	0.0515	0.07020	0.0984	06
P_{KL}	0.0	0.90030	0.22960	0.01830	0.0505	0.06880	0.0965	07
P_{KL}	0.0	0.94940	0.23030	0.01840	0.0506	0.06900	0.0968	08
P_{KL}	0.0	0.97910	0.23640	0.01890	0.0520	0.07090	0.0993	09
P_{KL}	0.0	0.89660	0.21540	0.01750	0.0483	0.06580	0.0923	10
P_{KL}	0.0	0.93897	0.23174	0.01899	0.0515	0.07006	0.0980	Mean
Proposed Method DP Only on Wind Turbine Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.0095	0.0628	0.08520	0.10680	01
P_{PRO}	0.0	1.0	0.25540	0.0095	0.0630	0.08530	0.10690	02
P_{PRO}	0.0	1.0	0.25470	0.0095	0.0627	0.08510	0.10670	03
P_{PRO}	0.0	1.0	0.25300	0.0095	0.0627	0.08520	0.10650	04
P_{PRO}	0.0	1.0	0.25500	0.0095	0.0629	0.08520	0.10700	05
P_{PRO}	0.0	1.0	0.25460	0.0095	0.0628	0.08510	0.10690	06
P_{PRO}	0.0	1.0	0.25440	0.0095	0.0627	0.08500	0.10640	07
P_{PRO}	0.0	1.0	0.25450	0.0095	0.0629	0.08520	0.10660	08
P_{PRO}	0.0	1.0	0.25460	0.0095	0.0627	0.08500	0.10670	09
P_{PRO}	0.0	1.0	0.25430	0.0095	0.0628	0.08510	0.10670	10
P_{PRO}	0.0	1.0	0.25444	0.0095	0.0628	0.08514	0.10672	Mean
Proposed Method DP Only on Wind Turbine Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.99700	0.14460	0.00570	0.03540	0.07380	0.08510	01
P_{PCC}	0.0	0.99160	0.14780	0.00710	0.03690	0.07530	0.08660	02
P_{PCC}	0.0	1.00000	0.14730	0.00570	0.03580	0.07440	0.08580	03
P_{PCC}	0.0	0.99770	0.14560	0.00570	0.03570	0.07420	0.08560	04
P_{PCC}	0.0	0.98680	0.14540	0.00560	0.03530	0.07340	0.08470	05
P_{PCC}	0.0	0.99300	0.14560	0.00560	0.03530	0.07360	0.08490	06
P_{PCC}	0.0	1.00000	0.14530	0.00570	0.03420	0.07270	0.08410	07
P_{PCC}	0.0	0.98390	0.14580	0.00560	0.03540	0.07360	0.08490	08
P_{PCC}	0.0	0.99440	0.14470	0.00570	0.03410	0.07240	0.08370	09
P_{PCC}	0.0	1.00000	0.14530	0.00570	0.03420	0.07270	0.08410	10
P_{PCC}	0.0	0.99444	0.14574	0.00581	0.03523	0.07361	0.08495	Mean

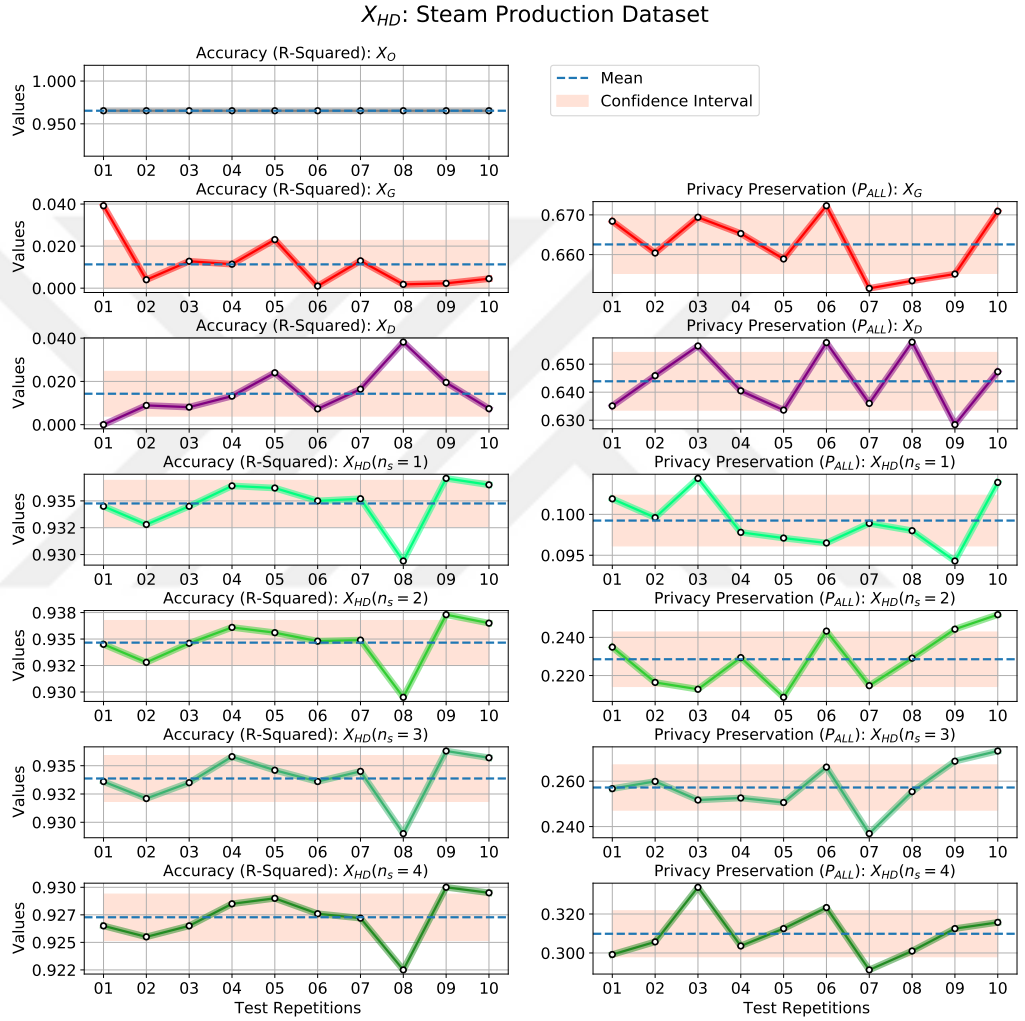


Figure C.10 Variations Among 10 Tests (X_{HD}) - Steam Production Dataset

Table C.19 Experiments in Chapter 4.7 - DP Only - Steam Production

Proposed Method DP Only on Steam Production Dataset: R -Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
R -Squared	0.9653	0.03920	0.0000	0.93450	0.93450	0.93360	0.92650	01
R -Squared	0.9653	0.00400	0.0089	0.93280	0.93280	0.93210	0.92550	02
R -Squared	0.9653	0.01280	0.0081	0.93450	0.93460	0.93350	0.92650	03
R -Squared	0.9653	0.01140	0.0132	0.93640	0.93610	0.93580	0.92850	04
R -Squared	0.9653	0.02310	0.0240	0.93620	0.93560	0.93460	0.92900	05
R -Squared	0.9653	0.00100	0.0073	0.93500	0.93480	0.93360	0.92760	06
R -Squared	0.9653	0.01300	0.0164	0.93520	0.93490	0.93450	0.92720	07
R -Squared	0.9653	0.00180	0.0382	0.92940	0.92950	0.92900	0.92250	08
R -Squared	0.9653	0.00230	0.0195	0.93710	0.93730	0.93630	0.93000	09
R -Squared	0.9653	0.00450	0.0074	0.93650	0.93650	0.93570	0.92950	10
R -Squared	0.9653	0.01131	0.0143	0.93476	0.93466	0.93387	0.92728	Mean
Proposed Method DP Only on Steam Production Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.30040	0.5477	0.02680	0.09500	0.12180	0.18620	01
P_{MSE}	0.0	0.29920	0.5409	0.02320	0.08820	0.11520	0.18020	02
P_{MSE}	0.0	0.28540	0.5402	0.02300	0.09120	0.11800	0.18240	03
P_{MSE}	0.0	0.27980	0.5154	0.02230	0.08460	0.11050	0.17290	04
P_{MSE}	0.0	0.29330	0.4976	0.02170	0.08230	0.10750	0.16810	05
P_{MSE}	0.0	0.31050	0.5447	0.02320	0.09200	0.11900	0.18400	06
P_{MSE}	0.0	0.27640	0.5164	0.02200	0.08720	0.11280	0.17440	07
P_{MSE}	0.0	0.27750	0.5432	0.02330	0.09240	0.11950	0.18470	08
P_{MSE}	0.0	0.28990	0.4953	0.02140	0.08480	0.10970	0.16960	09
P_{MSE}	0.0	0.28720	0.5056	0.02200	0.08360	0.10920	0.17080	10
P_{MSE}	0.0	0.28996	0.5247	0.02289	0.08813	0.11432	0.17733	Mean
Proposed Method DP Only on Steam Production Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.04800	0.26020	0.27310	0.30360	01
P_{VAR}	0.0	0.65740	1.0	0.04800	0.25950	0.27210	0.30270	02
P_{VAR}	0.0	0.63460	1.0	0.04800	0.26020	0.27290	0.30340	03
P_{VAR}	0.0	0.66250	1.0	0.04790	0.26000	0.27280	0.30330	04
P_{VAR}	0.0	0.68900	1.0	0.04800	0.25980	0.27260	0.30320	05
P_{VAR}	0.0	0.64830	1.0	0.04790	0.26050	0.27330	0.30380	06
P_{VAR}	0.0	0.66160	1.0	0.04790	0.26020	0.27280	0.30330	07
P_{VAR}	0.0	0.63930	1.0	0.04800	0.25960	0.27250	0.30300	08
P_{VAR}	0.0	0.67320	1.0	0.04790	0.26030	0.27320	0.30370	09
P_{VAR}	0.0	0.64810	1.0	0.04800	0.26000	0.27290	0.30340	10
P_{VAR}	0.0	0.65727	1.0	0.04796	0.26003	0.27282	0.30334	Mean
Proposed Method DP Only on Steam Production Dataset: P_{HAUS} values								
Method	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.38260	0.61780	0.61780	0.61780	01
P_{HAUS}	0.0	0.70670	0.92660	0.37440	0.51710	0.64130	0.65460	02
P_{HAUS}	0.0	0.80610	0.98410	0.40450	0.49470	0.59600	0.82760	03
P_{HAUS}	0.0	0.73450	0.92090	0.36420	0.59600	0.60080	0.65050	04
P_{HAUS}	0.0	0.64090	0.88860	0.36020	0.47900	0.59720	0.71020	05
P_{HAUS}	0.0	0.75500	0.98880	0.35440	0.66910	0.66910	0.75460	06
P_{HAUS}	0.0	0.67750	0.90110	0.37220	0.51060	0.51400	0.58350	07
P_{HAUS}	0.0	0.71480	1.00000	0.36360	0.58910	0.61240	0.62290	08
P_{HAUS}	0.0	0.63940	0.87210	0.34940	0.69270	0.70670	0.71140	09
P_{HAUS}	0.0	0.79240	0.98850	0.40220	0.73680	0.73680	0.73680	10
P_{HAUS}	0.0	0.72235	0.93403	0.37277	0.59029	0.62921	0.68699	Mean

Table C.20 Experiments in Chapter 4.7 - DP Only - Steam Production

Proposed Method DP Only on Steam Production Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.01750	0.0432	0.0635	0.09240	01
P_{KL}	0.0	0.30100	0.25430	0.01770	0.0437	0.0643	0.09360	02
P_{KL}	0.0	0.29800	0.26450	0.01840	0.0455	0.0669	0.09730	03
P_{KL}	0.0	0.31620	0.26190	0.01820	0.0450	0.0662	0.09640	04
P_{KL}	0.0	0.34160	0.26850	0.01850	0.0456	0.0671	0.09770	05
P_{KL}	0.0	0.33080	0.26230	0.01800	0.0475	0.0685	0.09840	06
P_{KL}	0.0	0.29520	0.25720	0.01770	0.0437	0.0643	0.09360	07
P_{KL}	0.0	0.29030	0.25810	0.01770	0.0439	0.0645	0.09390	08
P_{KL}	0.0	0.33710	0.25890	0.01490	0.0421	0.0632	0.09340	09
P_{KL}	0.0	0.30220	0.23780	0.01690	0.0418	0.0615	0.08950	10
P_{KL}	0.0	0.31095	0.25746	0.01755	0.0442	0.0650	0.09462	Mean
Proposed Method DP Only on Steam Production Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.09600	0.28930	0.34290	0.44430	01
P_{PRO}	0.0	0.99820	0.70340	0.09550	0.28750	0.34510	0.45030	02
P_{PRO}	0.0	0.99830	0.70060	0.09500	0.28600	0.33970	0.44410	03
P_{PRO}	0.0	0.99900	0.69780	0.09540	0.28840	0.34430	0.44720	04
P_{PRO}	0.0	0.99870	0.70310	0.09570	0.28610	0.34220	0.44770	05
P_{PRO}	0.0	0.99900	0.70190	0.09590	0.28930	0.34680	0.44850	06
P_{PRO}	0.0	0.99840	0.69810	0.09540	0.28700	0.34070	0.44500	07
P_{PRO}	0.0	0.99850	0.70110	0.09580	0.28800	0.34400	0.45020	08
P_{PRO}	0.0	0.99830	0.70100	0.09500	0.28670	0.34230	0.44800	09
P_{PRO}	0.0	0.99830	0.70120	0.09550	0.28750	0.34050	0.44430	10
P_{PRO}	0.0	0.99848	0.70044	0.09552	0.28758	0.34285	0.44696	Mean
Proposed Method DP Only on Steam Production Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	1.0000	0.44580	0.04060	0.10370	0.12090	0.15070	01
P_{PCC}	0.0	1.0000	0.45020	0.03890	0.10210	0.12110	0.15280	02
P_{PCC}	0.0	0.9941	0.44970	0.03740	0.09920	0.11660	0.14750	03
P_{PCC}	0.0	1.0000	0.44690	0.03910	0.10260	0.12080	0.15170	04
P_{PCC}	0.0	0.9899	0.44380	0.03860	0.09880	0.11680	0.14820	05
P_{PCC}	0.0	0.9901	0.44820	0.03960	0.10170	0.12060	0.15030	06
P_{PCC}	0.0	0.9998	0.44320	0.03810	0.09970	0.11700	0.14780	07
P_{PCC}	0.0	1.0000	0.44510	0.03960	0.10180	0.11980	0.15130	08
P_{PCC}	0.0	0.9930	0.44330	0.03700	0.09930	0.11740	0.14900	09
P_{PCC}	0.0	0.9971	0.45050	0.03900	0.10150	0.11880	0.14960	10
P_{PCC}	0.0	0.9964	0.44667	0.03879	0.10104	0.11898	0.14989	Mean

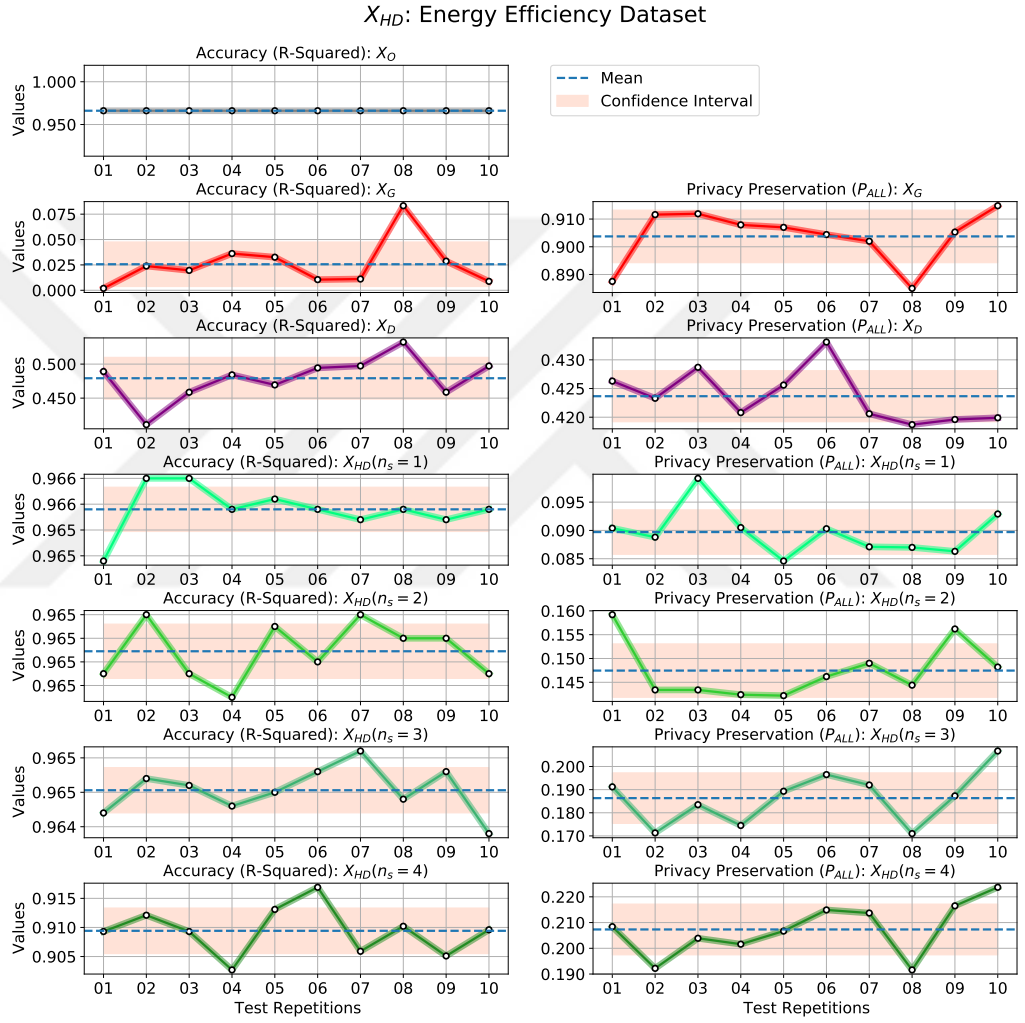


Figure C.11 Variations Among 10 Tests (X_{HD}) - Energy Efficiency Dataset

Table C.21 Experiments in Chapter 4.7 - DP Only - Energy Efficiency

Proposed Method DP Only on Energy Efficiency Dataset: R -Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
R -Squared	0.9661	0.00190	0.48910	0.9652	0.96490	0.96470	0.90930	01
R -Squared	0.9661	0.02390	0.41100	0.9660	0.96540	0.96520	0.91210	02
R -Squared	0.9661	0.01960	0.45860	0.9660	0.96490	0.96510	0.90930	03
R -Squared	0.9661	0.03620	0.48420	0.9657	0.96470	0.96480	0.90270	04
R -Squared	0.9661	0.03260	0.46940	0.9658	0.96530	0.96500	0.91310	05
R -Squared	0.9661	0.01050	0.49440	0.9657	0.96500	0.96530	0.91690	06
R -Squared	0.9661	0.01110	0.49730	0.9656	0.96540	0.96560	0.90590	07
R -Squared	0.9661	0.08330	0.53240	0.9657	0.96520	0.96490	0.91020	08
R -Squared	0.9661	0.02870	0.45870	0.9656	0.96520	0.96530	0.90510	09
R -Squared	0.9661	0.00880	0.49720	0.9657	0.96490	0.96440	0.90960	10
R -Squared	0.9661	0.02566	0.47923	0.9657	0.96509	0.96503	0.90942	Mean
Proposed Method DP Only on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	1.0	0.53140	0.03350	0.10460	0.15480	0.1841	01
P_{MSE}	0.0	1.0	0.51900	0.03380	0.10130	0.15190	0.1772	02
P_{MSE}	0.0	1.0	0.52300	0.03350	0.09620	0.15060	0.1799	03
P_{MSE}	0.0	1.0	0.49390	0.03240	0.09720	0.14980	0.1781	04
P_{MSE}	0.0	1.0	0.50000	0.03150	0.09450	0.14570	0.1732	05
P_{MSE}	0.0	1.0	0.54010	0.03800	0.10970	0.16460	0.1941	06
P_{MSE}	0.0	1.0	0.48800	0.03200	0.09600	0.14800	0.1720	07
P_{MSE}	0.0	1.0	0.52970	0.03390	0.10590	0.15680	0.1822	08
P_{MSE}	0.0	1.0	0.48250	0.03110	0.09340	0.14400	0.1712	09
P_{MSE}	0.0	1.0	0.50000	0.03200	0.09600	0.14800	0.1760	10
P_{MSE}	0.0	1.0	0.51076	0.03317	0.09948	0.15142	0.1788	Mean
Proposed Method DP Only on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.01190	0.03260	0.04400	0.05710	01
P_{VAR}	0.0	0.52300	0.15090	0.01190	0.03260	0.04390	0.05700	02
P_{VAR}	0.0	0.52600	0.15130	0.01190	0.03260	0.04400	0.05710	03
P_{VAR}	0.0	0.53810	0.15030	0.01160	0.03230	0.04360	0.05670	04
P_{VAR}	0.0	0.55510	0.15130	0.01190	0.03260	0.04400	0.05710	05
P_{VAR}	0.0	0.52160	0.15060	0.01160	0.03230	0.04360	0.05680	06
P_{VAR}	0.0	0.53130	0.15080	0.01190	0.03250	0.04390	0.05700	07
P_{VAR}	0.0	0.51010	0.15130	0.01160	0.03260	0.04390	0.05710	08
P_{VAR}	0.0	0.56190	0.15090	0.01160	0.03230	0.04360	0.05680	09
P_{VAR}	0.0	0.54090	0.15110	0.01160	0.03230	0.04370	0.05680	10
P_{VAR}	0.0	0.53234	0.15096	0.01175	0.03247	0.04382	0.05695	Mean
Proposed Method DP Only on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.42120	0.62210	0.65830	0.6583	01
P_{HAUS}	0.0	0.97250	1.00000	0.41080	0.53360	0.54410	0.5669	02
P_{HAUS}	0.0	0.95840	1.00000	0.46790	0.53240	0.60780	0.6249	03
P_{HAUS}	0.0	0.91530	1.00000	0.41750	0.52020	0.54950	0.6069	04
P_{HAUS}	0.0	0.89050	1.00000	0.38230	0.52420	0.64370	0.6437	05
P_{HAUS}	0.0	0.92740	1.00000	0.41140	0.53190	0.66780	0.6678	06
P_{HAUS}	0.0	0.89190	1.00000	0.40210	0.57130	0.66270	0.6978	07
P_{HAUS}	0.0	0.83290	0.94430	0.40040	0.53160	0.53480	0.5569	08
P_{HAUS}	0.0	0.87520	1.00000	0.39780	0.61680	0.64210	0.7148	09
P_{HAUS}	0.0	0.95280	1.00000	0.43770	0.56460	0.75900	0.7590	10
P_{HAUS}	0.0	0.90552	0.99443	0.41491	0.55487	0.62698	0.6497	Mean

Table C.22 Experiments in Chapter 4.7 - DP Only - Energy Efficiency

Proposed Method DP Only on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.02540	0.06030	0.09520	0.11750	01
P_{KL}	0.0	1.0	0.32150	0.02570	0.05790	0.09320	0.11580	02
P_{KL}	0.0	1.0	0.34450	0.03010	0.06350	0.10370	0.12710	03
P_{KL}	0.0	1.0	0.33110	0.02980	0.06620	0.10600	0.12910	04
P_{KL}	0.0	1.0	0.34560	0.03020	0.06380	0.10400	0.12750	05
P_{KL}	0.0	1.0	0.34100	0.02950	0.06560	0.10490	0.13110	06
P_{KL}	0.0	1.0	0.32800	0.02570	0.05790	0.09970	0.11900	07
P_{KL}	0.0	1.0	0.33230	0.02580	0.06130	0.09680	0.11940	08
P_{KL}	0.0	1.0	0.33110	0.02650	0.05960	0.09930	0.12250	09
P_{KL}	0.0	1.0	0.31690	0.02460	0.05850	0.09230	0.11380	10
P_{KL}	0.0	1.0	0.33126	0.02733	0.06146	0.09951	0.12228	Mean
Proposed Method DP Only on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.03380	0.09370	0.13620	0.15720	01
P_{PRO}	0.0	0.9946	0.34650	0.03380	0.09360	0.13570	0.15770	02
P_{PRO}	0.0	0.9948	0.34910	0.03400	0.09380	0.13600	0.15710	03
P_{PRO}	0.0	0.9949	0.34690	0.03350	0.09430	0.13730	0.15880	04
P_{PRO}	0.0	0.9966	0.35290	0.03380	0.09470	0.13810	0.15960	05
P_{PRO}	0.0	0.9944	0.35920	0.03350	0.09420	0.13760	0.15970	06
P_{PRO}	0.0	0.9944	0.35240	0.03400	0.09410	0.13780	0.15890	07
P_{PRO}	0.0	0.9941	0.35050	0.03350	0.09350	0.13540	0.15670	08
P_{PRO}	0.0	0.9947	0.34950	0.03380	0.09370	0.13620	0.15730	09
P_{PRO}	0.0	0.9972	0.34910	0.03360	0.09390	0.13650	0.15750	10
P_{PRO}	0.0	0.9950	0.35075	0.03373	0.09395	0.13668	0.15805	Mean
Proposed Method DP Only on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.0169	0.04170	0.05860	0.07640	01
P_{PCC}	0.0	0.97960	0.20190	0.0169	0.04180	0.05870	0.07860	02
P_{PCC}	0.0	0.99240	0.20430	0.0180	0.04210	0.05910	0.07710	03
P_{PCC}	0.0	0.99910	0.20280	0.0180	0.04400	0.06090	0.07990	04
P_{PCC}	0.0	1.00000	0.20360	0.0178	0.04350	0.06030	0.07910	05
P_{PCC}	0.0	0.98320	0.20790	0.0178	0.04360	0.06040	0.08020	06
P_{PCC}	0.0	0.99410	0.20440	0.0170	0.04190	0.05980	0.07780	07
P_{PCC}	0.0	0.97310	0.20410	0.0168	0.04160	0.05850	0.07730	08
P_{PCC}	0.0	1.00000	0.20360	0.0169	0.04170	0.05860	0.07650	09
P_{PCC}	0.0	0.99810	0.20240	0.0179	0.04390	0.06080	0.07880	10
P_{PCC}	0.0	0.98965	0.20385	0.0174	0.04258	0.05957	0.07817	Mean

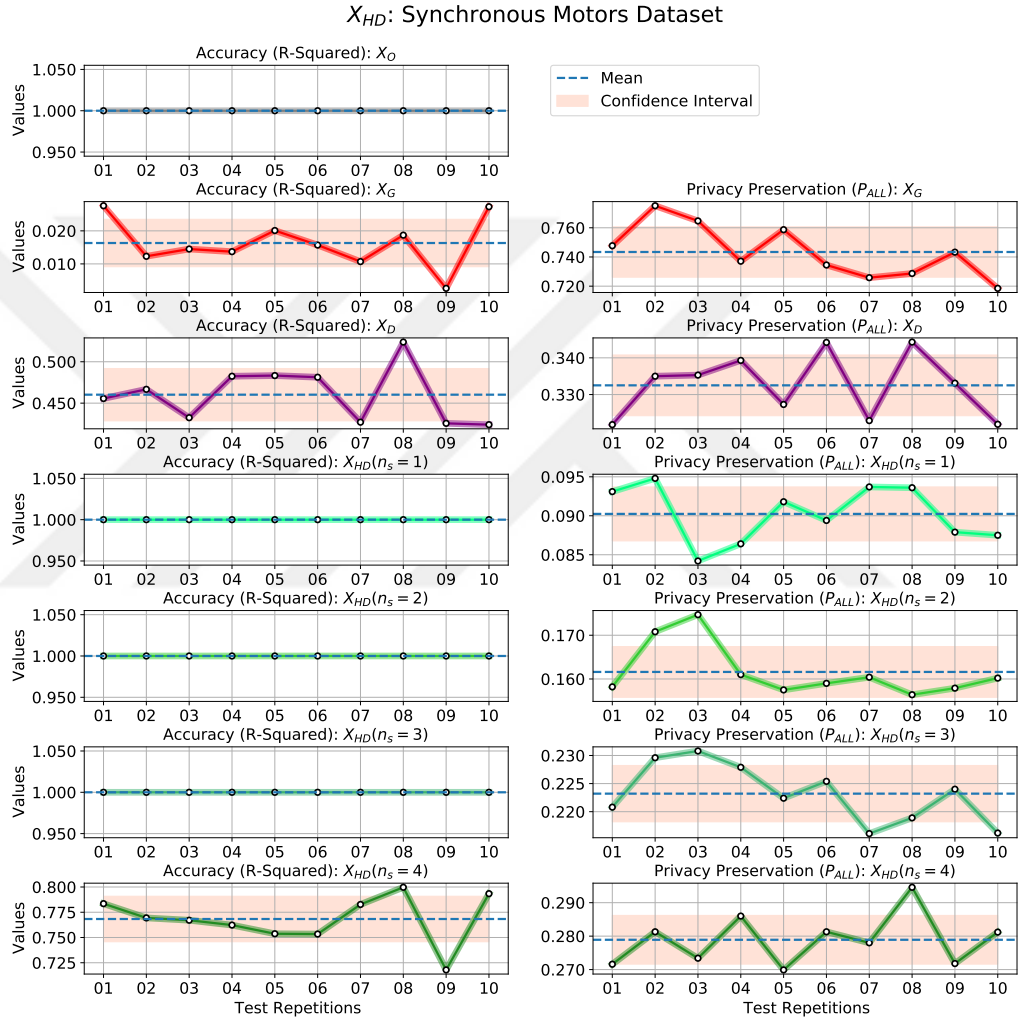


Figure C.12 Variations Among 10 Tests (X_{HD}) - Synchronous Motors Dataset

Table C.23 Experiments in Chapter 4.7 - DP Only - Synchronous Motors

Proposed Method DP Only on Synchronous Motors Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	1.0	0.02760	0.45560	1.0	1.0	1.0	0.78350	01
R -Squared	1.0	0.01230	0.46670	1.0	1.0	1.0	0.76940	02
R -Squared	1.0	0.01450	0.43240	1.0	1.0	1.0	0.76710	03
R -Squared	1.0	0.01370	0.48250	1.0	1.0	1.0	0.76230	04
R -Squared	1.0	0.02010	0.48330	1.0	1.0	1.0	0.75370	05
R -Squared	1.0	0.01570	0.48130	1.0	1.0	1.0	0.75340	06
R -Squared	1.0	0.01070	0.42700	1.0	1.0	1.0	0.78270	07
R -Squared	1.0	0.01870	0.52380	1.0	1.0	1.0	0.79970	08
R -Squared	1.0	0.00260	0.42560	1.0	1.0	1.0	0.71790	09
R -Squared	1.0	0.02730	0.42400	1.0	1.0	1.0	0.79340	10
R -Squared	1.0	0.01632	0.46022	1.0	1.0	1.0	0.76831	Mean
Proposed Method DP Only on Synchronous Motors Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.71970	0.47700	0.11720	0.21760	0.31800	0.39330	01
P_{MSE}	0.0	0.76790	0.46410	0.10130	0.20250	0.31220	0.38820	02
P_{MSE}	0.0	0.76990	0.46030	0.10040	0.20920	0.30960	0.38490	03
P_{MSE}	0.0	0.72870	0.45340	0.09720	0.19430	0.30770	0.38060	04
P_{MSE}	0.0	0.72440	0.45670	0.10240	0.20470	0.30710	0.37800	05
P_{MSE}	0.0	0.74260	0.48100	0.10970	0.21940	0.32910	0.40510	06
P_{MSE}	0.0	0.69600	0.45600	0.10400	0.20800	0.30400	0.37600	07
P_{MSE}	0.0	0.70340	0.49150	0.11020	0.22030	0.33050	0.41530	08
P_{MSE}	0.0	0.67700	0.44360	0.10120	0.20230	0.30350	0.37350	09
P_{MSE}	0.0	0.64800	0.45600	0.10400	0.20800	0.30400	0.38400	10
P_{MSE}	0.0	0.71776	0.46396	0.10476	0.20863	0.31257	0.38789	Mean
Proposed Method DP Only on Synchronous Motors Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.03490	0.07120	0.10710	0.14500	01
P_{VAR}	0.0	0.55430	0.18330	0.03530	0.07120	0.10700	0.14540	02
P_{VAR}	0.0	0.53210	0.18390	0.03540	0.07170	0.10760	0.14550	03
P_{VAR}	0.0	0.52760	0.18260	0.03530	0.07110	0.10640	0.14470	04
P_{VAR}	0.0	0.55830	0.18440	0.03590	0.07170	0.10760	0.14600	05
P_{VAR}	0.0	0.55590	0.18360	0.03530	0.07110	0.10690	0.14530	06
P_{VAR}	0.0	0.51540	0.18260	0.03530	0.07110	0.10640	0.14470	07
P_{VAR}	0.0	0.52320	0.18380	0.03540	0.07120	0.10710	0.14550	08
P_{VAR}	0.0	0.55810	0.18320	0.03530	0.07110	0.10700	0.14480	09
P_{VAR}	0.0	0.51130	0.18340	0.03540	0.07120	0.10710	0.14500	10
P_{VAR}	0.0	0.53506	0.18342	0.03535	0.07126	0.10702	0.14519	Mean
Proposed Method DP Only on Synchronous Motors Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.63530	0.35320	0.15980	0.20240	0.26280	0.31300	01
P_{HAUS}	0.0	0.55950	0.43530	0.18750	0.29140	0.30180	0.35610	02
P_{HAUS}	0.0	0.55380	0.43790	0.12860	0.30990	0.32640	0.32640	03
P_{HAUS}	0.0	0.42190	0.42220	0.13270	0.21810	0.27280	0.36090	04
P_{HAUS}	0.0	0.55260	0.35430	0.16040	0.19460	0.24770	0.27870	05
P_{HAUS}	0.0	0.42490	0.43450	0.14540	0.19840	0.24600	0.31420	06
P_{HAUS}	0.0	0.47330	0.39010	0.17160	0.22740	0.25310	0.37850	07
P_{HAUS}	0.0	0.50930	0.43720	0.15950	0.18620	0.21590	0.39500	08
P_{HAUS}	0.0	0.51960	0.39280	0.13360	0.19420	0.24570	0.28450	09
P_{HAUS}	0.0	0.58950	0.37890	0.13620	0.22470	0.24890	0.37700	10
P_{HAUS}	0.0	0.52397	0.40364	0.15153	0.22473	0.26211	0.33843	Mean

Table C.24 Experiments in Chapter 4.7 - DP Only - Synchronous Motors

Proposed Method DP Only on Synchronous Motors Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.0635	0.12060	0.17780	0.22220	01
P_{KL}	0.0	0.77810	0.27010	0.0579	0.11580	0.18010	0.22510	02
P_{KL}	0.0	0.74250	0.28760	0.0602	0.12710	0.18730	0.23410	03
P_{KL}	0.0	0.76160	0.30460	0.0662	0.13250	0.19870	0.25170	04
P_{KL}	0.0	0.72480	0.30200	0.0671	0.13420	0.20130	0.24830	05
P_{KL}	0.0	0.68850	0.30160	0.0656	0.13110	0.19670	0.24920	06
P_{KL}	0.0	0.67520	0.27650	0.0643	0.12220	0.18010	0.22510	07
P_{KL}	0.0	0.67740	0.29030	0.0645	0.12260	0.18710	0.23870	08
P_{KL}	0.0	0.71520	0.29800	0.0662	0.13250	0.19870	0.24500	09
P_{KL}	0.0	0.59690	0.27080	0.0615	0.12310	0.17850	0.22770	10
P_{KL}	0.0	0.70205	0.28745	0.0637	0.12617	0.18863	0.23671	Mean
Proposed Method DP Only on Synchronous Motors Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.13780	0.24410	0.32200	0.37120	01
P_{PRO}	0.0	0.99820	0.41830	0.13930	0.24620	0.33130	0.37990	02
P_{PRO}	0.0	0.99850	0.40750	0.13650	0.24010	0.31790	0.36510	03
P_{PRO}	0.0	0.99750	0.42730	0.13900	0.25020	0.33370	0.38210	04
P_{PRO}	0.0	0.99790	0.42730	0.13950	0.24660	0.33050	0.38070	05
P_{PRO}	0.0	0.99480	0.42480	0.13700	0.24270	0.33120	0.38180	06
P_{PRO}	0.0	0.99750	0.40320	0.13930	0.24180	0.31730	0.36250	07
P_{PRO}	0.0	0.99550	0.42320	0.14260	0.24470	0.33010	0.38100	08
P_{PRO}	0.0	0.99700	0.43450	0.14140	0.24980	0.33990	0.38810	09
P_{PRO}	0.0	0.99620	0.40890	0.14000	0.24230	0.32140	0.36830	10
P_{PRO}	0.0	0.99706	0.41868	0.13924	0.24485	0.32753	0.37607	Mean
Proposed Method DP Only on Synchronous Motors Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.04570	0.09330	0.13700	0.18470	01
P_{PCC}	0.0	0.99250	0.23870	0.04770	0.09750	0.14520	0.19290	02
P_{PCC}	0.0	0.99140	0.23430	0.04410	0.09010	0.13620	0.18430	03
P_{PCC}	0.0	0.98510	0.24580	0.04800	0.09990	0.14790	0.19580	04
P_{PCC}	0.0	0.99410	0.23910	0.04550	0.09290	0.14030	0.18770	05
P_{PCC}	0.0	1.00000	0.23960	0.04360	0.09110	0.14260	0.19210	06
P_{PCC}	0.0	0.99710	0.22930	0.04790	0.09170	0.13560	0.18150	07
P_{PCC}	0.0	0.96320	0.23980	0.04950	0.09320	0.14270	0.19230	08
P_{PCC}	0.0	0.99300	0.24630	0.04970	0.09730	0.14900	0.19460	09
P_{PCC}	0.0	0.96920	0.23330	0.04790	0.09170	0.13760	0.18550	10
P_{PCC}	0.0	0.98446	0.23785	0.04696	0.09387	0.14141	0.18914	Mean

APPENDIX D: Extended Version in Chapter 4.9 - Hidden Sensitive Data Experiments

Table D.1 Experiments in Chapter 4.9 - GAN + DP - Wind Turbine

Case Study: GAN + DP on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
$R\text{-Squared}$	0.93430	0.00580	0.64670	0.00220	0.00060	0.00190	0.00330	01
$R\text{-Squared}$	0.73490	0.00090	0.51060	0.00110	0.00160	0.00520	0.00040	02
$R\text{-Squared}$	0.93430	0.00100	0.64980	0.00090	0.00010	0.00230	0.00260	03
$R\text{-Squared}$	0.93430	0.00170	0.64640	0.00020	0.00170	0.00280	0.00140	04
$R\text{-Squared}$	0.59990	0.00070	0.41790	0.00020	0.00110	0.00180	0.00030	05
$R\text{-Squared}$	0.93430	0.01260	0.65920	0.00360	0.00290	0.00430	0.00450	06
$R\text{-Squared}$	0.93430	0.00510	0.63870	0.00180	0.00030	0.00110	0.00540	07
$R\text{-Squared}$	0.93430	0.00050	0.63990	0.00430	0.00080	0.00010	0.00150	08
$R\text{-Squared}$	0.48640	0.00460	0.33810	0.00020	0.00050	0.00060	0.00330	09
$R\text{-Squared}$	0.93430	0.00930	0.65130	0.00020	0.00050	0.00160	0.00060	10
$R\text{-Squared}$	0.83613	0.00422	0.57986	0.00147	0.00101	0.00217	0.00233	Mean
Case Study: GAN + DP on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.67160	0.21310	0.09590	0.20260	0.24510	0.43690	01
P_{MSE}	0.0	0.82570	0.24090	0.14470	0.26540	0.34350	0.63880	02
P_{MSE}	0.0	0.63450	0.17730	0.15960	0.28390	0.35060	0.53230	03
P_{MSE}	0.0	0.69490	0.19080	0.14180	0.24450	0.34250	0.53320	04
P_{MSE}	0.0	0.61870	0.20110	0.11060	0.23630	0.28170	0.41230	05
P_{MSE}	0.0	0.71960	0.21960	0.15380	0.30770	0.36270	0.54960	06
P_{MSE}	0.0	0.51680	0.16790	0.06740	0.15120	0.19320	0.30680	07
P_{MSE}	0.0	0.74130	0.21100	0.16250	0.29200	0.38980	0.60080	08
P_{MSE}	0.0	0.68830	0.20840	0.17190	0.30260	0.33910	0.49530	09
P_{MSE}	0.0	0.75400	0.22660	0.13060	0.24370	0.31190	0.53290	10
P_{MSE}	0.0	0.68654	0.20567	0.13388	0.25299	0.31601	0.50389	Mean
Case Study: GAN + DP on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10680	0.09460	0.20140	0.3465	0.45300	01
P_{VAR}	0.0	0.63820	0.10670	0.10960	0.21350	0.4028	0.52390	02
P_{VAR}	0.0	0.66980	0.10650	0.13720	0.25730	0.4649	0.57960	03
P_{VAR}	0.0	0.70830	0.10660	0.12430	0.23490	0.4889	0.60160	04
P_{VAR}	0.0	0.58470	0.10680	0.10470	0.22200	0.3663	0.45870	05
P_{VAR}	0.0	0.60790	0.10670	0.11890	0.23600	0.3953	0.49730	06
P_{VAR}	0.0	0.59730	0.10660	0.09010	0.19490	0.3538	0.45070	07
P_{VAR}	0.0	0.66230	0.10640	0.12450	0.23560	0.4693	0.57360	08
P_{VAR}	0.0	0.62930	0.10670	0.13230	0.24400	0.3911	0.49270	09
P_{VAR}	0.0	0.65040	0.10680	0.10830	0.21580	0.4161	0.52980	10
P_{VAR}	0.0	0.63441	0.10666	0.11445	0.22554	0.4095	0.51609	Mean
Case Study: GAN + DP on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.43200	0.55740	0.51720	0.40270	01
P_{HAUS}	0.0	0.36360	0.34300	0.37060	0.47260	0.44330	0.39580	02
P_{HAUS}	0.0	0.43430	0.36050	0.32550	0.50500	0.43140	0.36930	03
P_{HAUS}	0.0	0.41860	0.37460	0.38830	0.49510	0.52160	0.44880	04
P_{HAUS}	0.0	0.50480	0.36730	0.39920	0.46510	0.52190	0.33910	05
P_{HAUS}	0.0	0.46330	0.37670	0.42520	0.52900	0.51730	0.38000	06
P_{HAUS}	0.0	0.52240	0.40340	0.30100	0.50890	0.56960	0.43780	07
P_{HAUS}	0.0	0.37270	0.29740	0.38590	0.56220	0.52200	0.42220	08
P_{HAUS}	0.0	0.37460	0.31120	0.33110	0.47390	0.47100	0.38630	09
P_{HAUS}	0.0	0.46910	0.36650	0.52850	0.63550	0.67290	0.38370	10
P_{HAUS}	0.0	0.44229	0.36043	0.38873	0.52047	0.51882	0.39657	Mean

Table D.2 Experiments in Chapter 4.9 - GAN + DP - Wind Turbine

Case Study: GAN + DP on Wind Turbine Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	0.92060	0.23140	0.13620	0.2267	0.28130	0.45810	01
P_{KL}	0.0	0.93250	0.23890	0.16080	0.2617	0.33090	0.53280	02
P_{KL}	0.0	0.96520	0.24850	0.20540	0.3201	0.40130	0.59730	03
P_{KL}	0.0	0.97910	0.24600	0.17980	0.2838	0.37380	0.56290	04
P_{KL}	0.0	0.94430	0.24460	0.16310	0.2732	0.33090	0.48890	05
P_{KL}	0.0	0.92260	0.24820	0.17800	0.2905	0.35110	0.53410	06
P_{KL}	0.0	0.90030	0.23440	0.13310	0.2251	0.28010	0.44080	07
P_{KL}	0.0	0.94940	0.23970	0.17970	0.2765	0.36420	0.54840	08
P_{KL}	0.0	0.97910	0.24140	0.19870	0.3026	0.35960	0.53440	09
P_{KL}	0.0	0.89660	0.22430	0.14950	0.2418	0.31200	0.48340	10
P_{KL}	0.0	0.93897	0.23974	0.16843	0.2702	0.33852	0.51811	Mean
Case Study: GAN + DP on Wind Turbine Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.1653	0.35130	0.41130	0.65080	01
P_{PRO}	0.0	1.0	0.25540	0.1983	0.37570	0.44750	0.68660	02
P_{PRO}	0.0	1.0	0.25470	0.2441	0.42940	0.49410	0.70410	03
P_{PRO}	0.0	1.0	0.25300	0.2278	0.40360	0.49130	0.69760	04
P_{PRO}	0.0	1.0	0.25500	0.1846	0.38100	0.43480	0.64770	05
P_{PRO}	0.0	1.0	0.25460	0.2146	0.40650	0.46020	0.67520	06
P_{PRO}	0.0	1.0	0.25440	0.1547	0.34260	0.40790	0.64160	07
P_{PRO}	0.0	1.0	0.25450	0.2269	0.40550	0.48520	0.69490	08
P_{PRO}	0.0	1.0	0.25460	0.2372	0.41310	0.46630	0.67830	09
P_{PRO}	0.0	1.0	0.25430	0.1955	0.37520	0.45090	0.67690	10
P_{PRO}	0.0	1.0	0.25444	0.2049	0.38839	0.45495	0.67537	Mean
Case Study: GAN + DP on Wind Turbine Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.99700	0.15030	0.14190	0.28500	0.42830	0.56870	01
P_{PCC}	0.0	0.99160	0.15350	0.14060	0.28140	0.42340	0.56830	02
P_{PCC}	0.0	1.00000	0.15310	0.14590	0.28470	0.43060	0.57370	03
P_{PCC}	0.0	0.99770	0.15130	0.14130	0.28260	0.42390	0.56810	04
P_{PCC}	0.0	0.98680	0.14960	0.14120	0.28660	0.42780	0.56460	05
P_{PCC}	0.0	0.99300	0.15140	0.14150	0.28150	0.42570	0.56570	06
P_{PCC}	0.0	1.00000	0.14960	0.14250	0.28630	0.43020	0.57260	07
P_{PCC}	0.0	0.98390	0.15150	0.14010	0.28030	0.42190	0.56200	08
P_{PCC}	0.0	0.99440	0.14900	0.14330	0.28510	0.42410	0.56320	09
P_{PCC}	0.0	1.00000	0.14960	0.14100	0.28490	0.42880	0.57120	10
P_{PCC}	0.0	0.99444	0.15089	0.14193	0.28384	0.42647	0.56781	Mean

Table D.3 Experiments in Chapter 4.9 - GAN + DP - Steam Production

Case Study: GAN + DP on Steam Production Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	0.65850	0.01050	0.03590	0.01520	0.00490	0.00270	0.00560	01
R -Squared	0.51800	0.00400	0.04890	0.00330	0.00030	0.00550	0.00240	02
R -Squared	0.65850	0.00450	0.06200	0.00760	0.00100	0.00400	0.02270	03
R -Squared	0.65850	0.01470	0.07880	0.00510	0.00160	0.01670	0.00370	04
R -Squared	0.42280	0.00230	0.04350	0.00270	0.00590	0.00690	0.01570	05
R -Squared	0.65850	0.01750	0.05710	0.00460	0.01770	0.00460	0.00880	06
R -Squared	0.65850	0.00450	0.06250	0.00590	0.01250	0.01540	0.00010	07
R -Squared	0.65850	0.00260	0.06640	0.00060	0.00100	0.02110	0.01370	08
R -Squared	0.34280	0.00850	0.01070	0.00430	0.00110	0.00360	0.00580	09
R -Squared	0.65850	0.00200	0.07440	0.01400	0.00170	0.00370	0.00200	10
R -Squared	0.58931	0.00711	0.05402	0.00633	0.00477	0.00842	0.00805	Mean
Case Study: GAN + DP on Steam Production Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.2679	0.48840	0.05410	0.10860	0.19660	0.27460	01
P_{MSE}	0.0	0.2992	0.54090	0.03840	0.09200	0.17640	0.27220	02
P_{MSE}	0.0	0.2118	0.40090	0.02830	0.07330	0.17800	0.27110	03
P_{MSE}	0.0	0.2366	0.43600	0.02190	0.07160	0.13080	0.22090	04
P_{MSE}	0.0	0.2623	0.44510	0.02570	0.06090	0.13130	0.20490	05
P_{MSE}	0.0	0.2831	0.49650	0.02460	0.07350	0.13650	0.20960	06
P_{MSE}	0.0	0.2032	0.37970	0.04260	0.08030	0.14440	0.24590	07
P_{MSE}	0.0	0.2481	0.48560	0.03110	0.07920	0.13790	0.26520	08
P_{MSE}	0.0	0.2719	0.46460	0.04960	0.08940	0.15910	0.22880	09
P_{MSE}	0.0	0.2849	0.50160	0.03250	0.09010	0.15870	0.23100	10
P_{MSE}	0.0	0.2569	0.46393	0.03488	0.08189	0.15497	0.24242	Mean
Case Study: GAN + DP on Steam Production Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.13340	0.36450	0.45410	0.52620	01
P_{VAR}	0.0	0.65740	1.0	0.10050	0.34760	0.41190	0.49010	02
P_{VAR}	0.0	0.63460	1.0	0.10340	0.36220	0.43130	0.51610	03
P_{VAR}	0.0	0.66250	1.0	0.08920	0.34250	0.40780	0.48100	04
P_{VAR}	0.0	0.68900	1.0	0.10060	0.34150	0.40380	0.48600	05
P_{VAR}	0.0	0.64830	1.0	0.08970	0.33180	0.39300	0.47400	06
P_{VAR}	0.0	0.66160	1.0	0.13390	0.37620	0.44890	0.52700	07
P_{VAR}	0.0	0.63930	1.0	0.10420	0.33910	0.39760	0.48230	08
P_{VAR}	0.0	0.67320	1.0	0.12680	0.37910	0.43070	0.50530	09
P_{VAR}	0.0	0.64810	1.0	0.10200	0.35760	0.41780	0.48580	10
P_{VAR}	0.0	0.65727	1.0	0.10837	0.35421	0.41969	0.49738	Mean
Case Study: GAN + DP on Steam Production Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.6884	0.5214	0.78310	0.80120	01
P_{HAUS}	0.0	0.70670	0.92660	0.4516	0.5483	0.57990	0.81280	02
P_{HAUS}	0.0	0.73660	0.89930	0.4287	0.6313	0.62150	0.70810	03
P_{HAUS}	0.0	0.73450	0.92090	0.4044	0.5495	0.51510	0.63310	04
P_{HAUS}	0.0	0.64090	0.88860	0.3878	0.5416	0.65530	0.56000	05
P_{HAUS}	0.0	0.75500	0.98880	0.4048	0.5596	0.73130	0.77330	06
P_{HAUS}	0.0	0.67750	0.90110	0.4458	0.4583	0.63330	0.78630	07
P_{HAUS}	0.0	0.71480	1.00000	0.4477	0.5107	0.54520	0.55390	08
P_{HAUS}	0.0	0.63940	0.87210	0.4229	0.4469	0.68850	0.68800	09
P_{HAUS}	0.0	0.78590	0.98030	0.5219	0.5414	0.67720	0.70750	10
P_{HAUS}	0.0	0.71475	0.92473	0.4604	0.5309	0.64304	0.70242	Mean

Table D.4 Experiments in Chapter 4.9 - GAN + DP - Steam Production

Case Study: GAN + DP on Steam Production Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.03750	0.06350	0.13560	0.19050	01
P_{KL}	0.0	0.30100	0.25430	0.02640	0.05560	0.11990	0.18420	02
P_{KL}	0.0	0.29800	0.26450	0.02740	0.05790	0.14880	0.21270	03
P_{KL}	0.0	0.31620	0.26190	0.02420	0.05430	0.12350	0.18680	04
P_{KL}	0.0	0.34160	0.26850	0.02750	0.05810	0.12520	0.18620	05
P_{KL}	0.0	0.33080	0.26230	0.02390	0.05080	0.11640	0.17570	06
P_{KL}	0.0	0.29520	0.25720	0.03790	0.06720	0.13440	0.19870	07
P_{KL}	0.0	0.29030	0.25810	0.02650	0.05290	0.11740	0.18480	08
P_{KL}	0.0	0.33710	0.25890	0.03610	0.06620	0.13540	0.18970	09
P_{KL}	0.0	0.30220	0.23780	0.02520	0.05320	0.11480	0.17080	10
P_{KL}	0.0	0.31095	0.25746	0.02926	0.05797	0.12714	0.18801	Mean
Case Study: GAN + DP on Steam Production Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.22320	0.40520	0.6869	0.85660	01
P_{PRO}	0.0	0.99820	0.70340	0.17690	0.38440	0.6530	0.84370	02
P_{PRO}	0.0	0.99830	0.70060	0.18270	0.39290	0.6782	0.85910	03
P_{PRO}	0.0	0.99900	0.69780	0.16050	0.37560	0.6525	0.84740	04
P_{PRO}	0.0	0.99870	0.70310	0.17070	0.37650	0.6535	0.83920	05
P_{PRO}	0.0	0.99900	0.70190	0.15910	0.37080	0.6490	0.83800	06
P_{PRO}	0.0	0.99840	0.69810	0.22340	0.41080	0.6717	0.85700	07
P_{PRO}	0.0	0.99850	0.70110	0.17650	0.37200	0.6430	0.84040	08
P_{PRO}	0.0	0.99830	0.70100	0.21840	0.40820	0.6637	0.84630	09
P_{PRO}	0.0	0.99830	0.70120	0.17800	0.38410	0.6585	0.84200	10
P_{PRO}	0.0	0.99848	0.70044	0.18694	0.38805	0.6610	0.84697	Mean
Case Study: GAN + DP on Steam Production Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	1.0000	0.44580	0.09200	0.17780	0.27160	0.36010	01
P_{PCC}	0.0	1.0000	0.45020	0.08950	0.17990	0.27490	0.36620	02
P_{PCC}	0.0	0.9941	0.44970	0.08920	0.17940	0.27040	0.36500	03
P_{PCC}	0.0	1.0000	0.44690	0.08810	0.18070	0.27070	0.35790	04
P_{PCC}	0.0	0.9899	0.44380	0.09250	0.17960	0.27040	0.36210	05
P_{PCC}	0.0	0.9901	0.44820	0.08820	0.17820	0.26820	0.36370	06
P_{PCC}	0.0	0.9998	0.44320	0.09240	0.18130	0.27470	0.36080	07
P_{PCC}	0.0	1.0000	0.44510	0.09280	0.18470	0.27480	0.36670	08
P_{PCC}	0.0	0.9930	0.44330	0.08940	0.18320	0.27260	0.35930	09
P_{PCC}	0.0	0.9971	0.45050	0.09060	0.18500	0.27560	0.36250	10
P_{PCC}	0.0	0.9964	0.44667	0.09047	0.18098	0.27239	0.36243	Mean

Table D.5 Experiments in Chapter 4.9 - GAN + DP - Energy Efficiency

Case Study: GAN + DP on Energy Efficiency Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	1.00000	0.01360	0.57300	0.02550	0.01880	0.00340	0.00250	01
R -Squared	0.78660	0.00540	0.40650	0.04670	0.01590	0.03670	0.00350	02
R -Squared	1.00000	0.06790	0.47280	0.03080	0.04500	0.01470	0.02300	03
R -Squared	1.00000	0.03380	0.52350	0.03580	0.06490	0.03680	0.01160	04
R -Squared	0.64210	0.01790	0.29910	0.01630	0.02500	0.01750	0.00220	05
R -Squared	1.00000	0.01910	0.56580	0.03890	0.00880	0.00960	0.00690	06
R -Squared	1.00000	0.02990	0.53150	0.01840	0.05300	0.03370	0.06440	07
R -Squared	1.00000	0.00400	0.53060	0.00310	0.00530	0.05220	0.01640	08
R -Squared	0.52060	0.00190	0.28540	0.02740	0.02120	0.01990	0.04450	09
R -Squared	1.00000	0.00300	0.54070	0.01420	0.01370	0.02340	0.00950	10
R -Squared	0.89493	0.01965	0.47289	0.02571	0.02716	0.02479	0.01845	Mean
Case Study: GAN + DP on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	0.89180	0.47390	0.08580	0.24250	0.39180	0.52610	01
P_{MSE}	0.0	1.00000	0.51900	0.10970	0.24890	0.44730	0.57380	02
P_{MSE}	0.0	0.74220	0.38820	0.14600	0.25470	0.38820	0.46270	03
P_{MSE}	0.0	0.84590	0.41780	0.08220	0.18490	0.34590	0.39730	04
P_{MSE}	0.0	0.89440	0.44720	0.17250	0.30990	0.42960	0.58100	05
P_{MSE}	0.0	0.91150	0.49230	0.07690	0.22310	0.38080	0.47690	06
P_{MSE}	0.0	0.73530	0.35880	0.11760	0.22650	0.37350	0.50000	07
P_{MSE}	0.0	0.89390	0.47350	0.10980	0.22730	0.37500	0.44320	08
P_{MSE}	0.0	0.93800	0.45260	0.16060	0.29200	0.45620	0.63870	09
P_{MSE}	0.0	0.99210	0.49600	0.17060	0.30950	0.51980	0.64680	10
P_{MSE}	0.0	0.88451	0.45193	0.12317	0.25193	0.41081	0.52465	Mean
Case Study: GAN + DP on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.06340	0.14100	0.20140	0.28400	01
P_{VAR}	0.0	0.52300	0.15090	0.06180	0.13200	0.19810	0.27060	02
P_{VAR}	0.0	0.52600	0.15130	0.08740	0.15510	0.21980	0.28570	03
P_{VAR}	0.0	0.53810	0.15030	0.06710	0.13690	0.20350	0.26250	04
P_{VAR}	0.0	0.55510	0.15130	0.08640	0.16400	0.22280	0.31200	05
P_{VAR}	0.0	0.52160	0.15060	0.05850	0.13070	0.19400	0.26840	06
P_{VAR}	0.0	0.53130	0.15080	0.07490	0.14850	0.21360	0.29200	07
P_{VAR}	0.0	0.51010	0.15130	0.07170	0.13790	0.20130	0.25960	08
P_{VAR}	0.0	0.56190	0.15090	0.08120	0.15820	0.21900	0.31590	09
P_{VAR}	0.0	0.54090	0.15110	0.07910	0.15180	0.21900	0.29760	10
P_{VAR}	0.0	0.53234	0.15096	0.07315	0.14561	0.20925	0.28483	Mean
Case Study: GAN + DP on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.47720	0.72370	0.67900	0.8319	01
P_{HAUS}	0.0	0.97250	1.00000	0.50150	0.61500	0.80250	0.8487	02
P_{HAUS}	0.0	0.87580	0.91380	0.54120	0.73460	0.71340	1.0000	03
P_{HAUS}	0.0	0.91530	1.00000	0.50530	0.71720	0.62870	0.6653	04
P_{HAUS}	0.0	0.89050	1.00000	0.49700	0.66340	0.76300	0.7352	05
P_{HAUS}	0.0	0.92740	1.00000	0.49480	0.71490	0.79800	0.8823	06
P_{HAUS}	0.0	0.89190	1.00000	0.54290	0.62600	0.78080	0.8456	07
P_{HAUS}	0.0	0.83290	0.94430	0.52960	0.68500	0.68670	0.7250	08
P_{HAUS}	0.0	0.87520	1.00000	0.51140	0.55140	0.67980	0.7880	09
P_{HAUS}	0.0	0.94490	0.99170	0.57970	0.80640	0.84380	1.0000	10
P_{HAUS}	0.0	0.89647	0.98498	0.51806	0.68376	0.73757	0.8322	Mean

Table D.6 Experiments in Chapter 4.9 - GAN + DP - Energy Efficiency

Case Study: GAN + DP on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.0825	0.17140	0.29210	0.37780	01
P_{KL}	0.0	1.0	0.32150	0.0836	0.16080	0.29580	0.37300	02
P_{KL}	0.0	1.0	0.34450	0.1171	0.20400	0.34110	0.41810	03
P_{KL}	0.0	1.0	0.33110	0.0927	0.16560	0.31130	0.37420	04
P_{KL}	0.0	1.0	0.34560	0.1208	0.21140	0.33220	0.43960	05
P_{KL}	0.0	1.0	0.34100	0.0820	0.17050	0.28850	0.37380	06
P_{KL}	0.0	1.0	0.32800	0.1029	0.18650	0.32480	0.42120	07
P_{KL}	0.0	1.0	0.33230	0.0903	0.16770	0.28710	0.35160	08
P_{KL}	0.0	1.0	0.33110	0.1126	0.19870	0.33110	0.44040	09
P_{KL}	0.0	1.0	0.31690	0.1015	0.18150	0.31380	0.39080	10
P_{KL}	0.0	1.0	0.33126	0.0986	0.18181	0.31178	0.39605	Mean
Case Study: GAN + DP on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.11460	0.31260	0.50020	0.5595	01
P_{PRO}	0.0	0.9946	0.34650	0.11060	0.29540	0.49600	0.5541	02
P_{PRO}	0.0	0.9948	0.34910	0.15960	0.32140	0.52070	0.5652	03
P_{PRO}	0.0	0.9949	0.34690	0.10370	0.29190	0.47290	0.5421	04
P_{PRO}	0.0	0.9966	0.35290	0.15680	0.32890	0.52320	0.5846	05
P_{PRO}	0.0	0.9944	0.35920	0.09440	0.28960	0.49240	0.5505	06
P_{PRO}	0.0	0.9944	0.35240	0.14460	0.32350	0.51720	0.5800	07
P_{PRO}	0.0	0.9941	0.35050	0.12490	0.30130	0.49360	0.5360	08
P_{PRO}	0.0	0.9947	0.34950	0.15450	0.34010	0.52580	0.5941	09
P_{PRO}	0.0	0.9972	0.34910	0.15320	0.33580	0.52430	0.5739	10
P_{PRO}	0.0	0.9950	0.35075	0.13169	0.31405	0.50663	0.5640	Mean
Case Study: GAN + DP on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.09730	0.19360	0.28990	0.39510	01
P_{PCC}	0.0	0.97960	0.20190	0.09850	0.19690	0.29240	0.39380	02
P_{PCC}	0.0	0.99240	0.20430	0.10510	0.20830	0.30440	0.40860	03
P_{PCC}	0.0	0.99910	0.20280	0.11090	0.21080	0.31870	0.41060	04
P_{PCC}	0.0	1.00000	0.20360	0.10280	0.20360	0.30040	0.40020	05
P_{PCC}	0.0	0.98320	0.20790	0.10200	0.20000	0.30100	0.40100	06
P_{PCC}	0.0	0.99410	0.20440	0.09770	0.19340	0.29810	0.38690	07
P_{PCC}	0.0	0.97310	0.20410	0.10310	0.19620	0.30320	0.39240	08
P_{PCC}	0.0	1.00000	0.20360	0.09930	0.20060	0.29590	0.40020	09
P_{PCC}	0.0	0.99810	0.20240	0.09770	0.19240	0.30010	0.38790	10
P_{PCC}	0.0	0.98965	0.20385	0.10144	0.19958	0.30041	0.39767	Mean

Table D.7 Experiments in Chapter 4.9 - GAN + DP - Synchronous Motors

Case Study: GAN + DP on Synchronous Motors Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.63400	0.01520	0.08940	0.0246	0.20980	0.01840	0.00230	01
<i>R-Squared</i>	0.49870	0.03180	0.05950	0.0954	0.05210	1.00000	0.04590	02
<i>R-Squared</i>	0.63400	0.00670	0.06920	0.0065	0.02800	0.02560	0.00570	03
<i>R-Squared</i>	0.63400	0.01480	0.13740	0.0197	0.05900	0.00820	0.00760	04
<i>R-Squared</i>	0.40710	0.02030	0.02040	0.0112	0.01800	1.00000	0.00810	05
<i>R-Squared</i>	0.63400	0.01150	0.08030	0.0131	0.00390	0.03930	0.00510	06
<i>R-Squared</i>	0.63400	0.03780	0.01280	0.0018	0.11450	0.00450	0.03010	07
<i>R-Squared</i>	0.63400	0.04710	0.20970	0.0084	0.05980	0.14050	0.01290	08
<i>R-Squared</i>	0.33010	0.00330	0.07190	0.0076	0.02150	1.00000	0.01520	09
<i>R-Squared</i>	0.63400	0.05200	0.12030	0.0217	0.01870	0.05310	0.06150	10
<i>R-Squared</i>	0.56739	0.02405	0.08709	0.0210	0.05853	0.32896	0.01944	Mean
Case Study: GAN + DP on Synchronous Motors Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.64180	0.42540	0.21640	0.65670	0.91790	1.0000	01
P_{MSE}	0.0	0.76790	0.46410	0.23630	0.47260	0.62450	0.9620	02
P_{MSE}	0.0	0.57140	0.34160	0.17390	0.29810	0.70190	1.0000	03
P_{MSE}	0.0	0.61640	0.38360	0.27400	0.41780	0.78770	1.0000	04
P_{MSE}	0.0	0.64790	0.40850	0.31690	0.52110	0.71830	1.0000	05
P_{MSE}	0.0	0.67690	0.43850	0.17690	0.50770	0.80770	1.0000	06
P_{MSE}	0.0	0.51180	0.33530	0.34710	0.61760	0.77650	1.0000	07
P_{MSE}	0.0	0.62880	0.43940	0.31060	0.50000	0.74240	1.0000	08
P_{MSE}	0.0	0.63500	0.41610	0.29930	0.53280	0.73720	1.0000	09
P_{MSE}	0.0	0.64290	0.45240	0.18250	0.45240	0.73020	1.0000	10
P_{MSE}	0.0	0.63408	0.41049	0.25339	0.49768	0.75443	0.9962	Mean
Case Study: GAN + DP on Synchronous Motors Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.13950	0.30620	0.44670	0.5604	01
P_{VAR}	0.0	0.55430	0.18330	0.14540	0.29430	0.40840	0.5810	02
P_{VAR}	0.0	0.53210	0.18390	0.14000	0.25210	0.43450	0.6200	03
P_{VAR}	0.0	0.52760	0.18260	0.15330	0.27390	0.43580	0.5921	04
P_{VAR}	0.0	0.55830	0.18440	0.16220	0.30670	0.43350	0.6240	05
P_{VAR}	0.0	0.55590	0.18360	0.12610	0.29060	0.45810	0.5912	06
P_{VAR}	0.0	0.51540	0.18260	0.16190	0.32180	0.46100	0.6198	07
P_{VAR}	0.0	0.52320	0.18380	0.13230	0.27530	0.39550	0.5727	08
P_{VAR}	0.0	0.55810	0.18320	0.16300	0.31640	0.44860	0.6116	09
P_{VAR}	0.0	0.51130	0.18340	0.12380	0.26580	0.41130	0.5902	10
P_{VAR}	0.0	0.53506	0.18342	0.14475	0.29031	0.43334	0.5963	Mean
Case Study: GAN + DP on Synchronous Motors Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.6353	0.35320	0.19940	0.35140	0.44670	0.70300	01
P_{HAUS}	0.0	0.5595	0.43530	0.17960	0.25780	0.38090	0.66820	02
P_{HAUS}	0.0	0.5060	0.40010	0.20000	0.28210	0.43950	0.72090	03
P_{HAUS}	0.0	0.4219	0.42220	0.20410	0.28750	0.52980	0.83970	04
P_{HAUS}	0.0	0.5526	0.35430	0.18500	0.26020	0.48840	0.62000	05
P_{HAUS}	0.0	0.4249	0.43450	0.21730	0.33260	0.48420	0.56220	06
P_{HAUS}	0.0	0.4733	0.39010	0.26610	0.43620	0.43060	0.77450	07
P_{HAUS}	0.0	0.5093	0.43720	0.21680	0.28990	0.44210	0.54820	08
P_{HAUS}	0.0	0.5196	0.39280	0.21370	0.36830	0.37990	0.57020	09
P_{HAUS}	0.0	0.5846	0.37570	0.23230	0.34640	0.40660	0.76850	10
P_{HAUS}	0.0	0.5187	0.39954	0.21143	0.32124	0.44287	0.67754	Mean

Table D.8 Experiments in Chapter 4.9 - GAN + DP - Synchronous Motors

Case Study: GAN + DP on Synchronous Motors Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.15240	0.37460	0.55240	0.65400	01
P_{KL}	0.0	0.77810	0.27010	0.16720	0.33440	0.48230	0.66880	02
P_{KL}	0.0	0.74250	0.28760	0.18730	0.32780	0.59530	0.78260	03
P_{KL}	0.0	0.76160	0.30460	0.19870	0.35760	0.58940	0.72190	04
P_{KL}	0.0	0.72480	0.30200	0.20130	0.36910	0.52350	0.71140	05
P_{KL}	0.0	0.68850	0.30160	0.16390	0.35410	0.55080	0.67540	06
P_{KL}	0.0	0.67520	0.27650	0.22510	0.42440	0.57880	0.75240	07
P_{KL}	0.0	0.67740	0.29030	0.18060	0.33550	0.48390	0.64520	08
P_{KL}	0.0	0.71520	0.29800	0.21190	0.38410	0.55630	0.70200	09
P_{KL}	0.0	0.59690	0.27080	0.14150	0.28920	0.46770	0.62770	10
P_{KL}	0.0	0.70205	0.28745	0.18299	0.35508	0.53804	0.69414	Mean
Case Study: GAN + DP on Synchronous Motors Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.43180	0.67980	0.85610	0.9237	01
P_{PRO}	0.0	0.99820	0.41830	0.42350	0.64690	0.83220	0.9234	02
P_{PRO}	0.0	0.99850	0.40750	0.43020	0.61200	0.84920	0.9350	03
P_{PRO}	0.0	0.99750	0.42730	0.46500	0.65030	0.84720	0.9232	04
P_{PRO}	0.0	0.99790	0.42730	0.48140	0.67750	0.84190	0.9291	05
P_{PRO}	0.0	0.99480	0.42480	0.39570	0.64790	0.86150	0.9241	06
P_{PRO}	0.0	0.99750	0.40320	0.47340	0.68900	0.85770	0.9293	07
P_{PRO}	0.0	0.99550	0.42320	0.42890	0.64390	0.82560	0.9259	08
P_{PRO}	0.0	0.99700	0.43450	0.46290	0.68160	0.84560	0.9322	09
P_{PRO}	0.0	0.99620	0.40890	0.39430	0.62650	0.83140	0.9181	10
P_{PRO}	0.0	0.99706	0.41868	0.43871	0.65554	0.84484	0.9264	Mean
Case Study: GAN + DP on Synchronous Motors Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.19260	0.38120	0.57380	0.77830	01
P_{PCC}	0.0	0.99250	0.23870	0.20690	0.40970	0.61060	0.79360	02
P_{PCC}	0.0	0.99140	0.23430	0.19830	0.40060	0.60890	0.79510	03
P_{PCC}	0.0	0.98510	0.24580	0.19380	0.37570	0.58750	0.76730	04
P_{PCC}	0.0	0.99410	0.23910	0.19170	0.38340	0.58300	0.77670	05
P_{PCC}	0.0	1.00000	0.23960	0.20000	0.40200	0.60000	0.78610	06
P_{PCC}	0.0	0.99710	0.22930	0.20140	0.39490	0.60230	0.80170	07
P_{PCC}	0.0	0.96320	0.23980	0.17240	0.38650	0.56090	0.76310	08
P_{PCC}	0.0	0.99300	0.24630	0.20260	0.40120	0.58990	0.77660	09
P_{PCC}	0.0	0.96920	0.23330	0.19340	0.39490	0.57830	0.80370	10
P_{PCC}	0.0	0.98446	0.23785	0.19531	0.39301	0.58952	0.78422	Mean

Table D.9 Experiments in Chapter 4.9 - GAN Only - Wind Turbine

Case Study: GAN Only on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.9343	0.00580	0.64670	0.00260	0.00040	0.00370	0.00470	01
<i>R-Squared</i>	0.9343	0.00110	0.64910	0.00070	0.00030	0.00060	0.00140	02
<i>R-Squared</i>	0.9343	0.00100	0.64980	0.00110	0.00070	0.00170	0.00060	03
<i>R-Squared</i>	0.9343	0.00170	0.64640	0.00050	0.00080	0.00170	0.00080	04
<i>R-Squared</i>	0.9343	0.00110	0.65070	0.00020	0.00120	0.00270	0.00160	05
<i>R-Squared</i>	0.9343	0.01260	0.65920	0.00330	0.00230	0.01110	0.01130	06
<i>R-Squared</i>	0.9343	0.00510	0.63870	0.00020	0.00090	0.00030	0.00490	07
<i>R-Squared</i>	0.9343	0.00050	0.63990	0.00100	0.00040	0.00200	0.00130	08
<i>R-Squared</i>	0.9343	0.00890	0.64940	0.00080	0.00020	0.00120	0.01170	09
<i>R-Squared</i>	0.9343	0.00930	0.65130	0.00090	0.00070	0.00490	0.00600	10
<i>R-Squared</i>	0.9343	0.00471	0.64812	0.00113	0.00079	0.00299	0.00443	Mean
Case Study: GAN Only on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.75310	0.23890	0.08370	0.16150	0.19120	0.32890	01
P_{MSE}	0.0	0.82570	0.24090	0.10250	0.17470	0.21690	0.38570	02
P_{MSE}	0.0	0.85480	0.23890	0.13140	0.21510	0.26320	0.41260	03
P_{MSE}	0.0	0.82150	0.22550	0.10970	0.19070	0.24860	0.38740	04
P_{MSE}	0.0	0.69170	0.22480	0.09020	0.16300	0.19130	0.30350	05
P_{MSE}	0.0	0.78950	0.24090	0.11430	0.19870	0.23500	0.36750	06
P_{MSE}	0.0	0.70280	0.22840	0.07440	0.14280	0.17720	0.29720	07
P_{MSE}	0.0	0.82920	0.23600	0.12120	0.19960	0.25420	0.39960	08
P_{MSE}	0.0	0.73390	0.22220	0.11670	0.18910	0.21670	0.33890	09
P_{MSE}	0.0	0.76000	0.22840	0.09720	0.16560	0.20560	0.34280	10
P_{MSE}	0.0	0.77622	0.23249	0.10413	0.18008	0.21999	0.35641	Mean
Case Study: GAN Only on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10680	0.08810	0.17290	0.28770	0.37640	01
P_{VAR}	0.0	0.63820	0.10670	0.09880	0.17740	0.31980	0.41650	02
P_{VAR}	0.0	0.66980	0.10650	0.11590	0.20540	0.35510	0.44860	03
P_{VAR}	0.0	0.70830	0.10660	0.10840	0.19780	0.38760	0.48020	04
P_{VAR}	0.0	0.58470	0.10680	0.09460	0.18050	0.29130	0.37320	05
P_{VAR}	0.0	0.60790	0.10670	0.10310	0.18840	0.30850	0.39460	06
P_{VAR}	0.0	0.59730	0.10660	0.08360	0.16570	0.29250	0.37830	07
P_{VAR}	0.0	0.66230	0.10640	0.10890	0.19050	0.35210	0.44230	08
P_{VAR}	0.0	0.62930	0.10670	0.11390	0.19970	0.31790	0.40650	09
P_{VAR}	0.0	0.65040	0.10680	0.09850	0.18120	0.33420	0.42730	10
P_{VAR}	0.0	0.63441	0.10666	0.10138	0.18595	0.32467	0.41439	Mean
Case Study: GAN Only on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.36710	0.42210	0.4830	0.43030	01
P_{HAUS}	0.0	0.36360	0.34300	0.37060	0.47260	0.4956	0.33350	02
P_{HAUS}	0.0	0.47530	0.39450	0.35530	0.35520	0.4341	0.40710	03
P_{HAUS}	0.0	0.41860	0.37460	0.38830	0.49510	0.5211	0.31830	04
P_{HAUS}	0.0	0.50480	0.36730	0.36830	0.38050	0.3753	0.37520	05
P_{HAUS}	0.0	0.46330	0.37670	0.31920	0.40810	0.3982	0.35290	06
P_{HAUS}	0.0	0.52240	0.40340	0.30100	0.46810	0.4719	0.49060	07
P_{HAUS}	0.0	0.37270	0.29740	0.38590	0.42860	0.4602	0.35920	08
P_{HAUS}	0.0	0.37460	0.31120	0.31670	0.33260	0.3666	0.34670	09
P_{HAUS}	0.0	0.47310	0.36960	0.38170	0.41690	0.4780	0.38210	10
P_{HAUS}	0.0	0.44679	0.36414	0.35541	0.41798	0.4484	0.37959	Mean

Table D.10 Experiments in Chapter 4.9 - GAN Only - Wind Turbine

Case Study: GAN Only on Wind Turbine Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.92060	0.23140	0.13140	0.22220	0.27210	0.42630	01
P_{KL}	0.0	0.93250	0.23890	0.14690	0.22960	0.28010	0.44570	02
P_{KL}	0.0	0.96520	0.24850	0.16720	0.25790	0.31070	0.47290	03
P_{KL}	0.0	0.97910	0.24600	0.16090	0.25530	0.31690	0.47780	04
P_{KL}	0.0	0.94430	0.24460	0.15340	0.24460	0.29230	0.45070	05
P_{KL}	0.0	0.92260	0.24820	0.15930	0.24360	0.29050	0.44950	06
P_{KL}	0.0	0.90030	0.23440	0.13310	0.22060	0.27110	0.42730	07
P_{KL}	0.0	0.94940	0.23970	0.15190	0.23520	0.28580	0.44230	08
P_{KL}	0.0	0.97910	0.24140	0.17020	0.26030	0.31230	0.47320	09
P_{KL}	0.0	0.89660	0.22430	0.14060	0.22860	0.27690	0.43080	10
P_{KL}	0.0	0.93897	0.23974	0.15149	0.23979	0.29087	0.44965	Mean
Case Study: GAN Only on Wind Turbine Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.14960	0.30270	0.35320	0.59820	01
P_{PRO}	0.0	1.0	0.25540	0.17610	0.32010	0.38200	0.62650	02
P_{PRO}	0.0	1.0	0.25470	0.20420	0.35920	0.41910	0.64420	03
P_{PRO}	0.0	1.0	0.25300	0.19680	0.35030	0.42650	0.64690	04
P_{PRO}	0.0	1.0	0.25500	0.16230	0.31870	0.36730	0.59610	05
P_{PRO}	0.0	1.0	0.25460	0.18290	0.33490	0.38460	0.61430	06
P_{PRO}	0.0	1.0	0.25440	0.14040	0.29440	0.35290	0.59660	07
P_{PRO}	0.0	1.0	0.25450	0.19610	0.34120	0.40880	0.63310	08
P_{PRO}	0.0	1.0	0.25460	0.20230	0.35020	0.39930	0.62830	09
P_{PRO}	0.0	1.0	0.25430	0.17460	0.32400	0.39020	0.62760	10
P_{PRO}	0.0	1.0	0.25444	0.17853	0.32957	0.38839	0.62118	Mean
Case Study: GAN Only on Wind Turbine Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.99700	0.15030	0.14190	0.28650	0.42980	0.57290	01
P_{PCC}	0.0	0.99160	0.15350	0.14060	0.28140	0.42480	0.56690	02
P_{PCC}	0.0	1.00000	0.15310	0.14590	0.28610	0.43060	0.57220	03
P_{PCC}	0.0	0.99770	0.15130	0.13990	0.28260	0.42670	0.56810	04
P_{PCC}	0.0	0.98680	0.14960	0.14260	0.28370	0.42630	0.56750	05
P_{PCC}	0.0	0.99300	0.15140	0.14150	0.28290	0.42570	0.56720	06
P_{PCC}	0.0	1.00000	0.14960	0.14250	0.28490	0.42880	0.57120	07
P_{PCC}	0.0	0.98390	0.15150	0.14010	0.27890	0.42050	0.56200	08
P_{PCC}	0.0	0.99440	0.14900	0.14330	0.28660	0.42700	0.56600	09
P_{PCC}	0.0	1.00000	0.14960	0.14100	0.28350	0.42740	0.56970	10
P_{PCC}	0.0	0.99444	0.15089	0.14193	0.28371	0.42676	0.56837	Mean

Table D.11 Experiments in Chapter 4.9 - GAN Only - Steam Production

Case Study: GAN Only on Steam Production Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.6585	0.01050	0.03590	0.01440	0.00930	0.00860	0.00890	01
<i>R-Squared</i>	0.6585	0.00510	0.06220	0.00010	0.00310	0.00190	0.00690	02
<i>R-Squared</i>	0.6585	0.00450	0.06200	0.00220	0.00190	0.00480	0.00250	03
<i>R-Squared</i>	0.6585	0.01470	0.07880	0.00480	0.00850	0.01000	0.01470	04
<i>R-Squared</i>	0.6585	0.00360	0.06780	0.01060	0.00940	0.01620	0.01580	05
<i>R-Squared</i>	0.6585	0.01750	0.05710	0.00440	0.00390	0.00450	0.00160	06
<i>R-Squared</i>	0.6585	0.00450	0.06250	0.00740	0.00660	0.00210	0.00220	07
<i>R-Squared</i>	0.6585	0.00260	0.06640	0.00380	0.00040	0.00390	0.00040	08
<i>R-Squared</i>	0.6585	0.01630	0.02050	0.00830	0.00910	0.00920	0.00780	09
<i>R-Squared</i>	0.6585	0.00200	0.07440	0.00660	0.00030	0.00050	0.00490	10
<i>R-Squared</i>	0.6585	0.00813	0.05876	0.00626	0.00525	0.00617	0.00657	Mean
Case Study: GAN Only on Steam Production Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.30040	0.5477	0.01880	0.0305	0.08740	0.14440	01
P_{MSE}	0.0	0.29920	0.5409	0.01520	0.0308	0.08820	0.14560	02
P_{MSE}	0.0	0.28540	0.5402	0.01510	0.0268	0.09120	0.14440	03
P_{MSE}	0.0	0.27980	0.5154	0.01460	0.0259	0.07370	0.13240	04
P_{MSE}	0.0	0.29330	0.4976	0.01420	0.0358	0.08580	0.14690	05
P_{MSE}	0.0	0.31050	0.5447	0.01520	0.0308	0.08820	0.15740	06
P_{MSE}	0.0	0.27640	0.5164	0.01800	0.0328	0.08360	0.13440	07
P_{MSE}	0.0	0.27750	0.5432	0.01530	0.0271	0.08090	0.13090	08
P_{MSE}	0.0	0.28990	0.4953	0.01400	0.0319	0.08130	0.14510	09
P_{MSE}	0.0	0.28720	0.5056	0.01440	0.0256	0.08000	0.14200	10
P_{MSE}	0.0	0.28996	0.5247	0.01548	0.0298	0.08403	0.14235	Mean
Case Study: GAN Only on Steam Production Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.06340	0.13000	0.18580	0.23890	01
P_{VAR}	0.0	0.65740	1.0	0.05350	0.13520	0.18790	0.23870	02
P_{VAR}	0.0	0.63460	1.0	0.05790	0.12540	0.17910	0.22900	03
P_{VAR}	0.0	0.66250	1.0	0.05800	0.12840	0.17720	0.23220	04
P_{VAR}	0.0	0.68900	1.0	0.06130	0.17130	0.22280	0.27770	05
P_{VAR}	0.0	0.64830	1.0	0.05480	0.12980	0.18250	0.23710	06
P_{VAR}	0.0	0.66160	1.0	0.06700	0.14600	0.19850	0.24690	07
P_{VAR}	0.0	0.63930	1.0	0.06170	0.13240	0.18250	0.22930	08
P_{VAR}	0.0	0.67320	1.0	0.06330	0.15280	0.20320	0.26050	09
P_{VAR}	0.0	0.64810	1.0	0.05970	0.13140	0.18470	0.23680	10
P_{VAR}	0.0	0.65727	1.0	0.06006	0.13827	0.19042	0.24271	Mean
Case Study: GAN Only on Steam Production Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.38970	0.49590	0.49770	0.53420	01
P_{HAUS}	0.0	0.70670	0.92660	0.52300	0.53600	0.54090	0.56630	02
P_{HAUS}	0.0	0.80610	0.98410	0.40210	0.43220	0.57970	0.69080	03
P_{HAUS}	0.0	0.73450	0.92090	0.41000	0.48590	0.53650	0.57850	04
P_{HAUS}	0.0	0.64090	0.88860	0.34610	0.38600	0.56580	0.69720	05
P_{HAUS}	0.0	0.75500	0.98880	0.40960	0.42720	0.66530	0.66540	06
P_{HAUS}	0.0	0.67750	0.90110	0.34710	0.53240	0.50010	0.57250	07
P_{HAUS}	0.0	0.71480	1.00000	0.42500	0.43410	0.51990	0.53140	08
P_{HAUS}	0.0	0.63940	0.87210	0.32710	0.42090	0.44510	0.53420	09
P_{HAUS}	0.0	0.79240	0.98850	0.42950	0.50270	0.58970	0.53840	10
P_{HAUS}	0.0	0.72235	0.93403	0.40092	0.46533	0.54407	0.59089	Mean

Table D.12 Experiments in Chapter 4.9 - GAN Only - Steam Production

Case Study: GAN Only on Steam Production Dataset: P_{KL} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.01750	0.02890	0.08950	0.14440	01
P_{KL}	0.0	0.30100	0.25430	0.01450	0.03220	0.09650	0.14920	02
P_{KL}	0.0	0.29800	0.26450	0.01840	0.03040	0.08830	0.13380	03
P_{KL}	0.0	0.31620	0.26190	0.02420	0.03910	0.10230	0.15960	04
P_{KL}	0.0	0.34160	0.26850	0.02450	0.05200	0.11580	0.17990	05
P_{KL}	0.0	0.33080	0.26230	0.02100	0.03570	0.11020	0.17870	06
P_{KL}	0.0	0.29520	0.25720	0.01770	0.03500	0.09650	0.13730	07
P_{KL}	0.0	0.29030	0.25810	0.02060	0.03520	0.10260	0.14350	08
P_{KL}	0.0	0.33710	0.25890	0.01820	0.03910	0.10530	0.16850	09
P_{KL}	0.0	0.30220	0.23780	0.01970	0.03350	0.10060	0.16220	10
P_{KL}	0.0	0.31095	0.25746	0.01963	0.03611	0.10076	0.15571	Mean

Case Study: GAN Only on Steam Production Dataset: P_{PRO} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.11140	0.18960	0.53950	0.77080	01
P_{PRO}	0.0	0.99820	0.70340	0.09100	0.19020	0.51320	0.75540	02
P_{PRO}	0.0	0.99830	0.70060	0.10010	0.18200	0.52220	0.75490	03
P_{PRO}	0.0	0.99900	0.69780	0.10200	0.18410	0.50960	0.76480	04
P_{PRO}	0.0	0.99870	0.70310	0.10380	0.22330	0.53790	0.77150	05
P_{PRO}	0.0	0.99900	0.70190	0.09510	0.18620	0.52500	0.77010	06
P_{PRO}	0.0	0.99840	0.69810	0.11850	0.20990	0.53330	0.76280	07
P_{PRO}	0.0	0.99850	0.70110	0.10230	0.18590	0.50860	0.74170	08
P_{PRO}	0.0	0.99830	0.70100	0.11330	0.21230	0.53300	0.77570	09
P_{PRO}	0.0	0.99830	0.70120	0.10380	0.18970	0.52230	0.76270	10
P_{PRO}	0.0	0.99848	0.70044	0.10413	0.19532	0.52446	0.76304	Mean

Case Study: GAN Only on Steam Production Dataset: P_{PCC} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	1.0000	0.44580	0.09020	0.1823	0.27250	0.36010	01
P_{PCC}	0.0	1.0000	0.45020	0.09130	0.1781	0.27210	0.36250	02
P_{PCC}	0.0	0.9941	0.44970	0.09100	0.1812	0.27220	0.36320	03
P_{PCC}	0.0	1.0000	0.44690	0.08990	0.1816	0.27160	0.36240	04
P_{PCC}	0.0	0.9899	0.44380	0.09160	0.1842	0.27310	0.36210	05
P_{PCC}	0.0	0.9901	0.44820	0.08910	0.1791	0.26910	0.35820	06
P_{PCC}	0.0	0.9998	0.44320	0.08970	0.1804	0.27190	0.35990	07
P_{PCC}	0.0	1.0000	0.44510	0.09360	0.1847	0.27570	0.36760	08
P_{PCC}	0.0	0.9930	0.44330	0.08940	0.1823	0.26990	0.35840	09
P_{PCC}	0.0	0.9971	0.45050	0.09150	0.1831	0.27470	0.36170	10
P_{PCC}	0.0	0.9964	0.44667	0.09073	0.1817	0.27228	0.36161	Mean

Table D.13 Experiments in Chapter 4.9 - GAN Only - Energy Efficiency

Case Study: GAN Only on Energy Efficiency Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	1.0	0.01360	0.57300	0.04420	0.01560	0.0004	0.00050	01
<i>R-Squared</i>	1.0	0.00680	0.51680	0.04870	0.04360	0.0472	0.02150	02
<i>R-Squared</i>	1.0	0.06790	0.47280	0.04590	0.04050	0.0307	0.06550	03
<i>R-Squared</i>	1.0	0.03380	0.52350	0.03590	0.03090	0.0246	0.03150	04
<i>R-Squared</i>	1.0	0.02790	0.46580	0.02250	0.01860	0.0141	0.01040	05
<i>R-Squared</i>	1.0	0.01910	0.56580	0.05060	0.01710	0.0193	0.01440	06
<i>R-Squared</i>	1.0	0.02990	0.53150	0.02480	0.03830	0.0340	0.03510	07
<i>R-Squared</i>	1.0	0.00400	0.53060	0.02100	0.01830	0.0283	0.01810	08
<i>R-Squared</i>	1.0	0.00360	0.54820	0.05870	0.04710	0.0430	0.04700	09
<i>R-Squared</i>	1.0	0.00300	0.54070	0.00650	0.00960	0.0044	0.01830	10
<i>R-Squared</i>	1.0	0.02096	0.52687	0.03588	0.02796	0.0246	0.02623	Mean
Case Study: GAN Only on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	1.0	0.53140	0.07950	0.16740	0.27620	0.33890	01
P_{MSE}	0.0	1.0	0.51900	0.07170	0.15190	0.27000	0.33760	02
P_{MSE}	0.0	1.0	0.52300	0.09210	0.17990	0.30130	0.35560	03
P_{MSE}	0.0	1.0	0.49390	0.07690	0.17000	0.28340	0.34410	04
P_{MSE}	0.0	1.0	0.50000	0.08660	0.17720	0.29530	0.36610	05
P_{MSE}	0.0	1.0	0.54010	0.08020	0.16030	0.28270	0.34180	06
P_{MSE}	0.0	1.0	0.48800	0.08400	0.16000	0.26800	0.33600	07
P_{MSE}	0.0	1.0	0.52970	0.08470	0.17370	0.28810	0.33900	08
P_{MSE}	0.0	1.0	0.48250	0.08950	0.17120	0.27240	0.35020	09
P_{MSE}	0.0	1.0	0.50000	0.08800	0.17200	0.28800	0.35600	10
P_{MSE}	0.0	1.0	0.51076	0.08332	0.16836	0.28254	0.34653	Mean
Case Study: GAN Only on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.05380	0.10940	0.15640	0.21020	01
P_{VAR}	0.0	0.52300	0.15090	0.04920	0.09970	0.15020	0.20770	02
P_{VAR}	0.0	0.52600	0.15130	0.05910	0.11390	0.16570	0.21630	03
P_{VAR}	0.0	0.53810	0.15030	0.05450	0.11270	0.16270	0.21610	04
P_{VAR}	0.0	0.55510	0.15130	0.05860	0.11770	0.17180	0.23270	05
P_{VAR}	0.0	0.52160	0.15060	0.05220	0.10190	0.15410	0.20830	06
P_{VAR}	0.0	0.53130	0.15080	0.05850	0.10770	0.15530	0.20960	07
P_{VAR}	0.0	0.51010	0.15130	0.05860	0.11290	0.16240	0.20760	08
P_{VAR}	0.0	0.56190	0.15090	0.06260	0.11880	0.16630	0.22580	09
P_{VAR}	0.0	0.54090	0.15110	0.05860	0.11440	0.16600	0.22540	10
P_{VAR}	0.0	0.53234	0.15096	0.05657	0.11091	0.16109	0.21597	Mean
Case Study: GAN Only on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.53700	0.56350	0.72780	0.70070	01
P_{HAUS}	0.0	0.97250	1.00000	0.52250	0.58100	0.55590	0.60010	02
P_{HAUS}	0.0	0.95840	1.00000	0.57690	0.61270	0.74350	0.80190	03
P_{HAUS}	0.0	0.91530	1.00000	0.57780	0.56780	0.69120	0.72320	04
P_{HAUS}	0.0	0.89050	1.00000	0.48840	0.51730	0.48210	0.53550	05
P_{HAUS}	0.0	0.92740	1.00000	0.53920	0.74300	0.71730	0.76210	06
P_{HAUS}	0.0	0.89190	1.00000	0.50210	0.60910	0.62740	0.64600	07
P_{HAUS}	0.0	0.83290	0.94430	0.53350	0.55770	0.52820	0.63340	08
P_{HAUS}	0.0	0.87520	1.00000	0.47230	0.51360	0.58190	0.68950	09
P_{HAUS}	0.0	0.95280	1.00000	0.52490	0.67220	0.67810	0.69280	10
P_{HAUS}	0.0	0.90552	0.99443	0.52746	0.59379	0.63334	0.67852	Mean

Table D.14 Experiments in Chapter 4.9 - GAN Only - Energy Efficiency

Case Study: GAN Only on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.09210	0.16510	0.26670	0.32700	01
P_{KL}	0.0	1.0	0.32150	0.07400	0.14150	0.25080	0.31830	02
P_{KL}	0.0	1.0	0.34450	0.08360	0.16050	0.29100	0.36450	03
P_{KL}	0.0	1.0	0.33110	0.08940	0.17220	0.29470	0.37750	04
P_{KL}	0.0	1.0	0.34560	0.08390	0.16780	0.27850	0.35230	05
P_{KL}	0.0	1.0	0.34100	0.09840	0.16390	0.28200	0.35410	06
P_{KL}	0.0	1.0	0.32800	0.08040	0.14790	0.26370	0.32480	07
P_{KL}	0.0	1.0	0.33230	0.08710	0.16450	0.28060	0.34520	08
P_{KL}	0.0	1.0	0.33110	0.08610	0.16560	0.27810	0.34770	09
P_{KL}	0.0	1.0	0.31690	0.07690	0.14770	0.25850	0.32000	10
P_{KL}	0.0	1.0	0.33126	0.08519	0.15967	0.27446	0.34314	Mean
Case Study: GAN Only on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.09150	0.24310	0.41740	0.46880	01
P_{PRO}	0.0	0.9946	0.34650	0.07710	0.22330	0.41200	0.46890	02
P_{PRO}	0.0	0.9948	0.34910	0.09800	0.24480	0.43080	0.47720	03
P_{PRO}	0.0	0.9949	0.34690	0.07720	0.23610	0.41920	0.46580	04
P_{PRO}	0.0	0.9966	0.35290	0.09760	0.25290	0.44030	0.49440	05
P_{PRO}	0.0	0.9944	0.35920	0.07900	0.22590	0.41500	0.46610	06
P_{PRO}	0.0	0.9944	0.35240	0.09950	0.24170	0.42660	0.47840	07
P_{PRO}	0.0	0.9941	0.35050	0.09110	0.24710	0.42810	0.47200	08
P_{PRO}	0.0	0.9947	0.34950	0.10870	0.26110	0.43990	0.49230	09
P_{PRO}	0.0	0.9972	0.34910	0.09990	0.25720	0.44010	0.49150	10
P_{PRO}	0.0	0.9950	0.35075	0.09196	0.24332	0.42694	0.47754	Mean
Case Study: GAN Only on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.09630	0.19560	0.29290	0.39020	01
P_{PCC}	0.0	0.97960	0.20190	0.09750	0.19590	0.29640	0.39380	02
P_{PCC}	0.0	0.99240	0.20430	0.10210	0.20630	0.30740	0.40560	03
P_{PCC}	0.0	0.99910	0.20280	0.10790	0.21180	0.31470	0.41560	04
P_{PCC}	0.0	1.00000	0.20360	0.09980	0.20160	0.30340	0.40510	05
P_{PCC}	0.0	0.98320	0.20790	0.10200	0.19800	0.30100	0.39800	06
P_{PCC}	0.0	0.99410	0.20440	0.09870	0.19540	0.29310	0.38890	07
P_{PCC}	0.0	0.97310	0.20410	0.10410	0.20220	0.30220	0.39140	08
P_{PCC}	0.0	1.00000	0.20360	0.10030	0.20160	0.29690	0.39720	09
P_{PCC}	0.0	0.99810	0.20240	0.09870	0.19840	0.29610	0.39680	10
P_{PCC}	0.0	0.98965	0.20385	0.10074	0.20068	0.30041	0.39826	Mean

Table D.15 Experiments in Chapter 4.9 - GAN Only - Synchronous Motors

Case Study: GAN Only on Synchronous Motors Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.634	0.01520	0.08940	0.0257	0.00370	0.01510	0.01850	01
<i>R-Squared</i>	0.634	0.04040	0.07560	0.0852	0.03960	0.48510	0.04430	02
<i>R-Squared</i>	0.634	0.00670	0.06920	0.0111	0.03330	0.02220	0.01860	03
<i>R-Squared</i>	0.634	0.01480	0.13740	0.0010	0.00300	0.02450	0.02170	04
<i>R-Squared</i>	0.634	0.03160	0.03180	0.0018	0.03180	0.04300	0.04260	05
<i>R-Squared</i>	0.634	0.01150	0.08030	0.0132	0.00260	0.00940	0.01260	06
<i>R-Squared</i>	0.634	0.03780	0.01280	0.0161	0.01510	0.00770	0.03620	07
<i>R-Squared</i>	0.634	0.04710	0.20970	0.0251	0.04220	0.02040	0.04760	08
<i>R-Squared</i>	0.634	0.00640	0.13820	0.0061	0.00660	0.09700	0.00640	09
<i>R-Squared</i>	0.634	0.05200	0.12030	0.0157	0.01180	0.07400	0.05700	10
<i>R-Squared</i>	0.634	0.02635	0.09647	0.0201	0.01897	0.07984	0.03055	Mean
Case Study: GAN Only on Synchronous Motors Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.71970	0.47700	0.15900	0.33470	0.47700	0.59410	01
P_{MSE}	0.0	0.76790	0.46410	0.17720	0.34600	0.52320	0.65820	02
P_{MSE}	0.0	0.76990	0.46030	0.16740	0.32640	0.51050	0.66110	03
P_{MSE}	0.0	0.72870	0.45340	0.15380	0.34010	0.51010	0.61540	04
P_{MSE}	0.0	0.72440	0.45670	0.16540	0.32280	0.48030	0.59840	05
P_{MSE}	0.0	0.74260	0.48100	0.16880	0.33760	0.50630	0.62450	06
P_{MSE}	0.0	0.69600	0.45600	0.16800	0.31200	0.45600	0.57600	07
P_{MSE}	0.0	0.70340	0.49150	0.14410	0.29660	0.45760	0.58470	08
P_{MSE}	0.0	0.67700	0.44360	0.17120	0.31910	0.45910	0.56810	09
P_{MSE}	0.0	0.64800	0.45600	0.14400	0.28000	0.43200	0.54400	10
P_{MSE}	0.0	0.71776	0.46396	0.16189	0.32153	0.48121	0.60245	Mean
Case Study: GAN Only on Synchronous Motors Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.10760	0.20510	0.3032	0.40220	01
P_{VAR}	0.0	0.55430	0.18330	0.11810	0.22770	0.3357	0.45130	02
P_{VAR}	0.0	0.53210	0.18390	0.11370	0.20670	0.3102	0.42240	03
P_{VAR}	0.0	0.52760	0.18260	0.10240	0.21490	0.3182	0.42260	04
P_{VAR}	0.0	0.55830	0.18440	0.11270	0.22480	0.3365	0.45020	05
P_{VAR}	0.0	0.55590	0.18360	0.10640	0.21540	0.3309	0.44440	06
P_{VAR}	0.0	0.51540	0.18260	0.09530	0.19320	0.2966	0.40450	07
P_{VAR}	0.0	0.52320	0.18380	0.09190	0.19390	0.3020	0.41820	08
P_{VAR}	0.0	0.55810	0.18320	0.12210	0.23060	0.3310	0.43850	09
P_{VAR}	0.0	0.51130	0.18340	0.09800	0.19660	0.3027	0.41230	10
P_{VAR}	0.0	0.53506	0.18342	0.10682	0.21089	0.3167	0.42666	Mean
Case Study: GAN Only on Synchronous Motors Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.63530	0.35320	0.18820	0.39970	0.46760	0.65080	01
P_{HAUS}	0.0	0.55950	0.43530	0.16650	0.32470	0.41120	0.57670	02
P_{HAUS}	0.0	0.55380	0.43790	0.20010	0.32090	0.39970	0.62570	03
P_{HAUS}	0.0	0.42190	0.42220	0.14830	0.30190	0.32220	0.47490	04
P_{HAUS}	0.0	0.55260	0.35430	0.16540	0.26940	0.29300	0.48180	05
P_{HAUS}	0.0	0.42490	0.43450	0.22660	0.36660	0.38090	0.40320	06
P_{HAUS}	0.0	0.47330	0.39010	0.25490	0.40600	0.43780	0.49690	07
P_{HAUS}	0.0	0.50930	0.43720	0.16190	0.26380	0.37030	0.54820	08
P_{HAUS}	0.0	0.51960	0.39280	0.18430	0.26620	0.34840	0.57020	09
P_{HAUS}	0.0	0.58950	0.37890	0.25840	0.30110	0.41100	0.41840	10
P_{HAUS}	0.0	0.52397	0.40364	0.19546	0.32203	0.38421	0.52468	Mean

Table D.16 Experiments in Chapter 4.9 - GAN Only - Synchronous Motors

Case Study: GAN Only on Synchronous Motors Dataset: P_{KL} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.14600	0.27300	0.40000	0.55870	01
P_{KL}	0.0	0.77810	0.27010	0.18010	0.34730	0.54660	0.65590	02
P_{KL}	0.0	0.74250	0.28760	0.18060	0.37460	0.51510	0.62210	03
P_{KL}	0.0	0.76160	0.30460	0.14570	0.36420	0.49670	0.60260	04
P_{KL}	0.0	0.72480	0.30200	0.14770	0.30870	0.46310	0.57050	05
P_{KL}	0.0	0.68850	0.30160	0.17700	0.31480	0.47210	0.58360	06
P_{KL}	0.0	0.67520	0.27650	0.12860	0.26370	0.41800	0.52090	07
P_{KL}	0.0	0.67740	0.29030	0.12260	0.27740	0.41940	0.52900	08
P_{KL}	0.0	0.71520	0.29800	0.17880	0.32450	0.47680	0.58280	09
P_{KL}	0.0	0.59690	0.27080	0.14150	0.26460	0.40000	0.50460	10
P_{KL}	0.0	0.70205	0.28745	0.15486	0.31128	0.46078	0.57307	Mean

Case Study: GAN Only on Synchronous Motors Dataset: P_{PRO} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.35650	0.54910	0.7742	0.88090	01
P_{PRO}	0.0	0.99820	0.41830	0.35580	0.56160	0.7884	0.89330	02
P_{PRO}	0.0	0.99850	0.40750	0.35750	0.53110	0.7767	0.88900	03
P_{PRO}	0.0	0.99750	0.42730	0.35330	0.56180	0.7828	0.88930	04
P_{PRO}	0.0	0.99790	0.42730	0.37680	0.56910	0.7918	0.89460	05
P_{PRO}	0.0	0.99480	0.42480	0.32870	0.54790	0.7832	0.88510	06
P_{PRO}	0.0	0.99750	0.40320	0.30310	0.50440	0.7577	0.87580	07
P_{PRO}	0.0	0.99550	0.42320	0.32030	0.51850	0.7599	0.87610	08
P_{PRO}	0.0	0.99700	0.43450	0.37870	0.57410	0.7899	0.89220	09
P_{PRO}	0.0	0.99620	0.40890	0.31360	0.51680	0.7604	0.87640	10
P_{PRO}	0.0	0.99706	0.41868	0.34443	0.54344	0.7765	0.88527	Mean

Case Study: GAN Only on Synchronous Motors Dataset: P_{PCC} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.1926	0.37920	0.5698	0.7724	01
P_{PCC}	0.0	0.99250	0.23870	0.2089	0.40580	0.6067	0.7976	02
P_{PCC}	0.0	0.99140	0.23430	0.2043	0.40260	0.5988	0.7871	03
P_{PCC}	0.0	0.98510	0.24580	0.1858	0.38570	0.5855	0.7733	04
P_{PCC}	0.0	0.99410	0.23910	0.1877	0.39130	0.5909	0.7846	05
P_{PCC}	0.0	1.00000	0.23960	0.2059	0.39600	0.6119	0.8020	06
P_{PCC}	0.0	0.99710	0.22930	0.1994	0.39490	0.5963	0.7977	07
P_{PCC}	0.0	0.96320	0.23980	0.1724	0.36860	0.5609	0.7611	08
P_{PCC}	0.0	0.99300	0.24630	0.2026	0.39920	0.5919	0.7865	09
P_{PCC}	0.0	0.96920	0.23330	0.1974	0.38690	0.5823	0.7837	10
P_{PCC}	0.0	0.98446	0.23785	0.1957	0.39102	0.5895	0.7846	Mean

Table D.17 Experiments in Chapter 4.9 - DP Only - Wind Turbine

Case Study: DP Only on Wind Turbine Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.9343	0.00580	0.64670	0.85120	0.83320	0.81680	0.79880	01
<i>R-Squared</i>	0.9343	0.00110	0.64910	0.85880	0.84190	0.82310	0.80060	02
<i>R-Squared</i>	0.9343	0.00100	0.64980	0.85450	0.84090	0.82190	0.79780	03
<i>R-Squared</i>	0.9343	0.00170	0.64640	0.85200	0.83570	0.81760	0.79440	04
<i>R-Squared</i>	0.9343	0.00110	0.65070	0.84930	0.83490	0.81290	0.78560	05
<i>R-Squared</i>	0.9343	0.01260	0.65920	0.85720	0.84010	0.82400	0.80280	06
<i>R-Squared</i>	0.9343	0.00510	0.63870	0.85340	0.83410	0.81610	0.79420	07
<i>R-Squared</i>	0.9343	0.00050	0.63990	0.85210	0.84130	0.82070	0.80140	08
<i>R-Squared</i>	0.9343	0.00890	0.64940	0.85150	0.83550	0.81590	0.79410	09
<i>R-Squared</i>	0.9343	0.00930	0.65130	0.85590	0.83890	0.81740	0.79340	10
<i>R-Squared</i>	0.9343	0.00471	0.64812	0.85359	0.83765	0.81864	0.79631	Mean
Case Study: DP Only on Wind Turbine Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.75310	0.23890	0.00590	0.04180	0.0598	0.07780	01
P_{MSE}	0.0	0.82570	0.24090	0.00590	0.04220	0.0603	0.07850	02
P_{MSE}	0.0	0.85480	0.23890	0.00590	0.04180	0.0598	0.07780	03
P_{MSE}	0.0	0.82150	0.22550	0.00570	0.04050	0.0579	0.07530	04
P_{MSE}	0.0	0.69170	0.22480	0.00550	0.03940	0.0563	0.07320	05
P_{MSE}	0.0	0.78950	0.24090	0.00590	0.04220	0.0603	0.07850	06
P_{MSE}	0.0	0.70280	0.22840	0.00560	0.04000	0.0572	0.07440	07
P_{MSE}	0.0	0.82920	0.23600	0.00590	0.04240	0.0606	0.07880	08
P_{MSE}	0.0	0.73390	0.22220	0.00540	0.03890	0.0556	0.07240	09
P_{MSE}	0.0	0.76000	0.22840	0.00560	0.04000	0.0572	0.07440	10
P_{MSE}	0.0	0.77622	0.23249	0.00573	0.04092	0.0585	0.07611	Mean
Case Study: DP Only on Wind Turbine Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.59590	0.10680	0.0036	0.0249	0.05670	0.06280	01
P_{VAR}	0.0	0.63820	0.10670	0.0036	0.0249	0.05660	0.06270	02
P_{VAR}	0.0	0.66980	0.10650	0.0036	0.0249	0.05670	0.06280	03
P_{VAR}	0.0	0.70830	0.10660	0.0036	0.0249	0.05660	0.06270	04
P_{VAR}	0.0	0.58470	0.10680	0.0036	0.0249	0.05670	0.06280	05
P_{VAR}	0.0	0.60790	0.10670	0.0036	0.0249	0.05660	0.06270	06
P_{VAR}	0.0	0.59730	0.10660	0.0036	0.0249	0.05660	0.06270	07
P_{VAR}	0.0	0.66230	0.10640	0.0036	0.0249	0.05660	0.06280	08
P_{VAR}	0.0	0.62930	0.10670	0.0036	0.0249	0.05660	0.06270	09
P_{VAR}	0.0	0.65040	0.10680	0.0036	0.0249	0.05670	0.06280	10
P_{VAR}	0.0	0.63441	0.10666	0.0036	0.0249	0.05664	0.06275	Mean
Case Study: DP Only on Wind Turbine Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.49950	0.40370	0.18190	0.38460	0.38460	0.31010	01
P_{HAUS}	0.0	0.36360	0.34300	0.16030	0.33550	0.33550	0.32210	02
P_{HAUS}	0.0	0.47530	0.39450	0.19060	0.35420	0.44510	0.39450	03
P_{HAUS}	0.0	0.41860	0.37460	0.17060	0.40880	0.42710	0.37340	04
P_{HAUS}	0.0	0.50480	0.36730	0.14970	0.35040	0.35040	0.33240	05
P_{HAUS}	0.0	0.46330	0.37670	0.16100	0.36550	0.36550	0.27450	06
P_{HAUS}	0.0	0.52240	0.40340	0.17350	0.23800	0.31380	0.26950	07
P_{HAUS}	0.0	0.37270	0.29740	0.18360	0.32420	0.32420	0.30400	08
P_{HAUS}	0.0	0.37460	0.31120	0.15020	0.33110	0.34710	0.32970	09
P_{HAUS}	0.0	0.47310	0.36960	0.17640	0.40790	0.42060	0.33570	10
P_{HAUS}	0.0	0.44679	0.36414	0.16978	0.35002	0.37139	0.32459	Mean

Table D.18 Experiments in Chapter 4.9 - DP Only - Wind Turbine

Case Study: DP Only on Wind Turbine Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	0.92060	0.23140	0.01810	0.0498	0.06790	0.0952	01
P_{KL}	0.0	0.93250	0.23890	0.01830	0.0505	0.06880	0.0965	02
P_{KL}	0.0	0.96520	0.24850	0.01910	0.0525	0.07160	0.1003	03
P_{KL}	0.0	0.97910	0.24600	0.02350	0.0566	0.07580	0.1040	04
P_{KL}	0.0	0.94430	0.24460	0.01910	0.0527	0.07180	0.1007	05
P_{KL}	0.0	0.92260	0.24820	0.01870	0.0515	0.07020	0.0984	06
P_{KL}	0.0	0.90030	0.23440	0.01830	0.0505	0.06880	0.0965	07
P_{KL}	0.0	0.94940	0.23970	0.01840	0.0506	0.06900	0.0968	08
P_{KL}	0.0	0.97910	0.24140	0.01890	0.0520	0.07090	0.0993	09
P_{KL}	0.0	0.89660	0.22430	0.01750	0.0483	0.06580	0.0923	10
P_{KL}	0.0	0.93897	0.23974	0.01899	0.0515	0.07006	0.0980	Mean
Case Study: DP Only on Wind Turbine Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	1.0	0.25390	0.0095	0.0628	0.08520	0.10680	01
P_{PRO}	0.0	1.0	0.25540	0.0095	0.0630	0.08530	0.10690	02
P_{PRO}	0.0	1.0	0.25470	0.0095	0.0627	0.08510	0.10670	03
P_{PRO}	0.0	1.0	0.25300	0.0095	0.0627	0.08520	0.10650	04
P_{PRO}	0.0	1.0	0.25500	0.0095	0.0629	0.08520	0.10700	05
P_{PRO}	0.0	1.0	0.25460	0.0095	0.0628	0.08510	0.10690	06
P_{PRO}	0.0	1.0	0.25440	0.0095	0.0627	0.08500	0.10640	07
P_{PRO}	0.0	1.0	0.25450	0.0095	0.0629	0.08520	0.10660	08
P_{PRO}	0.0	1.0	0.25460	0.0095	0.0627	0.08500	0.10670	09
P_{PRO}	0.0	1.0	0.25430	0.0095	0.0628	0.08510	0.10670	10
P_{PRO}	0.0	1.0	0.25444	0.0095	0.0628	0.08514	0.10672	Mean
Case Study: DP Only on Wind Turbine Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.99700	0.15030	0.00570	0.03540	0.07380	0.08510	01
P_{PCC}	0.0	0.99160	0.15350	0.00710	0.03690	0.07530	0.08660	02
P_{PCC}	0.0	1.00000	0.15310	0.00570	0.03580	0.07440	0.08580	03
P_{PCC}	0.0	0.99770	0.15130	0.00570	0.03570	0.07420	0.08560	04
P_{PCC}	0.0	0.98680	0.14960	0.00560	0.03530	0.07340	0.08470	05
P_{PCC}	0.0	0.99300	0.15140	0.00560	0.03530	0.07360	0.08490	06
P_{PCC}	0.0	1.00000	0.14960	0.00570	0.03420	0.07270	0.08410	07
P_{PCC}	0.0	0.98390	0.15150	0.00560	0.03540	0.07360	0.08490	08
P_{PCC}	0.0	0.99440	0.14900	0.00570	0.03410	0.07240	0.08370	09
P_{PCC}	0.0	1.00000	0.14960	0.00570	0.03420	0.07270	0.08410	10
P_{PCC}	0.0	0.99444	0.15089	0.00581	0.03523	0.07361	0.08495	Mean

Table D.19 Experiments in Chapter 4.9 - DP Only - Steam Production

Case Study: DP Only on Steam Production Dataset: R-Squared values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
<i>R-Squared</i>	0.6585	0.01050	0.03590	0.11290	0.11570	0.11910	0.13650	01
<i>R-Squared</i>	0.6585	0.00510	0.06220	0.21180	0.21370	0.21560	0.21760	02
<i>R-Squared</i>	0.6585	0.00450	0.06200	0.21900	0.21400	0.21790	0.22440	03
<i>R-Squared</i>	0.6585	0.01470	0.07880	0.21460	0.21840	0.21840	0.21220	04
<i>R-Squared</i>	0.6585	0.00360	0.06780	0.20200	0.20370	0.20640	0.20830	05
<i>R-Squared</i>	0.6585	0.01750	0.05710	0.15870	0.16170	0.16380	0.18670	06
<i>R-Squared</i>	0.6585	0.00450	0.06250	0.24080	0.24630	0.23930	0.24060	07
<i>R-Squared</i>	0.6585	0.00260	0.06640	0.21650	0.21440	0.21060	0.20410	08
<i>R-Squared</i>	0.6585	0.01630	0.02050	0.15930	0.15850	0.15960	0.16400	09
<i>R-Squared</i>	0.6585	0.00200	0.07440	0.24770	0.24920	0.24700	0.24630	10
<i>R-Squared</i>	0.6585	0.00813	0.05876	0.19833	0.19956	0.19977	0.20407	Mean

Case Study: DP Only on Steam Production Dataset: P_{MSE} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.30040	0.5477	0.02680	0.09500	0.12180	0.18620	01
P_{MSE}	0.0	0.29920	0.5409	0.02320	0.08820	0.11520	0.18020	02
P_{MSE}	0.0	0.28540	0.5402	0.02300	0.09120	0.11800	0.18240	03
P_{MSE}	0.0	0.27980	0.5154	0.02230	0.08460	0.11050	0.17290	04
P_{MSE}	0.0	0.29330	0.4976	0.02170	0.08230	0.10750	0.16810	05
P_{MSE}	0.0	0.31050	0.5447	0.02320	0.09200	0.11900	0.18400	06
P_{MSE}	0.0	0.27640	0.5164	0.02200	0.08720	0.11280	0.17440	07
P_{MSE}	0.0	0.27750	0.5432	0.02330	0.09240	0.11950	0.18470	08
P_{MSE}	0.0	0.28990	0.4953	0.02140	0.08480	0.10970	0.16960	09
P_{MSE}	0.0	0.28720	0.5056	0.02200	0.08360	0.10920	0.17080	10
P_{MSE}	0.0	0.28996	0.5247	0.02289	0.08813	0.11432	0.17733	Mean

Case Study: DP Only on Steam Production Dataset: P_{VAR} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.65870	1.0	0.04800	0.26020	0.27310	0.30360	01
P_{VAR}	0.0	0.65740	1.0	0.04800	0.25950	0.27210	0.30270	02
P_{VAR}	0.0	0.63460	1.0	0.04800	0.26020	0.27290	0.30340	03
P_{VAR}	0.0	0.66250	1.0	0.04790	0.26000	0.27280	0.30330	04
P_{VAR}	0.0	0.68900	1.0	0.04800	0.25980	0.27260	0.30320	05
P_{VAR}	0.0	0.64830	1.0	0.04790	0.26050	0.27330	0.30380	06
P_{VAR}	0.0	0.66160	1.0	0.04790	0.26020	0.27280	0.30330	07
P_{VAR}	0.0	0.63930	1.0	0.04800	0.25960	0.27250	0.30300	08
P_{VAR}	0.0	0.67320	1.0	0.04790	0.26030	0.27320	0.30370	09
P_{VAR}	0.0	0.64810	1.0	0.04800	0.26000	0.27290	0.30340	10
P_{VAR}	0.0	0.65727	1.0	0.04796	0.26003	0.27282	0.30334	Mean

Case Study: DP Only on Steam Production Dataset: P_{HAUS} values

Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.75620	0.86960	0.38260	0.61780	0.61780	0.61780	01
P_{HAUS}	0.0	0.70670	0.92660	0.37440	0.51710	0.64130	0.65460	02
P_{HAUS}	0.0	0.80610	0.98410	0.40450	0.49470	0.59600	0.82760	03
P_{HAUS}	0.0	0.73450	0.92090	0.36420	0.59600	0.60080	0.65050	04
P_{HAUS}	0.0	0.64090	0.88860	0.36020	0.47900	0.59720	0.71020	05
P_{HAUS}	0.0	0.75500	0.98880	0.35440	0.66910	0.66910	0.75460	06
P_{HAUS}	0.0	0.67750	0.90110	0.37220	0.51060	0.51400	0.58350	07
P_{HAUS}	0.0	0.71480	1.00000	0.36360	0.58910	0.61240	0.62290	08
P_{HAUS}	0.0	0.63940	0.87210	0.34940	0.69270	0.70670	0.71140	09
P_{HAUS}	0.0	0.79240	0.98850	0.40220	0.73680	0.73680	0.73680	10
P_{HAUS}	0.0	0.72235	0.93403	0.37277	0.59029	0.62921	0.68699	Mean

Table D.20 Experiments in Chapter 4.9 - DP Only - Steam Production

Case Study: DP Only on Steam Production Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	0.29710	0.25110	0.01750	0.0432	0.0635	0.09240	01
P_{KL}	0.0	0.30100	0.25430	0.01770	0.0437	0.0643	0.09360	02
P_{KL}	0.0	0.29800	0.26450	0.01840	0.0455	0.0669	0.09730	03
P_{KL}	0.0	0.31620	0.26190	0.01820	0.0450	0.0662	0.09640	04
P_{KL}	0.0	0.34160	0.26850	0.01850	0.0456	0.0671	0.09770	05
P_{KL}	0.0	0.33080	0.26230	0.01800	0.0475	0.0685	0.09840	06
P_{KL}	0.0	0.29520	0.25720	0.01770	0.0437	0.0643	0.09360	07
P_{KL}	0.0	0.29030	0.25810	0.01770	0.0439	0.0645	0.09390	08
P_{KL}	0.0	0.33710	0.25890	0.01490	0.0421	0.0632	0.09340	09
P_{KL}	0.0	0.30220	0.23780	0.01690	0.0418	0.0615	0.08950	10
P_{KL}	0.0	0.31095	0.25746	0.01755	0.0442	0.0650	0.09462	Mean
Case Study: DP Only on Steam Production Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.99810	0.69620	0.09600	0.28930	0.34290	0.44430	01
P_{PRO}	0.0	0.99820	0.70340	0.09550	0.28750	0.34510	0.45030	02
P_{PRO}	0.0	0.99830	0.70060	0.09500	0.28600	0.33970	0.44410	03
P_{PRO}	0.0	0.99900	0.69780	0.09540	0.28840	0.34430	0.44720	04
P_{PRO}	0.0	0.99870	0.70310	0.09570	0.28610	0.34220	0.44770	05
P_{PRO}	0.0	0.99900	0.70190	0.09590	0.28930	0.34680	0.44850	06
P_{PRO}	0.0	0.99840	0.69810	0.09540	0.28700	0.34070	0.44500	07
P_{PRO}	0.0	0.99850	0.70110	0.09580	0.28800	0.34400	0.45020	08
P_{PRO}	0.0	0.99830	0.70100	0.09500	0.28670	0.34230	0.44800	09
P_{PRO}	0.0	0.99830	0.70120	0.09550	0.28750	0.34050	0.44430	10
P_{PRO}	0.0	0.99848	0.70044	0.09552	0.28758	0.34285	0.44696	Mean
Case Study: DP Only on Steam Production Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	1.0000	0.44580	0.04060	0.10370	0.12090	0.15070	01
P_{PCC}	0.0	1.0000	0.45020	0.03890	0.10210	0.12110	0.15280	02
P_{PCC}	0.0	0.9941	0.44970	0.03740	0.09920	0.11660	0.14750	03
P_{PCC}	0.0	1.0000	0.44690	0.03910	0.10260	0.12080	0.15170	04
P_{PCC}	0.0	0.9899	0.44380	0.03860	0.09880	0.11680	0.14820	05
P_{PCC}	0.0	0.9901	0.44820	0.03960	0.10170	0.12060	0.15030	06
P_{PCC}	0.0	0.9998	0.44320	0.03810	0.09970	0.11700	0.14780	07
P_{PCC}	0.0	1.0000	0.44510	0.03960	0.10180	0.11980	0.15130	08
P_{PCC}	0.0	0.9930	0.44330	0.03700	0.09930	0.11740	0.14900	09
P_{PCC}	0.0	0.9971	0.45050	0.03900	0.10150	0.11880	0.14960	10
P_{PCC}	0.0	0.9964	0.44667	0.03879	0.10104	0.11898	0.14989	Mean

Table D.21 Experiments in Chapter 4.9 - DP Only - Energy Efficiency

Case Study: DP Only on Energy Efficiency Dataset: R -Squared values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
R -Squared	1.0	0.01360	0.57300	0.68380	0.68350	0.68570	0.6839	01
R -Squared	1.0	0.00680	0.51680	0.67370	0.67520	0.66930	0.6705	02
R -Squared	1.0	0.06790	0.47280	0.63390	0.61270	0.63710	0.6364	03
R -Squared	1.0	0.03380	0.52350	0.66440	0.66220	0.66520	0.6651	04
R -Squared	1.0	0.02790	0.46580	0.63700	0.63440	0.62580	0.6289	05
R -Squared	1.0	0.01910	0.56580	0.68160	0.68130	0.68140	0.6795	06
R -Squared	1.0	0.02990	0.53150	0.67600	0.67600	0.67410	0.6755	07
R -Squared	1.0	0.00400	0.53060	0.66890	0.67030	0.66680	0.6702	08
R -Squared	1.0	0.00360	0.54820	0.68680	0.68720	0.68960	0.6926	09
R -Squared	1.0	0.00300	0.54070	0.71870	0.72060	0.71970	0.7154	10
R -Squared	1.0	0.02096	0.52687	0.67248	0.67034	0.67147	0.6718	Mean
Case Study: DP Only on Energy Efficiency Dataset: P_{MSE} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{MSE}	0.0	1.0	0.53140	0.03350	0.10460	0.15480	0.1841	01
P_{MSE}	0.0	1.0	0.51900	0.03380	0.10130	0.15190	0.1772	02
P_{MSE}	0.0	1.0	0.52300	0.03350	0.09620	0.15060	0.1799	03
P_{MSE}	0.0	1.0	0.49390	0.03240	0.09720	0.14980	0.1781	04
P_{MSE}	0.0	1.0	0.50000	0.03150	0.09450	0.14570	0.1732	05
P_{MSE}	0.0	1.0	0.54010	0.03800	0.10970	0.16460	0.1941	06
P_{MSE}	0.0	1.0	0.48800	0.03200	0.09600	0.14800	0.1720	07
P_{MSE}	0.0	1.0	0.52970	0.03390	0.10590	0.15680	0.1822	08
P_{MSE}	0.0	1.0	0.48250	0.03110	0.09340	0.14400	0.1712	09
P_{MSE}	0.0	1.0	0.50000	0.03200	0.09600	0.14800	0.1760	10
P_{MSE}	0.0	1.0	0.51076	0.03317	0.09948	0.15142	0.1788	Mean
Case Study: DP Only on Energy Efficiency Dataset: P_{VAR} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{VAR}	0.0	0.51540	0.15110	0.01190	0.03260	0.04400	0.05710	01
P_{VAR}	0.0	0.52300	0.15090	0.01190	0.03260	0.04390	0.05700	02
P_{VAR}	0.0	0.52600	0.15130	0.01190	0.03260	0.04400	0.05710	03
P_{VAR}	0.0	0.53810	0.15030	0.01160	0.03230	0.04360	0.05670	04
P_{VAR}	0.0	0.55510	0.15130	0.01190	0.03260	0.04400	0.05710	05
P_{VAR}	0.0	0.52160	0.15060	0.01160	0.03230	0.04360	0.05680	06
P_{VAR}	0.0	0.53130	0.15080	0.01190	0.03250	0.04390	0.05700	07
P_{VAR}	0.0	0.51010	0.15130	0.01160	0.03260	0.04390	0.05710	08
P_{VAR}	0.0	0.56190	0.15090	0.01160	0.03230	0.04360	0.05680	09
P_{VAR}	0.0	0.54090	0.15110	0.01160	0.03230	0.04370	0.05680	10
P_{VAR}	0.0	0.53234	0.15096	0.01175	0.03247	0.04382	0.05695	Mean
Case Study: DP Only on Energy Efficiency Dataset: P_{HAUS} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{HAUS}	0.0	0.83830	1.00000	0.42120	0.62210	0.65830	0.6583	01
P_{HAUS}	0.0	0.97250	1.00000	0.41080	0.53360	0.54410	0.5669	02
P_{HAUS}	0.0	0.95840	1.00000	0.46790	0.53240	0.60780	0.6249	03
P_{HAUS}	0.0	0.91530	1.00000	0.41750	0.52020	0.54950	0.6069	04
P_{HAUS}	0.0	0.89050	1.00000	0.38230	0.52420	0.64370	0.6437	05
P_{HAUS}	0.0	0.92740	1.00000	0.41140	0.53190	0.66780	0.6678	06
P_{HAUS}	0.0	0.89190	1.00000	0.40210	0.57130	0.66270	0.6978	07
P_{HAUS}	0.0	0.83290	0.94430	0.40040	0.53160	0.53480	0.5569	08
P_{HAUS}	0.0	0.87520	1.00000	0.39780	0.61680	0.64210	0.7148	09
P_{HAUS}	0.0	0.95280	1.00000	0.43770	0.56460	0.75900	0.7590	10
P_{HAUS}	0.0	0.90552	0.99443	0.41491	0.55487	0.62698	0.6497	Mean

Table D.22 Experiments in Chapter 4.9 - DP Only - Energy Efficiency

Case Study: DP Only on Energy Efficiency Dataset: P_{KL} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{KL}	0.0	1.0	0.32060	0.02540	0.06030	0.09520	0.11750	01
P_{KL}	0.0	1.0	0.32150	0.02570	0.05790	0.09320	0.11580	02
P_{KL}	0.0	1.0	0.34450	0.03010	0.06350	0.10370	0.12710	03
P_{KL}	0.0	1.0	0.33110	0.02980	0.06620	0.10600	0.12910	04
P_{KL}	0.0	1.0	0.34560	0.03020	0.06380	0.10400	0.12750	05
P_{KL}	0.0	1.0	0.34100	0.02950	0.06560	0.10490	0.13110	06
P_{KL}	0.0	1.0	0.32800	0.02570	0.05790	0.09970	0.11900	07
P_{KL}	0.0	1.0	0.33230	0.02580	0.06130	0.09680	0.11940	08
P_{KL}	0.0	1.0	0.33110	0.02650	0.05960	0.09930	0.12250	09
P_{KL}	0.0	1.0	0.31690	0.02460	0.05850	0.09230	0.11380	10
P_{KL}	0.0	1.0	0.33126	0.02733	0.06146	0.09951	0.12228	Mean
Case Study: DP Only on Energy Efficiency Dataset: P_{PRO} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PRO}	0.0	0.9943	0.35140	0.03380	0.09370	0.13620	0.15720	01
P_{PRO}	0.0	0.9946	0.34650	0.03380	0.09360	0.13570	0.15770	02
P_{PRO}	0.0	0.9948	0.34910	0.03400	0.09380	0.13600	0.15710	03
P_{PRO}	0.0	0.9949	0.34690	0.03350	0.09430	0.13730	0.15880	04
P_{PRO}	0.0	0.9966	0.35290	0.03380	0.09470	0.13810	0.15960	05
P_{PRO}	0.0	0.9944	0.35920	0.03350	0.09420	0.13760	0.15970	06
P_{PRO}	0.0	0.9944	0.35240	0.03400	0.09410	0.13780	0.15890	07
P_{PRO}	0.0	0.9941	0.35050	0.03350	0.09350	0.13540	0.15670	08
P_{PRO}	0.0	0.9947	0.34950	0.03380	0.09370	0.13620	0.15730	09
P_{PRO}	0.0	0.9972	0.34910	0.03360	0.09390	0.13650	0.15750	10
P_{PRO}	0.0	0.9950	0.35075	0.03373	0.09395	0.13668	0.15805	Mean
Case Study: DP Only on Energy Efficiency Dataset: P_{PCC} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{PCC}	0.0	0.97690	0.20350	0.0169	0.04170	0.05860	0.07640	01
P_{PCC}	0.0	0.97960	0.20190	0.0169	0.04180	0.05870	0.07860	02
P_{PCC}	0.0	0.99240	0.20430	0.0180	0.04210	0.05910	0.07710	03
P_{PCC}	0.0	0.99910	0.20280	0.0180	0.04400	0.06090	0.07990	04
P_{PCC}	0.0	1.00000	0.20360	0.0178	0.04350	0.06030	0.07910	05
P_{PCC}	0.0	0.98320	0.20790	0.0178	0.04360	0.06040	0.08020	06
P_{PCC}	0.0	0.99410	0.20440	0.0170	0.04190	0.05980	0.07780	07
P_{PCC}	0.0	0.97310	0.20410	0.0168	0.04160	0.05850	0.07730	08
P_{PCC}	0.0	1.00000	0.20360	0.0169	0.04170	0.05860	0.07650	09
P_{PCC}	0.0	0.99810	0.20240	0.0179	0.04390	0.06080	0.07880	10
P_{PCC}	0.0	0.98965	0.20385	0.0174	0.04258	0.05957	0.07817	Mean

Table D.23 Experiments in Chapter 4.9 - DP Only - Synchronous Motors

Case Study: DP Only on Synchronous Motors Dataset: R-Squared values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
$R\text{-Squared}$	0.634	0.01520	0.08940	0.36140	0.36700	0.16440	0.15510	01
$R\text{-Squared}$	0.634	0.04040	0.07560	0.38970	0.35790	0.16470	0.18040	02
$R\text{-Squared}$	0.634	0.00670	0.06920	0.38960	0.36740	0.15160	0.14860	03
$R\text{-Squared}$	0.634	0.01480	0.13740	0.32510	0.32490	0.23810	0.23610	04
$R\text{-Squared}$	0.634	0.03160	0.03180	0.33760	0.33530	0.05610	0.10740	05
$R\text{-Squared}$	0.634	0.01150	0.08030	0.21210	0.25240	0.11710	0.10260	06
$R\text{-Squared}$	0.634	0.03780	0.01280	0.39000	0.37210	0.15330	0.16480	07
$R\text{-Squared}$	0.634	0.04710	0.20970	0.42360	0.42130	0.31330	0.29790	08
$R\text{-Squared}$	0.634	0.00640	0.13820	0.33110	0.33370	0.18120	0.18700	09
$R\text{-Squared}$	0.634	0.05200	0.12030	0.31540	0.30120	0.12980	0.10520	10
$R\text{-Squared}$	0.634	0.02635	0.09647	0.34756	0.34332	0.16696	0.16851	Mean
Case Study: DP Only on Synchronous Motors Dataset: P_{MSE} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{MSE}	0.0	0.71970	0.47700	0.11720	0.21760	0.31800	0.39330	01
P_{MSE}	0.0	0.76790	0.46410	0.10130	0.20250	0.31220	0.38820	02
P_{MSE}	0.0	0.76990	0.46030	0.10040	0.20920	0.30960	0.38490	03
P_{MSE}	0.0	0.72870	0.45340	0.09720	0.19430	0.30770	0.38060	04
P_{MSE}	0.0	0.72440	0.45670	0.10240	0.20470	0.30710	0.37800	05
P_{MSE}	0.0	0.74260	0.48100	0.10970	0.21940	0.32910	0.40510	06
P_{MSE}	0.0	0.69600	0.45600	0.10400	0.20800	0.30400	0.37600	07
P_{MSE}	0.0	0.70340	0.49150	0.11020	0.22030	0.33050	0.41530	08
P_{MSE}	0.0	0.67700	0.44360	0.10120	0.20230	0.30350	0.37350	09
P_{MSE}	0.0	0.64800	0.45600	0.10400	0.20800	0.30400	0.38400	10
P_{MSE}	0.0	0.71776	0.46396	0.10476	0.20863	0.31257	0.38789	Mean
Case Study: DP Only on Synchronous Motors Dataset: P_{VAR} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{VAR}	0.0	0.51440	0.18340	0.03490	0.07120	0.10710	0.14500	01
P_{VAR}	0.0	0.55430	0.18330	0.03530	0.07120	0.10700	0.14540	02
P_{VAR}	0.0	0.53210	0.18390	0.03540	0.07170	0.10760	0.14550	03
P_{VAR}	0.0	0.52760	0.18260	0.03530	0.07110	0.10640	0.14470	04
P_{VAR}	0.0	0.55830	0.18440	0.03590	0.07170	0.10760	0.14600	05
P_{VAR}	0.0	0.55590	0.18360	0.03530	0.07110	0.10690	0.14530	06
P_{VAR}	0.0	0.51540	0.18260	0.03530	0.07110	0.10640	0.14470	07
P_{VAR}	0.0	0.52320	0.18380	0.03540	0.07120	0.10710	0.14550	08
P_{VAR}	0.0	0.55810	0.18320	0.03530	0.07110	0.10700	0.14480	09
P_{VAR}	0.0	0.51130	0.18340	0.03540	0.07120	0.10710	0.14500	10
P_{VAR}	0.0	0.53506	0.18342	0.03535	0.07126	0.10702	0.14519	Mean
Case Study: DP Only on Synchronous Motors Dataset: P_{HAUS} values								
Test	Original	GAN	DP	($n_s=1$)	($n_s=2$)	($n_s=3$)	($n_s=4$)	Test Nb.
P_{HAUS}	0.0	0.63530	0.35320	0.15980	0.20240	0.26280	0.31300	01
P_{HAUS}	0.0	0.55950	0.43530	0.18750	0.29140	0.30180	0.35610	02
P_{HAUS}	0.0	0.55380	0.43790	0.12860	0.30990	0.32640	0.32640	03
P_{HAUS}	0.0	0.42190	0.42220	0.13270	0.21810	0.27280	0.36090	04
P_{HAUS}	0.0	0.55260	0.35430	0.16040	0.19460	0.24770	0.27870	05
P_{HAUS}	0.0	0.42490	0.43450	0.14540	0.19840	0.24600	0.31420	06
P_{HAUS}	0.0	0.47330	0.39010	0.17160	0.22740	0.25310	0.37850	07
P_{HAUS}	0.0	0.50930	0.43720	0.15950	0.18620	0.21590	0.39500	08
P_{HAUS}	0.0	0.51960	0.39280	0.13360	0.19420	0.24570	0.28450	09
P_{HAUS}	0.0	0.58950	0.37890	0.13620	0.22470	0.24890	0.37700	10
P_{HAUS}	0.0	0.52397	0.40364	0.15153	0.22473	0.26211	0.33843	Mean

Table D.24 Experiments in Chapter 4.9 - DP Only - Synchronous Motors

Case Study: DP Only on Synchronous Motors Dataset: P_{KL} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{KL}	0.0	0.66030	0.27300	0.0635	0.12060	0.17780	0.22220	01
P_{KL}	0.0	0.77810	0.27010	0.0579	0.11580	0.18010	0.22510	02
P_{KL}	0.0	0.74250	0.28760	0.0602	0.12710	0.18730	0.23410	03
P_{KL}	0.0	0.76160	0.30460	0.0662	0.13250	0.19870	0.25170	04
P_{KL}	0.0	0.72480	0.30200	0.0671	0.13420	0.20130	0.24830	05
P_{KL}	0.0	0.68850	0.30160	0.0656	0.13110	0.19670	0.24920	06
P_{KL}	0.0	0.67520	0.27650	0.0643	0.12220	0.18010	0.22510	07
P_{KL}	0.0	0.67740	0.29030	0.0645	0.12260	0.18710	0.23870	08
P_{KL}	0.0	0.71520	0.29800	0.0662	0.13250	0.19870	0.24500	09
P_{KL}	0.0	0.59690	0.27080	0.0615	0.12310	0.17850	0.22770	10
P_{KL}	0.0	0.70205	0.28745	0.0637	0.12617	0.18863	0.23671	Mean
Case Study: DP Only on Synchronous Motors Dataset: P_{PRO} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PRO}	0.0	0.99750	0.41180	0.13780	0.24410	0.32200	0.37120	01
P_{PRO}	0.0	0.99820	0.41830	0.13930	0.24620	0.33130	0.37990	02
P_{PRO}	0.0	0.99850	0.40750	0.13650	0.24010	0.31790	0.36510	03
P_{PRO}	0.0	0.99750	0.42730	0.13900	0.25020	0.33370	0.38210	04
P_{PRO}	0.0	0.99790	0.42730	0.13950	0.24660	0.33050	0.38070	05
P_{PRO}	0.0	0.99480	0.42480	0.13700	0.24270	0.33120	0.38180	06
P_{PRO}	0.0	0.99750	0.40320	0.13930	0.24180	0.31730	0.36250	07
P_{PRO}	0.0	0.99550	0.42320	0.14260	0.24470	0.33010	0.38100	08
P_{PRO}	0.0	0.99700	0.43450	0.14140	0.24980	0.33990	0.38810	09
P_{PRO}	0.0	0.99620	0.40890	0.14000	0.24230	0.32140	0.36830	10
P_{PRO}	0.0	0.99706	0.41868	0.13924	0.24485	0.32753	0.37607	Mean
Case Study: DP Only on Synchronous Motors Dataset: P_{PCC} values								
Test	Original	GAN	DP	$(n_s=1)$	$(n_s=2)$	$(n_s=3)$	$(n_s=4)$	Test Nb.
P_{PCC}	0.0	0.95900	0.23230	0.04570	0.09330	0.13700	0.18470	01
P_{PCC}	0.0	0.99250	0.23870	0.04770	0.09750	0.14520	0.19290	02
P_{PCC}	0.0	0.99140	0.23430	0.04410	0.09010	0.13620	0.18430	03
P_{PCC}	0.0	0.98510	0.24580	0.04800	0.09990	0.14790	0.19580	04
P_{PCC}	0.0	0.99410	0.23910	0.04550	0.09290	0.14030	0.18770	05
P_{PCC}	0.0	1.00000	0.23960	0.04360	0.09110	0.14260	0.19210	06
P_{PCC}	0.0	0.99710	0.22930	0.04790	0.09170	0.13560	0.18150	07
P_{PCC}	0.0	0.96320	0.23980	0.04950	0.09320	0.14270	0.19230	08
P_{PCC}	0.0	0.99300	0.24630	0.04970	0.09730	0.14900	0.19460	09
P_{PCC}	0.0	0.96920	0.23330	0.04790	0.09170	0.13760	0.18550	10
P_{PCC}	0.0	0.98446	0.23785	0.04696	0.09387	0.14141	0.18914	Mean

CURRICULUM VITAE

Personal Information

Name and surname: Yavuz Selim Hindistan

Academic Background

Boğaziçi University, Faculty of Arts and Sciences, Physics

Istanbul Bilgi University, Master of Business Administration

Foreign Languages: English

Work Experience

Özyeğin University, Instructor, 2019-Present

Manage IT Remote, Founder, 2020-Present

Dohler Gıda, IT Manager, 2016-2020

Doğtaş Kelebek, IT Manager, 2013-2016

Organik Kimya, IT Manager, 2011-2013

Mercedes Benz Financial Services, IT Supervisor, 2004-2011

Smart Card Payment Systems, IT Project Manager, 1999-2004

Banking and Finance, Developer, 1997-1999