ACIBADEM MEHMET ALI AYDINLAR UNIVERSITY
INSTITUTE OF HEALTH SCIENCES

MECHANICAL COMPLICATION PREDICTION IN WELL-ALIGNED ADULT SPINAL DEFORMITY PATIENTS, PRIVACY-PRESERVING ML INFERENCES WITH BLOCKCHAIN APPLICATIONS

BARIŞ BALABAN
PH.D. THESIS

DEPARTMENT OF BIOSTATISTICS AND BIOINFORMATICS

SUPERVISOR
Prof. Osman Uğur SEZERMAN

ISTANBUL-2025

ACIBADEM MEHMET ALI AYDINLAR UNIVERSITY
INSTITUTE OF HEALTH SCIENCES

MECHANICAL COMPLICATION PREDICTION IN WELL-ALIGNED ADULT SPINAL DEFORMITY PATIENTS, PRIVACY-PRESERVING ML INFERENCES WITH BLOCKCHAIN APPLICATIONS

BARIŞ BALABAN

PH.D. THESIS

DEPARTMENT OF BIOSTATISTICS AND BIOINFORMATICS

SUPERVISOR
Prof. Osman Uğur SEZERMAN

ISTANBUL-2025

DECLARATION

I declare that this thesis work is my own work, I had no unethical behavior at any stage from the planning to the writing of the thesis, I obtained all the information in this thesis in accordance with academic and ethical rules, and I cited all the information and comments that were not obtained with this thesis work, and I provided resources in the list of references. I also declare that there was no violation of any patents and copyrights during the study and writing of this thesis.

15 / 04 / 2025

Barış Balaban

PREFACE AND ACKNOWLEDGEMENT

Before introduction, I would like to express my sincere gratitude to my academic advisor, Prof. Osman Uğur Sezerman. His guidance enabled me to create this multidisciplinary work and grow as a scientist. Prof. Çağlar Yılgör's and the European Spine Study Group's significant expertise in adult spinal deformity surgeries, Prof. Erkay Savaş and his fellow student Şeyma Selcan Mağara's guidance on cryptography made my academic journey a joy. I want to thank them, as well as Prof. Emel Timuçin, my industrial advisor Çetin Bağcı, and the Bilmed and TUBİTAK as I am a recipient of a bursary from the Scientific and Technological Research Council of Türkiye (TÜBİTAK) through the 2244 Industry Doctorate Program, in partnership with Bilmed Computer and Software Company. Lastly, I would like to thank my wife, Pelin Ece Balaban, and my family, Deniz, Aycan, and Özge Balaban, for their support.

TABLE OF CONTENTS

DECLARATION ..... **iii**

PREFACE AND ACKNOWLEDGEMENT ..... **iv**

LIST OF FIGURES ..... **xi**

LIST OF TABLES ..... **xiv**

ÖZET **1**

ABSTRACT **1**

1 INTRODUCTION **3**

1.1 Motivation **3**
1.2 Machine Learning Models in Healthcare **4**
1.3 Adult Spinal Deformity Surgeries **4**
1.3.1 GAP score methodology **5**
1.4 The Era of Distributed Systems **6**
1.4.1 Blockchain **6**
1.4.2 Smart contracts **7**
1.4.3 Solutions without blockchain **7**
1.5 Privacy-Preserving Machine Learning **8**
1.6 Our Contribution **9**
1.6.1 Mechanical complication prediction models **9**
1.6.2 Privacy-preserving machine learning use cases in ASD **10**

2 BACKGROUND **11**

2.1 Tree-based Supervised ML Models **11**
2.1.1 Decision trees **11**
2.1.2 Random forest **13**
2.1.3 XGBoost – Extreme gradient boosting **14**
2.2 Privacy-Preserving Machine Learning **15**
2.2.1 Homomorphic encryption **18**
2.2.2 Properties of homomorphic encryption **20**
2.2.3 Lattice-based cryptography **20**
2.3 The ESSG Dataset **21**
2.3.1 Pre-operation features **22**
2.3.2 Operation features **22**

2.3.3 Post-operation features 23

3 MATERIALS AND METHODS..... 24

3.1 Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm 24
3.1.1 Dataset 24
3.1.2 Data manipulation 24
3.1.3 Input feature selection..... 25
3.1.4 Expert-driven input variable selection 25
3.1.5 Optimization of XGBoost parameters via cross-validation..... 26
3.1.6 Evaluation of model performance and outputs 27
3.2Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm 27
3.2.1 Dataset 27
3.2.2 Random forest classification and interpreting random forest models with inTrees 28
3.2.3 Data imputation and manipulation..... 29
3.2.4 Random forest parameter tuning with cross-validation..... 29
3.2.5 Model outputs and performance metrics 30
3.3 Implementation of PPML Inference for Tree-Based Ensembles..... 31
3.3.1 Implementation of privacy-preserving XGBoost inference..... 33
3.3.2 Implementation of privacy-preserving RF inference..... 34
3.4 The Security Analysis of Privacy-Preserving XGBoost Inference..... 34
3.4.1 Dataset 34
3.4.2 Target model cross-validation and performance metrics 34
3.4.3 Security scenarios 36
3.4.4 Substitute models' cross-validation 38
3.4.5 Substitute models' outputs and performance metrics 39
3.5 The Security Analysis Of Privacy-Preserving RF Inference..... 40
3.5.1 Dataset 40
3.5.2 Target models and performance metrics 40
3.5.3 Security scenarios 41
3.5.4 Substitute models' outputs and performance metrics 42

4 RESULTS..... 44

4.1Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm 44
4.1.1 Descriptive statistics of important features..... 44
4.1.2 Cross-validation results for hyperparameter tuning 45
4.1.3 Unguided XGBoost model 46
4.1.4 Expert-driven ensemble for clinical use 48

4.2 Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm 51
4.2.1 Descriptive statistics 51
4.2.2 Cross-validation results 55
4.2.3 RF model with the whole training set 58
4.2.4 InTrees and STEL results 60
4.2.5 Permutation feature importance results 62
4.3 The Security Analysis of Privacy-Preserving XGBoost Inference 65
4.3.1 Synthetic Data Descriptive Statistics 65
4.3.2 Target model CV results 66
4.3.3 Performance metrics of the target model 66
4.3.4 Cross-validation results of the substitute models 68
4.3.5 Performance metrics of substitute models 70
4.3.6 Variable importance and leaked split points analysis 72
4.3.7 File sizes and execution times of the PPML inference 76
4.3.8 RF attack with default parameters 77
4.4 The Security Analysis of Privacy-Preserving RF Inference 79
4.4.1 Performance metrics of the target model 79
4.4.2 Performance metrics of substitute models 80
4.4.3 Finding a safe threshold to protect the privacy of the target model 87
4.4.4 File sizes and execution times of the PPML inference 103

5 DISCUSSION 105

5.1 Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm 105
5.2 Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm 107
5.3 Tree-based Modeling for Interpretability 108
5.4 The Security Analysis of Privacy-Preserving XGBoost Inference 109
5.5 The Security Analysis of Privacy-Preserving RF Inference 111
5.6 Differences in Query Limits of PPML Inferences 113
5.7 Our Contribution to PPML Field 114
5.8 Preferring A Blockchain-less Solution 115

6 CONCLUSION 116

7 REFERENCES 118

APPENDIX 125

APPENDIX 1: Permutation Feature Importance Training Set Results 125

APPENDIX 2: RF PPML Inference Attacks with Random Parameters – Scenario A 126
APPENDIX 3: RF PPML Inference Attacks with Random Parameters – Scenario B 127

9 CURRICULUM VITAE 128

LIST OF ABBREVIATIONS AND SYMBOL

AI	Artificial Intelligence
ASA	American Society of Anesthesiology
ASD	Adult Spinal Deformity
AUC	Area Under Receiver Operating Characteristic Curve
AUPRC	Area Under Precision Recall Curve
BMD	Body Mineral Density
BMI	Body Mass Index
COMI	Core Outcome Measures Index
CV	Cross Validation
CVP	Closest Vector Problem
DT	Decision Tree
EBL	Estimated Blood Loss
ESSG	European Spine Study Group
FHE	Fully Homomorphic Encryption
F-upTime	Follow-up time post-surgery
GAP	Global Alignment and Proportion
GDPR	General Data Protection Regulation
GS	Grid Search
HE	Homomorphic Encryption
HEAAN	HE for Arithmetic of Approximate Numbers
ICU	Intensive Care Unit
IND-CPA	Indistinguishability under Chosen Plaintext Attack
inTrees	Interpretable Trees
KS	Kolmogorov Smirnov
LDI	Lordosis Distribution Index
LIV	Lower Instrumented Vertebra
LIVLoc	Lower Instrumented Vertebra Location
MC	Mechanical Complication

MCS	Mental Component Summary
ML	Machine Learning
MLaaS	Machine Learning as a Service
MPC	Multi-Party Computation
NN	Neural Network
ODI	Oswestry Disability Index
ODI-ws	Pre-operative ODI Walking Score
PCS	Physical Component Summary
PHE	Partially Homomorphic Encryption
PoW	Proof of Work
PPML	Privacy-preserving Machine Learning
RF	Random Forest
RLL	Relative Lumbar Lordosis
RLWE	Ring learning with errors
RPV	Relative Pelvic Version
RSA	Relative Spinopelvic Alignment
SF	Short Form
SHE	Somewhat Homomorphic Encryption
SRS	Scoliosis Research Society
STEL	Simplified Tree Ensemble Learner
TNR	True Negative Rate - Specificity
TPR	True Positive Rate – Sensitivity
UIV	Upper Instrumented Vertebra
VAS	Visual Analog Scale
XGBoost	Extreme Gradient Boosting

LIST OF FIGURES

Figure 1: Feature importance of the unguided model constructed with gain values. 48

Figure 2: Unguided XGBoost model 49

Figure 3: Feature importance of the expert-driven model constructed with gain values. 50

Figure 4: Expert-driven XGBoost model 51

Figure 5: Mean F score vs. RF parameters in the second grid search 57

Figure 6: Mean F score vs. RF parameters in the third grid search. 57

Figure 7: Cumulative error rate of the RF model 58

Figure 8: Built-in feature importance graph of the RF model 60

Figure 9: F-score - permutation feature importance graph 63

Figure 10: Accuracy – permutation feature importance graph 64

Figure 11: Sensitivity – permutation feature importance graph 64

Figure 12: Target model feature importance graphs 68

Figure 13: Performance metrics of the substitute models - scenario A 71

Figure 14: Performance metrics of the substitute models – scenario B 71

Figure 15: Substitute model gain values - scenario A 73

Figure 16: Substitute model gain values - scenario B 74

Figure 17: Heat maps of the target model splits used in substitute models 75

Figure 18: Box plots of RF attack performance metrics – scenario A 78

Figure 19: Box plots of RF attack performance metrics – scenario B 78

Figure 20: Feature importance graph of target model scenario A 80

Figure 21: Feature importance graph of target model scenario B 80

Figure 22: Performance metrics of substitute models with default parameters – scenario A 81

Figure 23: Performance metrics of substitute models with default parameters – scenario B 82

Figure 24: Performance metrics of substitute models with improved default parameters – scenario A 84

Figure 25: Performance metrics of substitute models with improved default parameters – scenario B 85

Figure 26: Performance metrics of the best ten substitute models with random parameters – scenario A 85

Figure 27: Performance metrics of the best ten substitute models with random parameters – scenario B 86

Figure 28: Performance metrics of substitute models with default parameters and 30 queried data – scenario A 88

Figure 29: Performance metrics of substitute models with default parameters and 40 queried data – scenario A 88

Figure 30: Performance metrics of substitute models with default parameters and 50 queried data – scenario A 89

Figure 31: Performance metrics of substitute models with default parameters and 30 queried data – scenario B 90

Figure 32: Performance metrics of substitute models with default parameters and 40 queried data – scenario B 91

Figure 33: Performance metrics of substitute models with default parameters and 50 queried data – scenario B 91

Figure 34: Performance metrics of substitute models with altered default parameters and 30 queried data – scenario A 94

Figure 35: Performance metrics of substitute models with altered default parameters and 40 queried data – scenario A 94

Figure 36: Performance metrics of substitute models with altered default parameters and 50 queried data – scenario A 95

Figure 37: Performance metrics of substitute models with altered default parameters and 30 queried data – scenario B 96

Figure 38: Performance metrics of substitute models with altered default parameters and 40 queried data – scenario B 96

Figure 39: Performance metrics of substitute models with altered default parameters and 50 queried data – scenario B 97

Figure 40: Performance metrics of the best ten substitute models with random parameters and 30 queried data – scenario A 98

Figure 41: Performance metrics of the best ten substitute models with random parameters and 40 queried data – scenario A 99

Figure 42: Performance metrics of the best ten substitute models with random parameters and 50 queried data – scenario A 99

Figure 43: Performance metrics of the best ten substitute models with random parameters and 30 queried data – scenario B 101

Figure 44: Performance metrics of the best ten substitute models with random parameters and 40 queried data – scenario B 102

Figure 45: Performance metrics of the best ten substitute models with random parameters and 50 queried data – scenario B 102

LIST OF TABLES

Table 1: Descriptive statistics of the pre-operation features. 45

Table 2: Descriptive statistics of the operation and post-operation features. 45

Table 3: Cross-validation results of grid searches. 46

Table 4: Confusion matrices of unguided model 46

Table 5: Unguided ensemble leaf nodes 47

Table 6: Confusion matrices of the expert-driven model-building 50

Table 7: Expert-driven ensemble leaf nodes 50

Table 8: Descriptive statistics of pre-operation variables 52

Table 9: Descriptive statistics of operation and post-operation variables 53

Table 10: Cross-validation results 56

Table 11: Performance metrics of the RF model 59

Table 12: Frequent patterns 60

Table 13: The rule set 61

Table 14: Rule set performance metrics 61

Table 15: Clustered variables 62

Table 16: Descriptive Statistics of Important Features in Attack Scenarios 65

Table 17: Cross-validation results of target model 66

Table 18: Confusion matrices 67

Table 19: Target model performance metrics 67

Table 20: Cross-validation results of the substitute models 69

Table 21: Parameter set finalized with cross-validation analysis 69

Table 22: Attack with RF model with default parameters – scenario A 77

Table 23: Attack with RF model with default parameters – scenario B 77

Table 24: Confusion matrices – Scenario A 79

Table 25: Confusion matrices – Scenario B..... 79

Table 26: Target model performance metrics 79

Table 27: Feature importance ranking of substitute model with default parameters – scenario A..... 83

Table 28: Parameters of the best-performing models – scenario A 86

Table 29: Parameters of the best-performing models – scenario B 86

Table 30: Increase in error rates after permuting a cluster..... 125

Table 31: Parameters of the best-performing models with 30-query-attack datasets 126

Table 32: Parameters of the best-performing models with 40-query-attack datasets 126

Table 33: Parameters of the best-performing models with 50-query-attack datasets 126

Table 34: Parameters of the best-performing models with 30-query-attack datasets 127

Table 35: Parameters of the best-performing models with 40-query-attack datasets 127

Table 36: Parameters of the best-performing models with 50-query-attack datasets 127

ABSTRACT

Mechanical Complication Prediction in Well-Aligned Adult Spinal Deformity Patients, Privacy-Preserving ML Inferences with Blockchain Applications

This dissertation presents the development and validation of advanced machine learning models designed to predict mechanical complications in adult spinal deformity patients. Utilizing tree-based supervised machine learning algorithms, this study effectively enhances the predictive accuracy for post-operative complications in two distinct patient groups; well-aligned and moderately disproportioned ASD patients after surgery. The models demonstrated high predictive performance, achieving sensitivity values up to 90% and F-score values around 70%, which are critical metrics for clinical applications in surgical planning and risk management. A considerable part of the research was dedicated to implementing privacy-preserving machine learning techniques to secure the models and the data. By integrating homomorphic encryption, the study ensures that sensitive patient data remains confidential during processing. This approach addresses significant privacy and security concerns prevalent within healthcare data handling. The research findings suggest that these models could substantially aid medical professionals by providing reliable predictions that help in making informed decisions, thus potentially reducing the incidence of mechanical complications in ASD surgeries. Ultimately, the integration of privacy-preserving technological advancements promises to revolutionize the usage such models in surgical planning and postoperative care, significantly impacting patient safety and outcomes in the field of spine surgery.

Keywords: Adult spinal deformity, Mechanical complications, Machine learning, Privacy-preserving machine learning, Homomorphic encryption

ÖZET

İyi Hizalanmış Yetişkin Omurga Deformitesi Hastalarında Mekanik Komplikasyon Tahmini, Gizliliği Koruyan Makine Öğrenme Modelleri ve Blockchain Uygulamaları

Bu tez, yetişkin omurga deformitesi hastalarında mekanik komplikasyonları tahmin etmek için tasarlanmış ağaç-bazlı makine öğrenimi modellerinin geliştirilme ve hasta ve model veri gizliliklerini koruyan tahmin algoritmalarına dönüştürülme süreçleri sunulmuştur. Ağaç tabanlı denetimli makine öğrenimi algoritmalarını kullanarak, bu çalışma ameliyat sonrası komplikasyonlar için tahmin doğruluğunu iki belirgin hasta grubunda; ameliyat sonrası iyi hizalanmış ve orta derecede orantısız hastalada etkili bir şekilde artırmaktadır. Modeller yüksek tahmin performansı göstermiş, %90'e kadar duyarlılık ve %70 civarında F-skor değerlerine ulaşmıştır ki bu değerler, cerrahi planlama ve risk yönetimi için kritik metriklerdir. Homomorfik şifreleme entegrasyonuyla, çalışma işleme sırasında hassas hasta verilerinin gizli kalmasını sağlar. Modellerin gizli kalması ise güvenlik analizleri sonucu bir sorgu limiti atanarak gerçekleştirilmiştir. Bu yaklaşım, sağlık verilerinin işlenmesinde yaygın olan önemli gizlilik ve güvenlik endişelerini ve modelin çalınma risklerini ortadan kaldırmakadır. Araştırma bulguları, bu modellerin güvenilir tahminler sağlayarak tıp profesyonellerine bilinçli kararlar almada yardımcı olabileceğini ve böylece ASD ameliyatlarındaki mekanik komplikasyonların görülme sıklığını azaltabileceğini önermektedir. Modellerin ve hasta verisinin gizliliğinin korunmuş olması bu alanda ilerlemenin önünü açacaktır.

Anahtar Kelimeler: Yetişkin omurga deformitesi, Mekanik komplikasyonlar, Makine öğrenimi, Gizlilik koruyucu makine öğrenimi, Homomorfik şifreleme

1 INTRODUCTION

1.1 Motivation

The ability to collaborate has been crucial in driving progress and innovation throughout history. From the construction of massive architectural wonders to modern-day space exploration missions, the power of multidisciplinary working environments has been central to having an impact both in the industry and in the academy.

Every data scientist knows that it is impossible to build applicable models for real-world scenarios without expertise in a field. As a member of the Acıbadem University Spine Studies Group, I have the privilege to work closely with the European Spine Study Group. ESSG is one of the key stakeholders in this thesis, as it enables us to tackle mechanical complication prediction problems, increase the effectiveness of their surgeries, and decrease the MC rates.

As the world embraces technological advancements, privacy and security concerns are raised. Initial solutions to this problem were solved by encrypting all the records when uploading them on the cloud, and the ciphertexts were useless without the secret key. With the implementations of homomorphic encryption, it was possible to do calculations on the ciphertexts, enabling privacy-preserving machine learning algorithms and inferences. Considering the importance of the privacy and security of individual health records and the financial value of such models in healthcare, privacy-preserving machine learning is a must to deploy such models as a service.

In this thesis, we build a novel XGBoost model for MC prediction in well-aligned ASD patients, a novel random forest model, and a novel rule set for MC prediction in well-aligned and moderately disproportioned ASD patients. Next, to enable clinicians to use our models to secure the privacy of both our models and the query data, we implemented PPML inferences. We tested them to ensure that no information leaks from the models. Finally, we look into the potential of using a blockchain network to

form a public marketplace for model owners and users and consider the benefits and disadvantages of using a blockchain network.

1.2 Machine Learning Models in Healthcare

ML techniques refer to algorithms that learn through experience, which refers to the previous outcomes, namely the input data. Even though the first idea of a machine to become artificially intelligent was in 1950 (1), algorithms have continuously advanced over three decades due to the exponential growth in computing power and the vast amounts of data produced. These technological advancements have prompted companies to embrace data-driven strategies (2). Machine learning techniques have also become extensively utilized across various sectors, including healthcare.

One notable area where machine learning is making significant strides in healthcare is medical imaging analysis. Using deep learning, Esteva et al. enabled machines to classify skin cancer like a dermatologist (3). Shourie et al. trained an algorithm to predict pneumonia on chest X-rays like a radiologist (4). This not only helps to reduce the burden on clinicians but also enhances the speed and precision of diagnosis, ultimately leading to improved patient outcomes.

Moreover, machine learning models are increasingly used to predict patient outcomes and identify individuals at high risk of developing certain diseases (5,6). With technological advancements, tools can analyze large datasets to detect patterns and trends that may not be obvious to humans, allowing for early intervention and personalized treatment plans. With advancements in machine learning, healthcare providers can now leverage this technology to deliver more proactive and preventative care (7).

1.3 Adult Spinal Deformity Surgeries

The North America Spinal Surgery Market report in 2021 predicts that there will be approximately 1.62 million instrumented spinal surgeries annually in the US (8).

According to the American Society of Anesthesiologists (9), back surgeries have a 20% to 40% success rate and success rates vary widely, ranging from 30% to 70% among different back surgery papers (10, 11, 12).

To address MC problems after surgeries, The Scoliosis Research Society-Schwab classification suggests that satisfactory alignment can be achieved with a value of ≤10° for pelvic incidence minus lumbar lordosis, a pelvic tilt of <20°, and a sagittal vertical axis of <4 cm (13). Health-related quality of life scores are used rather than MCs to calculate the threshold values. Thus, an innovative approach is needed to reduce MC rates.

1.3.1 GAP score methodology

GAP score methodology is proposed to be a novel approach, and this method considers the explanatory variables about each other (14). With their method, it is possible to group ASD patients into three categories: Propotioned or well-aligned, moderately disproportioned, and severely disproportioned. Even though their study reported that 5% of the post-operatively well-aligned patients suffered from at least one MC in 2017, the updated data, including new patients, shows that 20% of patients that have a proportioned spinopelvic shape and alignment according to the GAP score will experience an MC. On the contrary, 95% of the severely disproportioned patients, according to the GAP score, suffered from at least one MC.

Yilgor et al. used an ESSG dataset that includes health records of six sites in Ankara, Barcelona, Bordeaux, İstanbul, Madrid, and Zurich between 2010 and 2019 (14). Besides operational data, quality-of-life measures are also included.

Today, the GAP Score methodology is used in the planning phase of adult spine deformity surgeries to predict the MC probability and decide the operational parameters.

1.4 The Era of Distributed Systems

1.4.1 Blockchain

Blockchain enters our lives with the invention of Bitcoin. It is found to propose a new e-commerce solution. Today, banking systems work as mediators. Since they are trusted third parties that do not want failed transactions and do not take a risk, they collect information from both parties, namely the sender and receiver. Their work incurs transaction costs as mediation costs. With their new model, cryptographic proof assures trust between parties, and counterparties can stay pseudonymous. This means that even if you use a new address for each transaction, it is possible to link your history of transactions with your real identity (15).

Before Bitcoin, e-cash was important for giving credit, which allowed anonymous transactions electronically, yet it still needed a trusted third party, particularly banks. The new trust model distributes trust with consensus mechanisms to agree upon previous transactions. However, mutual trust will be achieved among individuals. In the digital world, an individual's identity is a blurry concept since there are no restrictions on your number of addresses. This can be best explained by Sybil attacks, which refer to creating many accounts to have more impact on the system. In a decentralized system, having no authority to give credentials is impossible, so Bitcoin solves this problem with its consensus mechanism proof of work (PoW). With Bitcoin's consensus mechanism PoW, the double-spent problem is solved by agreement with a ledger of previous transactions. The rule of thumb for being sure is to wait 1 hour, approximately an hour, namely six blocks. Each block includes several transactions, and blocks are broadcasted to the Bitcoin network by the miners who solve a puzzle. The proportion of being able to be the first one to solve the puzzle is proportional to each miner's computing power. The only way to solve the puzzle is by brute force, which means trying possible combinations (15,16).

1.4.2 Smart contracts

Electronic coins are a chain of digital signatures. When you want to send bitcoins to someone, you hash the previous transaction and the public key of the next owner. Then, you sign it with your private key. There is a limited number of things that you can do on the Bitcoin network. Thus, the scripting language is basic and has limited functionality. The script is also not Turing-complete. It is because to avoid network crushes such as infinite loops. Not Turing complete scripting language solves the problem of predicting the execution cost. Ethereum is another famous blockchain like Bitcoin, but instead, it is Turing-complete, which supports loops. In Ethereum, transaction cost is calculated proportionally to the time needed to execute the smart contract, so the burden is on users. Ethereum's smart contracts also have a limit that automatically kills infinite loops. Users limit the amount of money they want to spend on gas to run their contracts. A good example of a smart contract is pay-to-script transactions in Bitcoin, a contract to redeem the money. Smart contracts make decentralized autonomous organizations possible since all the money decisions can be made via smart contracts (17). In 2016, a $50M valued Ethereum was stolen because of a decentralized autonomous organization contract flaw. The members of the Ethereum project had a significant stake in the decentralized autonomous organization, so they decided to push back those transactions, which hurt the ledger's immutability (18). Therefore, smart contracts need to be tested well before deployment.

1.4.3 Solutions without blockchain

Blockchain networks are valid candidates to serve as user-centric systems that protect users' private data (19, 20), even though many proposed models for given problems had little to do with blockchain. For example, in supply chains, the main reason for failures is in the data collection step, and blockchain can not solve anything at that step (21).

Blockchain's security and privacy aspects come from cryptography, with encryption schemes and protocols. Next, blockchain's main advantage is that it works

in a fully decentralized setting. However, it is rarely useful in real-world scenarios, as some sort of authority is always needed. Therefore, big technological companies started to provide services on the cloud, like machine learning as a service (MlaaS), where blockchain has no involvement.

Last but not least, transferring data via a public blockchain might be feasible in cases where the data is relatively small (19,22,23) but not in cases with large encrypted health records, like in our case of ASD.

1.5 Privacy-Preserving Machine Learning

ML techniques have revolutionized how industries operate, with finance leveraging predictive analytics for risk assessment and fraud detection. In healthcare and medicine, these techniques are used for diagnosing diseases and personalizing treatment plans based on patient data. Bioinformatics uses machine learning to analyze genomic data for drug discovery and personalized medicine. E-commerce platforms use recommendation systems powered by machine learning to enhance user experience and drive sales. Sensitive personal information is used in all fields.

Smartphone usage has further integrated machine learning into people's daily lives, allowing personalized services based on location, preferences, and behavior. While this offers convenience, it also raises concerns about users' personal information security. Without proper safeguards to protect privacy, there is a risk of sensitive data being leaked or misused by third parties without consent (24).

PPML has emerged as a critical area of focus in the industry and research community, driven primarily by the increasing importance of data privacy regulations such as GDPR compliance. This approach effectively addresses concerns regarding the protection of sensitive information while still leveraging advanced machine-learning techniques to extract valuable insights from data. Organizations can ensure that personal data is kept secure and confidential by incorporating encryption,

anonymization, and other privacy-enhancing technologies into machine learning processes (25).

1.6 Our Contribution

1.6.1 Mechanical complication prediction models

Two tree-based ML algorithms, random forest (26) and XGBoost (27), were used to develop two novel models predicting complications in subgroups of ASD patients.

Using the XGBoost algorithm to predict complications in postoperatively proportioned ASD patients sets a significant precedent in clinical and data engineering studies. Through meticulous data collection and analysis, this collaborative effort has identified key predictive features and optimized model hyperparameters to enhance accuracy and sensitivity. The expert-guided model, fine-tuned for clinical relevance, demonstrates promising results. The model achieved an accuracy of 74%, sensitivity of 80%, specificity of 73%, and a negative predictive value of 95%. The precision stood at 36%, with an F-score of 0.50 (28).

Next, by analyzing data from 295 patients, including both well-aligned and moderately disproportioned ASD patients, we build a second robust model consisting of 500 trees. The subsequent development of an 8-rule set using the same dataset further enhances clinical decision-making by providing specific criteria based on key features identified in the random forest model. What sets this study apart is the impressive performance measures achieved by both the random forest model and the rule set and the potential implications for improving patient care. With accuracy levels surpassing 70% and sensitivity values reaching 90%, these findings offer valuable insights to aid healthcare providers in making informed decisions regarding patient management and preventive treatment strategies.

1.6.2 Privacy-preserving machine learning use cases in ASD

For each machine learning model we build for ASD patients, we implemented PPML inference for the proposed model and discussed the data privacy of model users. Next, we run security test scenarios using synthetic datasets (29) to show not only that the user data remains private due to the encryption scheme but also to prove that the inferences we build do not share enough information with a user so that they can build their copy of our ML models (30).

Organizations like ESSG can open up new doors for monetizing their modeling efforts by ensuring the security of both the ML models and user data. This level of security builds trust with users and attracts potential clients looking to leverage the data for valuable insights. With robust security measures, the value of the models created by organizations like ESSG increases significantly, making them profitable.

Moreover, when patient data are secured, it becomes easier to comply with privacy regulations in healthcare. This not only safeguards the reputation of the model owners but also enables them to tap into new opportunities without fear of data breaches or legal repercussions. Ultimately, prioritizing security in modeling efforts adds a layer of protection and paves the way for innovative and sustainable ways to monetize data assets effectively.

2 BACKGROUND

2.1 Tree-based Supervised ML Models

ML, an AI field, is the leading field showing how we interact with data. Machine learning algorithms are designed to distinguish valuable insights from irrelevant noise by extracting patterns and signals from vast datasets. Through this process, machines can predict outcomes on new data with remarkable accuracy. Instead of unsupervised ML, supervised ML uses labeled datasets to make predictions.

Unlike linear models, which assume a linear relationship between the variables and the dependent variable, tree-based models can capture complex interactions and patterns that may not be easily represented by linear equations. By recursively partitioning the data into subsets based on feature values, tree-based models can effectively model non-linear relationships by creating decision boundaries that separate different classes or groups.

Compared to neural networks, tree-based ML algorithms may outperform small and medium-sized datasets smaller than ten thousand entries (31,32). Furthermore, tree-based ML algorithms remove irrelevant predictors automatically, as such variables would not be selected in branching, resulting in improved accuracy (33).

In the following subsections, we discuss three well-known tree-based supervised ML algorithms: decision trees (34), the building blocks of almost every tree-based ML algorithm, random forest (26), and XGBoost (27), the complexity of which increases as we move down the subsections.

2.1.1 Decision trees

DTs are widely used in many fields due to their interpretable visualizations and ease of usage, which started with their introduction to the scientific world (35). Target and independent variables may be discrete and/or continuous variables, which

minimizes the need for data manipulation prior to modeling. Furthermore, DTs are capable of handling both classification and regression problems. Besides prediction, DTs can be used for variable selection, data manipulation, and handling missing data.

Visually, a DT consists of nodes and branches that form hierarchically. Nodes can be a root node (the initial node), an internal node, or a leaf node without branches. Every node except the leaf nodes has a decision rule of a feature with a cut-off value, and the branches represent whether the statement (decision rule) is true or not. Due to the hierarchical nature, each data point ends in one of the leaf nodes used as predictions.

If a node branches into two subset nodes, they are called parent and child nodes. The DT algorithm calculates the impurity of the parent and child nodes and compares them if the two child nodes produce a better fit. Information gain (36) from information theory and the Gini index (37) is the most common impurity-based criteria. Information gain uses the entropy measure, while the Gini index measures the divergences between target values' probability distributions. If a DT model can classify all of its training data correctly with many nodes, our model will most likely overfit, resulting in poor predictions. In order to eliminate overfitting, stopping conditions for impurity-based criteria have to be applied. Pruning methodologies (35) are also available if the built model is still overfitting, and this can be visible by comparing the training and test results of the model.

The downsides of a DT algorithm are overfitting, poor accuracy, and the significant impact of adding or dropping a feature to the end model, which makes DT's not robust. Tree-based ensemble machine learning models have revolutionized healthcare research by increasing accuracy in predicting patient outcomes and recommending treatment plans. These models, such as random forests and gradient boosting machines, excel in handling complex medical data sets with high dimensionality and non-linear relationships. By leveraging ensemble techniques that combine multiple decision trees, these models can capture intricate patterns within the data that traditional linear models might overlook (38).

2.1.2 Random forest

RF (27) is a versatile and powerful ML algorithm that generates accurate regression and classification models. One of RF's key strengths is its ability to reduce overfitting by creating multiple decision trees through bootstrapping and averaging their predictions. This ensemble approach helps improve the model's accuracy and generalization on unseen data, making it one of the preferred methods for handling complex datasets.

Moreover, Random Forest has the advantage of being able to handle large datasets with high dimensionality without compromising performance. The algorithm is robust against noise, outliers, and missing values in the data, making it a reliable choice for real-world applications where data quality may vary. Additionally, RF provides insights into feature importance by measuring how much each feature contributes to the overall predictive power of the model. This information can be helpful in understanding which variables have the most significant impact on the target variable and guiding feature selection processes for more streamlined models.

By combining bagging (39) with feature randomness, random forests can effectively address overfitting and improve the model's generalization capability. This dual-layered randomization technique enhances the diversity among individual trees in the ensemble, leading to more accurate and stable predictions.

Next, the majority voting mechanism employed by RFs fosters a democratic decision-making process within the ensemble, ensuring that outliers or noisy data points do not sway predictions. This consensus-based approach helps mitigate biases present in individual trees and provides a more reliable prediction for each sample in the dataset.

The parameters to tune in an RF model are entry (the number of features that are candidates for each split), max nodes (the maximum number of terminal nodes in a single tree), node size (the minimum size of terminal nodes), and the ensemble's tree

count. In theory, the accuracy increases as we increase the number of trees in an ensemble. However, in real-world scenarios, after a certain threshold, the model's performance decreases or the gain is negligible (40).

2.1.3 XGBoost – Extreme gradient boosting

XGBoost's (27) success lies in its ability to optimize for parallelization, making it a popular choice among data scientists and Kaggle enthusiasts. By constantly refining weak classifiers based on the residuals of previous models, XGBoost ensures that each new classifier learns from the mistakes of its predecessors. This iterative approach enhances model performance and allows for more accurate predictions by incorporating nuanced patterns identified in residual errors.

A tree ensemble model for a training set of n rows and m features where $D = \{(x_i , y_i)\}$ ($|D| = n, x_i \in R\ m, y_i \in R$) uses K additive functions to make predictions where k represents the k-th boost and F represents the decision or regression tree space. x_i represents i-th instance of the dataset and y-hat denotes the prediction of i-th instance. The best set of K functions is found with the objective function with the loss function and the regularization term.

The loss function, which is the distance between the predicted and the desired outcome, shows how well the model fits the dataset. The regularization term deals with overfitting by penalizing the complexity. For each tree, the complexity term can be expressed as above, where T is the number of leaves and w is the prediction score. The summation of corresponding leaf scores is calculated w.

Objective functions have functions as parameters that cannot be optimized in Euclidean space with traditional methods. Hence, trees are generated in an additive manner. It is not feasible to calculate all the possible tree structures. Hence, the method uses a greedy algorithm to build a tree starting from a single leaf and trying to add branches iteratively. In Chen et al.'s work (27), detailed information about tree boosting and split-finding algorithms is available.

The prediction function for binary classification calculates by summing the corresponding leaf values of each tree for a given data instance. It then converts the total score into predictions between 0 and 1 using the formula $P = 1 / (1 + e^{-\text{Total score}})$.

The hyperparameters to tune in an XGBoost model are learning rate, minimum child weight, maximum depth, gamma, and the number of trees in the ensemble. Hyperparameter tuning goes beyond just optimizing for accuracy; it involves striking a delicate balance between bias and variance. By fine-tuning these parameters, data scientists can prevent overfitting or underfitting, ensuring their model generalizes well to unseen data. This process requires patience and experimentation but can ultimately lead to significant improvements in model performance.

2.2 Privacy-Preserving Machine Learning

ML processes have three subsections to study: data aggregation, training, and inference phases (25). Inference is a post-modeling phase, where the model users input their information to the model and receive model predictions as outputs.

Several PPML techniques exist, namely differential privacy, homomorphic encryption, multiparty computation, and federated learning. In differential privacy, random noise is added to the ciphertext with Laplace transformation, exponential, and randomized response techniques. This technique ensures the privacy of individuals in a dataset, allowing the whole dataset to be analyzed without revealing any sensitive information (41).

Secure multiparty computation (42) is possible with Shamir's secret sharing, where the secret information is stored in k parties. To trace back the secret information, some k parties should collaborate. However, if the secret is revealed in time for use, it becomes a single point of failure again. So secure multiparty computations are needed where they again collaborate without revealing the secret information. Yao introduced Yao's Millionaires problem, where the rich wanted to compare which had more money without revealing their exact fortunes in 1982. The paper uses directed acyclic graphs

where the nodes are operators and edges connect an operation's output to the next one's input. He introduces garbled circuits to solve the problem (43).

The beauty of federated learning lies in its ability to update a central ML model without exposing individual contributor datasets, thus reducing the risk of compromised sensitive information. Thus, the technique is mainly used for training. This decentralized approach not only safeguards data and identity privacy but also opens up possibilities for global collaboration. With federated learning, organizations can benefit from diverse datasets without consolidating them in one location, mitigating the risks associated with large-scale data breaches (44, 45).

Cryptography has close relations to mathematics and computer science; this resulted in cryptographic research to provide proof of security under assumptions of the mathematical models. In practice, security is reduced to a few assumptions. Although one can not be sure that the assumptions are held in real life, it is practical to prove security in that sense. On the other hand, this makes it possible to conceptualize attacks that are not possible with the current computing power but may be possible in the future. To sum up, to prove security, many schemes consider attackers computing power to a specific limit (46, 47).

Meng et al. from Amazon developed a privacy-preserving XGBoost inference that can support binary and multi-class classifications. Their experiment results suggest that such an algorithm can be used for inference tasks on smartphones (48). Magara et al. proposed a PPML inference for genome studies using data encoding and homomorphic encryption techniques for SVM and XGBoost classification (49, 50).

The privacy risks in ML span both the training and inference phases. During training, the main concern is the leakage of sensitive information from the training data, either through the model itself or through intermediate computations. At inference time, both the user's input and the model parameters may be sensitive: the client wishes to keep their query private, while the model owner wants to protect the proprietary structure and parameters of the model. Attacks such as membership

inference, attribute inference, and model extraction have demonstrated the practical risks of exposing ML models or their APIs to adversaries, motivating the need for robust privacy-preserving protocols. (51, 52, 53, 54)

For tree-based models, privacy-preserving training and inference have received particular attention due to the models' popularity in practical applications and their suitability for tabular data. In the training phase, protocols have been developed for both centralized and collaborative settings. These include MPC-based protocols for secure computation of split criteria and impurity measures, as well as differential privacy mechanisms that add noise to statistics or split decisions to prevent leakage of individual data points (53, 54). Recent work has explored the use of GPUs to accelerate MPC-based tree training and inference, significantly improving performance while maintaining strong security guarantees (52).

Homomorphic encryption-based protocols offer low communication overhead but high computational cost, as most cryptographic operations must be performed online and depend on the client's input. Garbling-based protocols leverage fast symmetric-key operations and allow extensive offline precomputation, resulting in efficient online phases but at the cost of high communication and storage requirements, especially for large or deep trees. Hybrid protocols that combine homomorphic encryption for selection or path evaluation with garbling for comparison have been shown to provide Pareto-optimal tradeoffs, particularly for large, sparse trees where the number of decision nodes is much smaller than the number of possible paths. (51)

Recent innovations have further improved the efficiency and practicality of privacy-preserving tree inference. For example, OnePath introduces selective traversal of only the relevant nodes on the prediction path, reducing both computation and leakage compared to protocols that traverse all internal nodes (55). Fully homomorphic encryption (FHE) has also reached a level of maturity where it can support practical, accurate inference on decision trees, random forests, and gradient-boosted trees, with minimal loss in accuracy compared to plaintext inference. These advances make it feasible to deploy privacy-preserving tree-based models in production settings, though

FHE-based approaches may still be too slow for real-time applications with complex models (56).

Collaborative and federated learning of tree-based models introduces additional challenges, as data and computation are distributed across multiple parties. Recent surveys have systematized the literature along axes such as the learning algorithm, collaborative model, protection mechanism, and threat model, and have provided frameworks for analyzing information leakage in distributed settings. Secure aggregation and model fusion protocols, often based on MPC or HE, have been developed for random forests and boosted trees, enabling privacy-preserving ensemble learning without exposing individual data contributions. (54)

Despite significant progress, several open challenges remain. Scalability to very large models and datasets, rigorous quantification and minimization of information leakage, efficient support for malicious adversaries, and generalization to other types of tree-based models and broader ML frameworks are active areas of research (51, 54, 55). The choice of protocol in practice depends on the application requirements, dataset size, model structure, and operational constraints such as network bandwidth and client/server capabilities (51).

In summary, privacy-preserving machine learning for tree-based models is a rapidly advancing field. The state of the art includes a rich set of protocols leveraging homomorphic encryption, garbled circuits, and hybrid approaches, with concrete recommendations available for different application scenarios. Recent work has demonstrated that efficient, accurate, and practical privacy-preserving inference on decision trees and ensembles is now feasible, paving the way for broader adoption in privacy-sensitive domains (51, 52, 53, 54, 55, 56).

2.2.1 Homomorphic encryption

Homomorphic encryption (57) enables computations on encrypted data, namely additions and multiplications. PHE allows only one operation, either multiplication or

addition. RSA (58) and ElGamal (59) schemes are examples. SHE allows both addition and multiplication but limits the amount of multiplication to one. Execution time rises quickly as the multiplication components become larger, limiting such schemes' usage. FHE was realized by Craig Gentry in 2009. The problem with achieving this is that each multiplication adds noise to the ciphertext, and after a point, it is impossible to recover the plaintext. He proposed to use bootstrapping occasionally after some multiplication operations are done. In practice, this means re-encrypting the ciphertext after some multiplication operations. This operation removes the noise, but it adds some noise. Since this operation must be done frequently and it is impossible to remove all the noise, although it is inefficient, progress is made yearly.

Homomorphism is defined as follows. In the context where a mapping f exists from set A to set B, the function f satisfies several conditions: it ensures that f(a) is never equal to f(b) for any distinct elements a and b in A. Furthermore, for every element b in B, there is an element a in A such that f(a)=b. Additionally, the structure of the function f is described as additively homomorphic if f(a+b)=f(a)+f(b) holds true for all pairs a,b in A. Similarly, the structure is recognized as multiplicatively homomorphic if f(a×b)=f(a)×f(b) for all pairs a,b in A (60).

A HE scheme is where a decryption of the result of an operation in the cipher texts gives you the same result in another operation in plain texts. An example of it is RSA algorithm where the encryption is homomorphic over (Z_N , x).

FHE schemes have the same property for two-operation duals. One is distributed over the other like the x operation is distributive over +. In these schemes, the multiplication operation adds significant noise to the cipher texts, which means that after some amount of multiplication, plain texts cannot be recovered. A bootstrapping operation can be used to overcome this, but it is a costly operation. Therefore, homomorphic encryption schemes tend to run slow in today's technology.

Bootstrapping is the operation that uses an encrypted private key to decrypt the output of a multiplication operation. Thus, the noise is gone, but the output is still the encrypted version of our initial plain text.

2.2.2 Properties of homomorphic encryption

By supporting addition and multiplication on encrypted data, HE schemes enable privacy-preserving computations while maintaining the security of the underlying information. The following are the properties of HE.

Firstly, HE schemes provide semantic security or IND-CPA security. The foundation of security in such schemes lies in the inability of an adversary to distinguish between different ciphertexts, thus maintaining the confidentiality and integrity of the encrypted messages. By incorporating randomization into the encryption process, homomorphic encryption prevents patterns from emerging in encrypted data, making it highly challenging for anyone to extract information from the ciphertext alone. Secondly, HE schemes are compact, meaning that the operations on ciphertexts are computed without expanding ciphertext length, ensuring efficient processing. Lastly, HE schemes have efficient decryption times because they are unaffected by the complexity of the functions computed on the ciphertexts (61, 62, 63).

2.2.3 Lattice-based cryptography

In lattice-based cryptography, noise is added to each blinded message so that the noise is hard to separate to extract the secret. This noisy decoding problem is addressed as CVP in a lattice. Generally, such implementations use polynomial rings. Palisade has two popular homomorphic encryption schemes: the BFV scheme is suitable for computations on integers, and the CKKS scheme is suitable for real numbers. Palisade is an open-source project that enables leading homomorphic encryption schemes with implementations of lattice-cryptography building blocks (24, 64, 65).

2.3 The ESSG Dataset

The ESSG dataset used in this thesis was first analyzed in Yilgor et al.'s work, which resulted in a brand-new idea for the ASD field: the GAP score methodology (14). The GAP Score and GAP parameters provide a more detailed evaluation of spinal deformities by considering specific angular measurements at different segments along the spine.

The data collected from six different European sites on adult spinal deformity surgery provides valuable insights into the outcomes of elective spine surgeries over almost a decade. By enrolling patients who met specific criteria, including age, Cobb angle measurements, and other deformity parameters, the study aimed to evaluate the effectiveness of 4-level posterior instrumented fusion procedures in addressing these complex spinal conditions. The requirement for at least 2 years of follow-up also allows for a more comprehensive understanding of patient recovery and long-term success rates (14).

With institutional review board approval obtained at each participating facility, this dataset signifies a collaborative effort among healthcare professionals to gather clinical data that can potentially influence future treatment strategies for adult spinal deformity. As the database continues to grow and incorporate more diverse patient populations, there is an opportunity for even deeper analyses and insights into factors influencing surgical outcomes in this challenging patient demographic.

The MCs defined in the dataset refer to one or more of the following: proximal junctional kyphosis and failure, distal junctional kyphosis and failure, rod breakage, and implant-related complications. The following subsections define the features included in the dataset.

2.3.1 Pre-operation features

The ESSG dataset stores demographic, radiographic, health-status-related, and patient-recorded quality-of-life features as pre-operation features.

Understanding the interplay between demographic features and medical history is crucial in navigating the complex landscape of spine-related issues. Gender, age, height, weight, and BMI are included in the pre-operation features. Moreover, delving into medical history unveils valuable insights regarding previous diagnoses, surgical interventions, and any resultant MCs.

Radiographic measurements also play a crucial role in assessing various spinal conditions and deformities, providing valuable insights for treatment planning and monitoring. Measuring pelvic incidence, T2-T5 kyphosis, T5-T12 kyphosis, and T10-L2 sagittal Cobb angle offers a comprehensive understanding of the spinal alignment and curvature. These parameters are essential in determining the severity of conditions such as scoliosis or kyphosis and guiding surgical interventions toward optimal correction.

Some of the health-status-related features are BMD, menopausal status, and the number of pregnancies, which offer insights into bone health, smoking habits, ASA score, and comorbidities. Using patient-reported outcomes such as VAS scores for back and leg pain, COMI scores, and ODI scores provides valuable insights into the impact of these procedures on different aspects of a patient's life. Additionally, incorporating measures like SF 36 MCS and PCS along with SRS 22 score offers a holistic view of a patient's overall well-being post-surgery.

2.3.2 Operation features

By analyzing demographic factors such as the surgery year and the years of experience of the surgeon and assistant, researchers can gain valuable insights into how these variables impact surgical efficiency. For instance, a study conducted over

multiple years may reveal trends in surgical techniques or technological advancements that influence outcomes. Additionally, considering the experience levels of the surgical team can shed light on how expertise and familiarity with procedures contribute to overall success rates.

Regarding surgical efficiency-related features, factors like duration of surgery, EBL, and urine output provide key indicators of procedural complexity and patient outcomes. Furthermore, assessing parameters such as ICU length of stay and hospitalization time offers valuable information on post-operative recovery processes and healthcare resource utilization.

Some of the important technique-related features are the construct location, the number of stages in surgery, the approach to the surgery, sacroiliac fixation, the number of levels fused, UIV and LIV, implant density, construct rigidity, rod diameter, domino technique, and the number of transverse connectors.

2.3.3 Post-operation features

All radiographic measurements used as pre-operation and operation data were also measured and stored as post-operation features. Furthermore, by using post and pre values of lumbar lordosis and thoracic kyphosis, the delta ,post-to-pre change, values are also stored in ESSG dataset.

3 MATERIALS AND METHODS

3.1 Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm

3.1.1 Dataset

Only well-aligned, proportioned ASD patients of the ESSG dataset used in this study and 160 pre-operation, operation, and post-operation features were included. 25% of the dataset was left out for testing purposes with stratified sampling using MCs and age groups as it is already an established feature regarding complications (14).

3.1.2 Data manipulation

We encoded the GAP parameters, RPV, RLL, LDI, and RSA, into distinct groups to account for their bimodal effects on complications. This encoding resulted in 32 Boolean features, which replaced the original GAP score evaluation features.

All other features were encoded ordinal to ensure informative splits in the resulting decision trees. We allowed missing variables to remain as they were, leveraging XGBoost’s ability to handle missing values.

We created five new variables by multiplying significant features in a clinical context. One example is the product of the construct location's coded value and the number of spinal levels fused. Additional combinations involved the construct's location with BMI, the type of rod material with the construct's location, the construct's location with patient weight, and the duration of follow-up with the surgery year.

To improve the model's efficiency, we rescaled EBL and the duration of the surgery according to the study group site, allowing the model to select the most

appropriate feature rather than omitting the original. To sum up, we engineered 39 variables and excluded eight original.

3.1.3 Input feature selection

We started with 191 features and split the dataset into training and test sets in a 75%-25% ratio, using stratified sampling based on MCs and age groups of twenty years. This ensured that age, a known factor in predicting MCs, was appropriately represented.

Next, we reduced high dimensionality by eliminating features with an absolute correlation value less than 0.1 with our response variable, MCs, using the Pearson correlation coefficient (66). This process removed 118 features. Given the high correlation among patient-reported outcome measures, we selected the preoperative ODI Walking score, which had the highest correlation with MCs, and eliminated 19 other patient-reported outcome measures. After these steps, we retained 54 features for modeling.

3.1.4 Expert-driven input variable selection

Our approach to selecting input variables incorporated guidance from clinical experts to ensure that the model would be clinically applicable and interpretable. Initially, we developed two models using the variables selected: a basic model and an expert-influenced model. The second model refined the feature set by applying specific criteria to maintain clinical relevance and explainability.

1. During this refinement process, we applied several rules to exclude features that were counterproductive or misleading.
2. We removed features that caused data to split in inconsistent directions across different nodes, as these do not allow for clear clinical interpretation.
3. We excluded features that conflicted with established clinical knowledge or effects.

We eliminated features whose threshold values fall within clinically normal ranges, thus providing no actionable insight. We discarded features that were not generalizable across different clinical scenarios or patient populations. Through iterative adjustments, the model-building continued until all remaining features were clinically sensible and resulted in no contradictory or ambiguous splits.

3.1.5 Optimization of XGBoost parameters via cross-validation

We meticulously tuned the hyperparameters of the XGBoost model to optimize performance. Initially, we set the following default values: learning rate at 0.3, 100 trees, minimum child weight of 1, maximum depth of 6, and gamma set to zero. We conducted a comprehensive grid search to identify the best combination of parameters to fit the data effectively.

Our tuning process involved dividing the training set into train and validation subsets using a 2:1 ratio, and employing stratified sampling based on MC outcomes and patient age groups in 20-year increments. We specifically limited the maximum number of trees to 28 in each search iteration to simplify the model and enhance its interpretability. Across multiple rounds of grid search:

1. The first round focused on adjusting the number of trees, maximum depth, and learning rate.
2. The second round refined the minimum child weight and gamma.
3. Subsequent searches further narrowed the parameters to the most effective settings.

We conducted the grid searches iteratively, ensuring each step was closer to the optimal setting. The RStudio (67) environment, using the Caret package (68) for CV and the Dplyr package (69) for data manipulation, facilitated this process.

We evaluated each hyperparameter set by comparing average performance metrics across validation sets. The best-performing set was chosen based on its F-score, considering the imbalanced nature of our dataset.

3.1.6 Evaluation of model performance and outputs

To validate our model's performance, we calculated several key metrics. These included the mean and standard deviation of the AUC, sensitivity, specificity, AUPRC, precision, and F score across the CV analyses.

We first performed descriptive statistical analysis for both our initial unguided model and the subsequent expert-guided model. We then produced confusion matrices for the training and testing sets and derived the corresponding performance metrics. Additionally, we generated feature importance matrices based on the variables' gain scores, which reflect each feature's relative contribution to the model based on the improvement it provides, weighted by the number of observations it influences.

To further interpret the models, we identified the most influential tree leaves by examining those with absolute leaf scores greater than 0.3. The expert-guided model's threshold was slightly adjusted to 0.27 for finer analysis. For each significant leaf, we calculated the MC rate and fold change ratios and conducted Chi-Square tests to compare the predicted group against the rest.

These analytical steps helped us understand the impact and significance of each variable in our models, enhancing their predictive power and clinical applicability.

3.2 Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm

3.2.1 Dataset

Only proportioned and moderately disproportioned ASD patients of the ESSG dataset used in this study and 166 pre-operation, operation, and post-operation features were included. 25% of the dataset was left out for testing purposes with stratified sampling using the response variable MC and post- and pre-operation GAP score subgroups (14).

3.2.2 Random forest classification and interpreting random forest models with inTrees

We construct all our Random Forest models using R (67), employing packages such as Caret (68), Dplyr (69), and RandomForest (70). To address the imbalance in our dataset, we adjust the class weight parameter to a 2:1 ratio, intensifying the penalty for incorrectly predicting patients with MCs.

While random forest models are known for their accuracy, their complex ensemble structure makes them difficult to interpret, especially in clinical settings. To address this, various methods have been developed to simplify interpretation. Friedman et al. used linear models (71), while Liu et al. (72) and Deng et al. employed association rule mining and classification (73). Inspired by these methodologies, the interpretable trees (inTrees) framework was developed to facilitate an easier understanding of tree ensembles (74).

The inTrees framework is designed to extract, prune, and measure rules from any tree ensemble, focusing on simplifying the model's complexity. It includes a component known as the Simplified Tree Ensemble Learner (STEL), which helps analyze frequent variable interactions and create a more manageable rule-based learner. The inTrees pipeline applied in this study involves three key steps, outlined briefly: identifying frequent patterns, selecting and pruning relevant rules, and ordering the rule set by priority for effective classification.

The guided regularized random forests technique (75) is employed within the inTrees framework to enhance feature selection using the importance scores from an existing random forest. This method penalizes information gain from less relevant

variables to refine the model's focus. The rule set is then constructed by prioritizing the most significant rules, first selecting those with minimal error and higher applicability. Each selected rule excludes the corresponding data subset from subsequent rounds, streamlining the rule refinement process.

3.2.3 Data imputation and manipulation

In this study, we faced the challenge of handling missing data, as 53 of the 163 features in our dataset were incomplete. We eliminated thirteen variables because most of the data were missing. We opted to impute other variables' missing values in the training dataset using median values from the respective subsets used for each tree in our random forest model. Unique to the features of menopause and physical activity levels, we introduced a new category instead of median values to address their missing data. We also used median values from the training set for the test set to maintain consistency in data handling across both sets. However, the STEL algorithm does not require complete data, so imputation was bypassed when using this specific model. Despite this, we utilized the fully imputed test set for performance evaluation.

Further, we adjusted the features based on the dual behavior of GAP parameters RPV, RLL, LDI, and RSA. These were substituted with statistical weights from the subgroups identified in Yilgor et al.'s work (14), enhancing our model's applicability to clinical contexts.

3.2.4 Random forest parameter tuning with cross-validation

In this section, we conducted extensive parameter tuning for our Random Forest model through a rigorous process involving nine iterations of three-fold cv. This extensive testing aimed to optimize the model's settings, focusing on key parameters such as the number of features at each split "mtry", maximum nodes in a tree "maxnodes", minimum node size "nodesize", and the total number of trees in the forest.

The accuracy of the RF tends to increase with the number of trees; however, beyond a certain point, additional trees do not significantly improve and can even degrade the model's performance (76). Therefore, we included varying numbers of trees in our tests to determine the optimal quantity for balancing performance and efficiency.

Initially, we divided the training data into training and validation subsets for each CV round, using stratified sampling to maintain representation across MC categories. We began the parameter tuning with a broad search over a predefined grid of parameters, focusing on average F-scores from the validation sets to identify promising parameter combinations.

In subsequent rounds, we refined our search, focusing iteratively on maxnodes, nodesize, mtry, and the number of trees. We incrementally fine-tuned the model by adjusting these parameters based on the F-scores obtained in earlier rounds. The ultimate goal was to select a parameter set that maximized the Random Forest model's F-score, balancing sensitivity and precision efficiently.

The entire process underscored the effectiveness of using the F-score, a harmonic mean of precision and recall, as our performance metric. This metric is handy in scenarios like ours, where the dataset is imbalanced, as it helps to ensure the model accurately classifies patients with MCs**.**

3.2.5 Model outputs and performance metrics

In the CV analysis for each grid search, we calculated the mean and standard deviations for several metrics: F-score, AUC, sensitivity, specificity, AUPRC, and precision for the validation sets. Additionally, scatter plots were produced to visually support the parameter selection process, comparing the mean F-score against parameter levels for the second and third grid searches.

We also discussed the out-of-bag error about the number of trees utilized, presenting these findings in the results section. We evaluated the performance metrics for both the training and testing sets of the final Random Forest model and the rules derived from the simplified tree ensemble learner.

Moreover, the feature importance for the final Random Forest was assessed using the "varImpPlot" function from the "RandomForest" library (70), which calculates the mean decrease in the Gini coefficient across all trees. This addressed data scenarios where traditional feature importance calculations might disproportionately favor high-dimensional categorical variables. To mitigate this and provide a fair comparison of feature importance between the Random Forest and the STEL models, we also calculated permutation feature importance (77) for F-score, accuracy, and sensitivity using the training and test sets. The process involved hierarchically clustering (78) the features based on their correlation coefficients and performing ten thousand permutations of the clustered and individual features. We then analyzed the impact of these permutations on performance metrics, storing these results for detailed evaluation. The decreases in F-score, accuracy, and sensitivity were presented as error rates, highlighting the impact of each feature on the model's predictive accuracy.

3.3 Implementation of PPML Inference for Tree-Based Ensembles

The tree-based ensemble inference is required to make comparison operations for every node within the ensemble. Because of the high computational cost, homomorphic operations to compare presented a significant challenge when implementing tree-based ensemble inference on encrypted data. To address this, we adopted an encoding method for privacy-preserving tree-based ensemble inference, following the approach outlined by Magara (50). This method involved encoding the model based on the split points within the tree-based ensemble and the query data.

Utilizing encodings that reflected the data's values allowed for digit-by-digit multiplication, simplifying the outcome of comparisons to either 1 or 0. Consequently, homomorphic comparisons were effectively reduced to simpler homomorphic

multiplications. For instance, if the encoding of a feature were multiplied by the encoding of a split point, the multiplication would yield zeros if the feature value is less than the split value or a one accompanied with all zeros otherwise.

Comparison operations were required at each node of tree ensembles during XGBoost inference. HE's batching capability allowed for the homomorphic evaluation of multiple nodes at once, enhancing computational efficiency. Consequently, the encodings of all nodes sharing the same index were consolidated into the slots of a ciphertext. This approach significantly streamlined the computational process.

The client began the inference protocol by establishing a connection with the server. Initially, the server transmitted the encoding rules, including the features' split points and the batch size. The batch size, the length of the vector to be encrypted varied based on the number of trees. We recommended using a batch size larger than necessary to obscure the direct relationship between the input array lengths and the number of trees. Consequently, we opted for the maximum batch size allowed in our CKKS setting. Additionally, the model owner introduced random elements into the split point set to disguise the actual split points, a modification that did not impact the model's performance or accuracy due to the batching feature. We set the batch size at 4,096, half of the minimum required ring dimension of 8,192 in our framework, and standardized the number of split points at a fixed number for each feature by incorporating random values.

Moreover, the protocol concealed the order of features within the nodes. To achieve this, the client encoded each feature's values into a separate ciphertext by concatenating their encoded values. On the server side, the features were then rearranged discreetly using masks created by the model owner. After reorganizing the features, the comparison operation for the j-th nodes of all trees was conducted by homomorphically multiplying the ciphertext of the feature values associated with the j-th nodes with the ciphertext of the model for those nodes. This operation effectively merged the encrypted data from the client with the encrypted model parameters to compute the encrypted outcomes. The result was a new ciphertext that contained the

outcome of these comparisons for the j-th nodes across all trees, indicating whether the feature values met the model criteria at each node.

To determine the scores for each tree, we homomorphically multiplied the comparison outcomes along a path starting from the root node and ending in a leaf node, assigning this product as the score of that particular leaf node. If all comparison results along the path were non-zero, the score of this leaf node represented the tree's score, as the scores for other leaf nodes in the same tree were zero. Subsequently, we summed the scores of all the leaf nodes within the same tree to calculate the overall tree score. This method ensured that the specific leaf node chosen remained undisclosed, thereby preventing any leakage of query data to the server.

Finally, we aggregated all individual tree scores using homomorphic operations like rotation and summation. Once decrypted, scores below zero indicated minimal or no risk of MCs. To preserve confidentiality, this number was homomorphically multiplied by a random positive number, and any other numbers were multiplied by zero to conceal the tree score information prior to decryption. This step was known as the final masking phase.

In implementing our method, we employed the open-source homomorphic encryption library, PALISADE, now OpenFHE1. This library supports HE schemes, such as BFV, HEAAN, or CKKS. The BFV scheme is optimized for integer calculations, whereas the CKKS scheme facilitates computations with real numbers and approximate calculations (24,79). Given our need to process real numbers, we utilized the CKKS scheme from the PALISADE library for our tree-based inference algorithms, aligning with the approach detailed in (49,50). The CKKS scheme is secured at the IND-CPA level and relies on the computational hardness of the RLWE problem (80).

3.3.1 Implementation of privacy-preserving XGBoost inference

In the CKKS scheme, we recognized the multiplicative depth and the scaling factor as critical parameters. The multiplicative depth determined how many consecutive multiplication operations could be executed within the algorithm. Additionally, homomorphic operations, especially multiplications, tended to introduce noise into the ciphertext. To curb the buildup of noise or error within the ciphertext, we increased the numerical value by a set number of scaling factor bits in the CKKS schemes. This scaling factor acted as a gauge of precision within the CKKS scheme. Adhering to the guidelines from Magara et al. (49), we set the multiplicative depth and scaling factor bits of our CKKS parameters to 5 and 16, respectively. (50)

3.3.2 Implementation of privacy-preserving RF inference

Following the formulation in Magara's thesis (50), we updated the multiplicative depth and scaling factor bits parameters of CKKS configuration to 6 and 37, respectively. In Magara's work (50) the desired relation between the depth of the trees in the ensemble and the parameters are explained.

3.4 The Security Analysis of Privacy-Preserving XGBoost Inference

3.4.1 Dataset

The dataset used to predict well-aligned ASD patients was used in this analysis. We divided the dataset into target and substitute model sets, assuming that an adversary controls the substitute set. We used stratified sampling to create test sets for both categories. We refer to the training and test datasets of the target model as the training and test sets, respectively, and those of the substitute model as the attacker training and test sets. For the attacker training data, we set the response variable to the predictions made by the target model, while for all other datasets, we used the actual outcomes of MCs.

3.4.2 Target model cross-validation and performance metrics

In our research, we trained target models to predict MCs in adult spinal deformity patients who adhere to the GAP score (14). Based on methodologies from previous studies, we classified all MCs into a single group, establishing a binary classification system where any MC is marked as a positive instance and its absence as a negative instance. Acknowledging the limitations of relying solely on the GAP score to predict these outcomes, we enhanced our predictive model by including additional vital pre-operative, operative, and post-operative features, as cited in related research (80, 81, 82). These features encompass the location of LIV, estimated blood loss, pre-operative ODI walking score, patient age, BMI, a combined measure of levels fused with construct location, and the time elapsed post-surgery.

This thorough method significantly improves our precision in predicting MCs, which is especially critical for the subgroup of ASD patients with proportional alignment. To more accurately detect these complications, we calibrated our model to favor positive instances by adjusting the scale positive weight parameter based on the ratio of negative to positive instances (83). The development of our models involved comprehensive hyperparameter tuning through nine cycles of threefold CV, ensuring the robustness of our target models derived from the training data sets. This approach reflects strategies for creating clinically explainable models, aiming for high sensitivity while ensuring acceptable precision across varied patient scenarios (28).

In our CV analysis, we calculated several metrics' means (and standard deviations), including AUROC, sensitivity, specificity, AUPRC, precision, and F-score. We also prepared confusion matrices with associated metrics for the training, target, and attacker test sets. Although we include AUC and AUPRC values in our CV analysis for comparing models, we adopted a default threshold of 0.5 for predicting MCs in constructing the target models. As a result, the tables from cv and the final models for both target and substitute groups mainly show accuracy results instead of AUC and AUPRC values. Furthermore, we produced graphs of feature importance using gain values to highlight the contribution of each feature to the model’s performance.

3.4.3 Security scenarios

We constructed target models using the training dataset and stored the encoded model on the server. The target model's owner allocated 16 split points to each variable within a feature set of 196 total features, using random values as dummy split points for features with fewer than 16 split points. It is crucial to note that all split points, both real and dummy, were made accessible to the model users. The attacker, aiming to develop substitute models, encoded and encrypted their training data before sending the encrypted query to the model server. After receiving the encrypted model outputs, the attacker decrypted them and used these results as labels to train their substitute models.

The model server was designed to not reveal encrypted user data, and the only information model users could obtain about the model came through the outputs it generated. Thus, we established security scenarios where substitute models were constructed by a designated malicious user-the attacker. We also assumed that this attacker, lacking expert knowledge, utilized all features available for modeling because the specific subset of features used in the target model was not disclosed. The attacker's approach to hyperparameter optimization and the performance metrics of the final substitute models are detailed later. In one scenario, the attacker utilized data from one operation site, and in another scenario, from two out of the six sites involved in this study, indicating that the attacker's training data lacked sufficient variability to replicate the target model using only their dataset.

Although the CKKS scheme provides IND-CPA security, each output disclosed to the client potentially leaks information. Therefore, it was essential to assess how much an adversary, posing as a legitimate client, could learn from querying the model. We conducted experiments to determine the security vulnerabilities of a specific model based on the number of queries a user was allowed. Our dataset included data from six operational sites, each containing a certain number of genuine data points.

In one experiment, scenario A, we isolated the dataset from one arbitrarily chosen site for the attacker's use, while the target model was constructed using data from the remaining sites. This scenario simulated a real-world situation where an attacker, equipped with a limited dataset, queried the private model, learned the outcomes of all patient inferences, and attempted to reconstruct the target model or create a model of similar accuracy. In this setup, 29% of the dataset and 29% of the patients with MCs were allocated to the attacker.

In scenario B, we allocated datasets from two sites for the attacker's use, comprising 39% of the total dataset and 33% of the patients with MCs. This scenario was designed to mimic an attack where a group of users with a relatively larger and more diverse dataset collaborated to appropriate the target model.

During the training phases of the target model, we divided the data into training and test samples at a 75%-25% split for both scenarios. For the training phase of the attack model, we arranged the training and test samples so that the test dataset included at least five patients with MCs, maintaining a 75%-25% split for scenario A and adjusting to a 68.75%-31.25% split for scenario B. We employed stratified sampling based on the presence of MCs in both instances.

Ultimately, the attacker or a group collaborating as attackers could enhance their ability to illicitly acquire the target model by generating synthetic data from their limited authentic dataset. Each scenario offered four distinct methods of attack:

- An attack using just the original dataset designated for attacking.
- Expanding the original dataset with an addition of 50 synthetically created data points for the attack.
- Expanding the original dataset with 150 synthetic data points for the attack.
- Significantly expanding the original dataset by adding 600 synthetic data points for the attack.

To produce these synthetic datasets, we utilized the Data Synthesizer library in Python, which allows for the creation of synthetic data excluding the response variable (29). Although our dataset involved correlated data, our initial attempt to employ the correlated dataset mode failed because the algorithm could not establish Bayesian networks. Following the recommendations by the authors of (29), we switched to the independent mode when faced with excessive computation times or the need to expand the dataset significantly. The descriptive statistics and two-sample discrete KS test (84) p-values were reported in the results section.

We chose a straightforward method for our attack algorithm by using the XGBoost algorithm to develop the attacker models. These models, acquired by attackers, are referred to as substitute models. Considering the necessity for clinical models to be sufficiently simple for incorporation into surgical planning due to the need for interpretability, we deliberately constructed shallow tree-based models. We also demonstrated possible RF attacks with various tree sizes and default parameters to check the model privacy, in a case where attackers only aimed to achieve performance metrics at least as good as the target models..

3.4.4 Substitute models' cross-validation

We conducted two grid searches for each scenario using AUC as the performance metric, as recommended by the XGBoost algorithm documentation (27). We employed nine rounds of threefold cv for parameter optimization. To determine the best-performing parameter set, we chose the simplest ensemble that achieved an AUC value at least equal to the highest AUC multiplied by the validation size minus one, then divided by the validation size. We considered an ensemble simpler if it had fewer trees, a higher gamma value, and larger minimum child weights.

For our CV analysis, we calculated the means and standard deviations of key metrics, including AUROC, sensitivity, specificity, AUPRC, precision, and F-score.

We also simulated the expert-driven phase of building the target model by using only the seven variables identified as important in Balaban et al.'s work (28). Assuming the attacker lacks expert knowledge, they used all 196 features available during the substitute model building. The attacker also attempted to optimize the column subsample parameter to adapt the XGBoost algorithm to handle excessive features with a small dataset.

3.4.5 Substitute models' outputs and performance metrics

To enhance our evaluation of the substitute models' performance, we utilized the selected hyperparameter set to construct 270 distinct models using ten seeds and 27 folds employed in cv. We selected the models with the highest AUC among these 270 for further analysis in the substitute model evaluation. By incorporating the column subsample parameter in our hyperparameter tuning, we chose each node in the substitute models from a subset of all features, adding an element of randomness to the model construction. We used seeds to ensure the reproducibility of the substitute models.

Our model analysis was divided into three distinct sections. In the initial section, we created box plots for the test set's accuracy, sensitivity, precision, and F-score, specifically focusing on those validation sets where the accuracy surpassed 85%. In the second section, we generated box plots to gain values for the features used by the primary models. In the concluding section, we evaluated the split points for each feature in the alternative models. We calculated their ratio compared to the split points of each feature in the primary models. This analysis considered unique split points, meaning repeated splits were considered only once.

Furthermore, we elaborated on the differences in prediction accuracy between the PPML and plaintext versions of the target models, particularly when an attacker used the training data from the substitute models to create these models. We also

documented the average and standard deviations of the execution times needed for preparing a single query, carrying out PPML inference computations for a single query, and decrypting the model output. Furthermore, we included the file sizes for the encoded input and the encrypted output of a single query. Lastly, we detailed the execution times required to calculate individual tree scores and to perform sorting and node comparison operations together.

3.5 The Security Analysis of Privacy-Preserving RF Inference

3.5.1 Dataset

We used the same dataset used in the "Predicting MCs in postoperatively well-aligned and moderately disproportioned ASD patients using RF algorithm" section. We left out the same operation sites for attacker use and used stratified sampling with MC, post, and pre-operation GAP subgroups to get 75% of our dataset for training and the rest for testing purposes.

3.5.2 Target models and performance metrics

We built the target models using all the training sets for each scenario, using the parameters in the RF modeling section. We altered the node size parameters proportionately to decrease the training dataset size. We reported the performance metrics, accuracy, sensitivity, specificity, precision, NPR, and F-score on data sets. We also reported the feature importance graph with a standardized mean decrease in GINI values scaled by standard deviation. With a correlated dataset like ours, it is hard to pinpoint the effect of a single feature because a similar feature might have been chosen there. Due to this, we used the hierarchical clusters used in the RF modeling section to group features and reported the grouped feature importances.

3.5.3 Security scenarios

We developed target models utilizing the training dataset and stored the encoded model on the server. The target model owner assigned 32 split points to each variable within a feature set totaling 153 features, introducing random values as dummy split points for features with fewer than 32 split points. It is important to emphasize that all real and dummy split points were made available to the model users. The attacker, aiming to create substitute models, encoded and encrypted their training data before transmitting the encrypted query to the model server. Upon receipt of the encrypted model outputs, the attacker decrypted these and utilized the results as labels to train their substitute models.

The model server was configured not to disclose encrypted user data, with the only information available to model users coming from the outputs it generated. Consequently, we set up security scenarios in which the attacker built substitute models. In one scenario, the attacker used data from one operation site, and in another scenario, from two out of the six sites involved in this study, suggesting that the attacker's training data did not have enough diversity to replicate the target model using their dataset alone.

Although the CKKS scheme ensures IND-CPA security, each output disclosed to the client potentially leaks information. Therefore, it was crucial to evaluate how much an adversary, acting as a legitimate client, could learn by querying the model. We conducted experiments to assess the security vulnerabilities of a specific model based on the permitted number of queries. Our dataset comprised data from six operational sites, each holding a specific number of genuine data points.

In one experiment, scenario A, we isolated the dataset from one arbitrarily chosen site for the attacker's use while the target model was built using data from the remaining sites. This scenario simulated a real-world situation where an attacker, with a limited dataset, queried the private model, learned the outcomes of all patient inferences, and sought to reconstruct the target model or develop a model of

comparable accuracy. In this setup, 38% of the dataset and 43% of the patients with MCs were designated for the attacker.

In scenario B, we allocated datasets from two sites for the attacker's use, which accounted for 35% of the total dataset and 31% of the patients with MCs. This scenario aimed to replicate an attack where a coalition of users with a more varied dataset collaborated to seize the target model.

There were three parts for each scenario where we built models with increasing numbers of trees. First, we built models with default RF parameters, then, with default RF parameters and the performance metric selected as F-score with a balanced dataset, and finally, 100 models with random parameters. For random parameter attacks, we executed each scenario with 30 predefined random seeds and reported the summaries of performance metrics as box plots. For default parameter attacks, we used 30 random seeds every time. Next, we executed the same procedures for each scenario with fewer data, with 30, 40, and 50 patient records. The original training sizes were 110 for scenario A and 105 for scenario B.

For assessing a safe query threshold, we conducted left-tailed t-tests to check that substitute model F-scores could achieve the target model F-score by hypothesizing that the substitute models' F-score is greater than or equal to the target models' F-score. This analysis was for a random-parameter attack with partial data limited by our proposed query limit for both scenarios A and B.

3.5.4 Substitute models' outputs and performance metrics

Each scenario part discussed in the previous subsection was summarized with the box plots of the performance measures, accuracy, F-score, sensitivity, and precision. Mean and standard deviations were stored. We reported the best-performing ten models out of a hundred for the random parameters scenario. In total, 24 scenario parts were reported in the results section. We also discussed the importance of features and

how they change in substitute models for the scenarios conducted with the whole dataset.

Finally, we explored differences in predictions between the PPML and plaintext versions of the target models when the attacker used the substitute models' training data to develop substitute models. We also documented the mean and standard deviations of the time taken to prepare a single query, execute PPML inference computations for a single query, and decrypt the model output. Additionally, we listed the file sizes for the encoded input and the encrypted model output for a single query. Then, we reported on the execution times for calculating tree scores individually and for performing ordering and node comparison operations collectively.

4 RESULTS

4.1 Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm

Our study included 244 patients, consisting of 200 females and 44 males, who underwent spine surgery with a minimum of four levels of fusion and were followed for at least two years postoperatively. The patients maintained proportionate postoperative sagittal alignments with GAP scores less than three. Among these patients, 42, representing 17% of the total, experienced MCs post-surgery.

We constructed two models to analyze these outcomes: an unguided and expert-guided model. The unguided model incorporated fourteen features to create five decision trees with twenty-three leaves, while the expert-guided model used seven features to build four trees with sixteen leaves.

4.1.1 Descriptive statistics of important features

Pre-operation features found most important included: Age, weight, height, BMI, T2-T5 Kyphosis, T10-L2 sagittal Cobb angle, ODI walking score. Correlation coefficients were 0.24, 0.16, -0.10, 0.25, -0.13, 0.18, and 0.28 respectively, with p-values ranging from 0.001 to less than 0.001. Pre-operation features' descriptive statistics was in **Table 1**.

Operative features determined to be most significant were: Lower instrumented vertebra (LIV) location, combination of the number of levels fused and the construct location, estimated blood loss. Correlation coefficients were -0.33, 0.31, 0.24, 0.15, and 0.16 respectively, with p-values all less than 0.001. The only post-operative feature found crucial was the duration of follow-up in months, with a correlation coefficient of 0.11 (p-value 0.13). Operative and post-operation features' descriptive statistics was in **Table 2**.

Table 1: Descriptive statistics of the pre-operation features.

Features	Age (years)	Weight (kg)	Height (m)	BMI (kg / m^2)	RSA aligned	T2-T5 Kyphosis (°)	T10-L2 Sagittal Cobb (°)	ODI Walking Score
Mean	44.28	63.99	1.62	24.38	0.56	11.47	9.71	1.55
Mean SE	1.39	0.94	0.01	0.31	0.04	0.77	1.33	0.11
Media	42.72	62.00	1.62	24.01	1.00	11.00	8.00	1.00
Std. Dev.	18.85	12.67	0.09	4.23	0.50	10.46	18.00	1.46
Kurtosis	-1.34	1.43	0.18	-0.04	-1.96	5.96	0.12	-0.69
Skewness	0.14	1.06	0.29	0.58	-0.23	1.32	0.43	0.63
Range	64.55	71.00	0.54	22.03	1.00	89.00	97.00	5.00
Min	17.81	39.00	1.38	16.14	0.00	-18.00	-34.00	0.00
Max	82.35	110.00	1.92	38.16	1.00	71.00	63.00	5.00
NA's	0	0	0	0	0	0	0	10

4.1.2 Cross-validation results for hyperparameter tuning

In our study, we conducted comprehensive hyperparameter tuning for our models through extensive grid searches, detailed in Table **3**. XGBoost parameters are listed in parameter set column were the number of trees in the forest, mtry, nodesize, and maxnodes parameters onward in this section. Initially, we explored 70 different parameter combinations across a spectrum of 7 tree counts and 5 learning rates, with depths capped at 3 due to the overfitting observed at greater depths. This first phase set the stage for further refinement. Subsequent phases expanded the scope to include 360 and then 625 combinations, progressively honing in on the optimal settings by adjusting gamma values and minimum child weights among other parameters.

Table 2: Descriptive statistics of the operation and post-operation features.

Features	LIV location	# of levels Fused construct Location	Estimated Blood Loss (ml)	Number of transvers connectors	Surgery time scaled by site	Follow-up duration after surgery (months)
Mean	2.38	39.93	12,812	0.32	0.28	41.60
Mean SE	0.14	1.48	80	0.05	0.01	1.40
Media	3.00	39.00	950	0.00	0.24	36.30
Std. Dev.	1.91	20.01	1,080	0.72	0.19	18.90
Kurtosis	-1.49	0.75	2,71	4.83	2.71	-0.01
Skewness	-0.09	0.95	1,54	2.27	1.36	0.88
Range	6.00	89.00	5,760	4.00	1.00	83.07
Min	0.00	6.00	40	0.00	0.00	18.37
Max	6.00	95.00	5,800	4.00	1.00	101.43
NA's	0	0	0	0	0	0

Table 3: Cross-validation results of grid searches.

Grid Search	Selected Parameter Set	F-score	AUC	Sensitivity	Specificity	AUPRC	Precision
1	0.3 / 4 / 1 / 3 / 0	0.35 (0.08)	0.67 (0.05)	0.50 (0.14)	0.72 (0.06)	0.22 (0.05)	0.27 (0.06)
2	0.3 / 6 / 9 / 3 / 3	0.40 (0.07)	0.71 (0.06)	0.57 (0.13)	0.73 (0.05)	0.30 (0.08)	0.31 (0.06)
3	0.4 / 5 / 10 / 3 / 3	0.42 (0.07)	0.73 (0.06)	0.62 (0.13)	0.72 (0.06)	0.31 (0.05)	0.32 (0.06)
4	0.3 / 4 / 10 / 2 / 3	0.41 (0.06)	0.70 (0.06)	0.63 (0.11)	0.69 (0.08)	0.26 (0.06)	0.30 (0.05)

Ultimately, this meticulous approach allowed us to select a subset of parameters that enhanced the model's performance while simplifying its complexity, achieving a fine balance as evidenced by the slight decrease in the F-score, which we deemed acceptable.

4.1.3 Unguided XGBoost model

In the final phase of our model development, we trained the unguided XGBoost model using the optimal parameter set identified from the third grid search. This model utilized 14 out of the 54 potential features and included 5 decision trees, collectively comprising 23 leaves. Table **4** displayed the confusion matrices for both the training and test datasets, and Table **5** summarized the purer leaf nodes of the ensemble.

Table 4: Confusion matrices of unguided model

Training Set Prediction	**Ref /Real**	**Healthy**	**Mech Comp**	**Test Set Prediction**	**Ref /Real**	**Healthy**	**Mech Comp**
	Healthy	124	2		**Healthy**	39	2
	Mech Comp	27	30		**Mech Comp**	12	8

In the test set, we achieved an accuracy of 77%, sensitivity of 80%, and specificity of 76%, with a negative predictive value of 95%. Although the precision was lower at 40%, the model successfully identified 80% of the patients who would experience MCs. These performance metrics were notably better than those observed during the cv phase, demonstrating the model's robustness despite a high false discovery rate of 60%.

Table 5: Unguided ensemble leaf nodes

Leaf Information	Number of Patients	Mech Comp's (%)	Fold Change	p-value
Influential leaves to predict MCs				
The lowest instrumented vertebra location is pelvis, and pre-operative relative spinopelvic alignment (RSA) is aligned according to GAP score. (Tree 0, node 0 with the exclusion of leaf 1)	37	57%	4.6-fold increase	<0.0001
The multiplication of the number of levels fused and construct location is greater than 41, the ODI walking score is greater than 1, and the height is greater than 1.62m. (Tree 3, leaf 5)	20	65%	4-fold increase	<0.0001
The lowest instrumented vertebra location is the pelvis. (Tree 0, node 0)	82	37%	3.9-fold increase	<0.0001
The estimated blood loss is greater than 1025 ml and the ODI walking score is greater than 1. (Tree 1, leaf 4)	54	43%	3.3-fold increase	<0.0001
Pre-operative T10-L2 sagittal Cobb is less than 1.5°, the operation duration scaled by each site between 0 and 1 is greater than 0.23, and the patient's weight is greater than 61.5 kg. (Tree 2, leaf 4)	57	40%	2.9-fold increase	<0.0001
The height is less than 1.59 meters and the duration of follow-up after surgery is more than 54.4 months. (Tree 4, leaf 2)	21	47%	2.3-fold increase	0.0003
Influential leaves to predict healthy patients				
The multiplication of the number of levels fused and construct location is less than 41 and the pre-operative T2-T5 Kyphosis is greater than 9.5°. (Tree 3, leaf 2)	80	0%	1-fold decrease	<0.0001
The lowest instrumented vertebra location is above the pelvis, the pre-operative T2 -T5 Kyphosis is greater than 6.5° and the BMI of the patient is less 23.4 kg/m^2. (Tree 0, leaf 5)	67	0%	1-fold decrease	<0.0001
The lowest instrumented vertebra location is above the pelvis, the pre-operative T2-T5 Kyphosis is greater than 6.5°. (Tree 0, node 2)	107	2%	0.94-fold decrease	<0.0001
The estimated blood loss is less than 1025 ml, and the age is less than 49.5 years. (Tree 1, leaf 1)	93	2%	0.92-fold decrease	<0.0001
The height is greater than 1.59 and the estimated blood loss in operation is less than 1350 ml. (Tree 4, leaf 3)	108	4%	0.87-fold decrease	<0.0001

Figure **1** displayed the importance matrix of the model with darker splits illustrated the path of the missing values, and Figure 2 illustrated the ensemble.

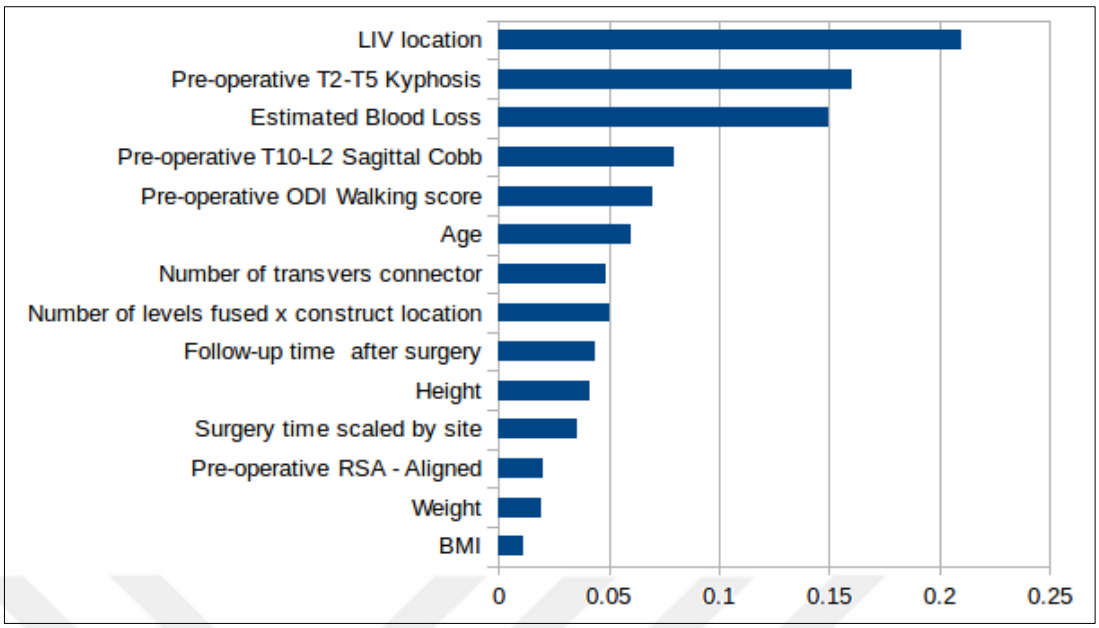

Figure 1: Feature importance of the unguided model constructed with gain values.

The expert-guided model underwent several modifications based on clinical insights, leading to the exclusion of several initial features such as height and the pre-operative T10-L2 sagittal Cobb angle. We also removed the relative spinopelvic alignment due to its binary nature and excluded weight since it was no longer necessary without height for normalization. The model eventually utilized only 7 of the original features deemed most clinically relevant. The model achieved an accuracy of 74% and sensitivity of 80% in the test dataset, demonstrating a slight decrease compared to the unguided model but maintaining a high level of clinical relevance with fewer variables.

4.1.4 Expert-driven ensemble for clinical use

Table 6 displayed the confusion matrices for both the training and test datasets, and Table **7** summarized the purer leaf nodes of the ensemble. Figure 3 displayed the importance matrix of the model, and Figure 4 illustrated the ensemble. Darker splits illustrated the path of the missing values.

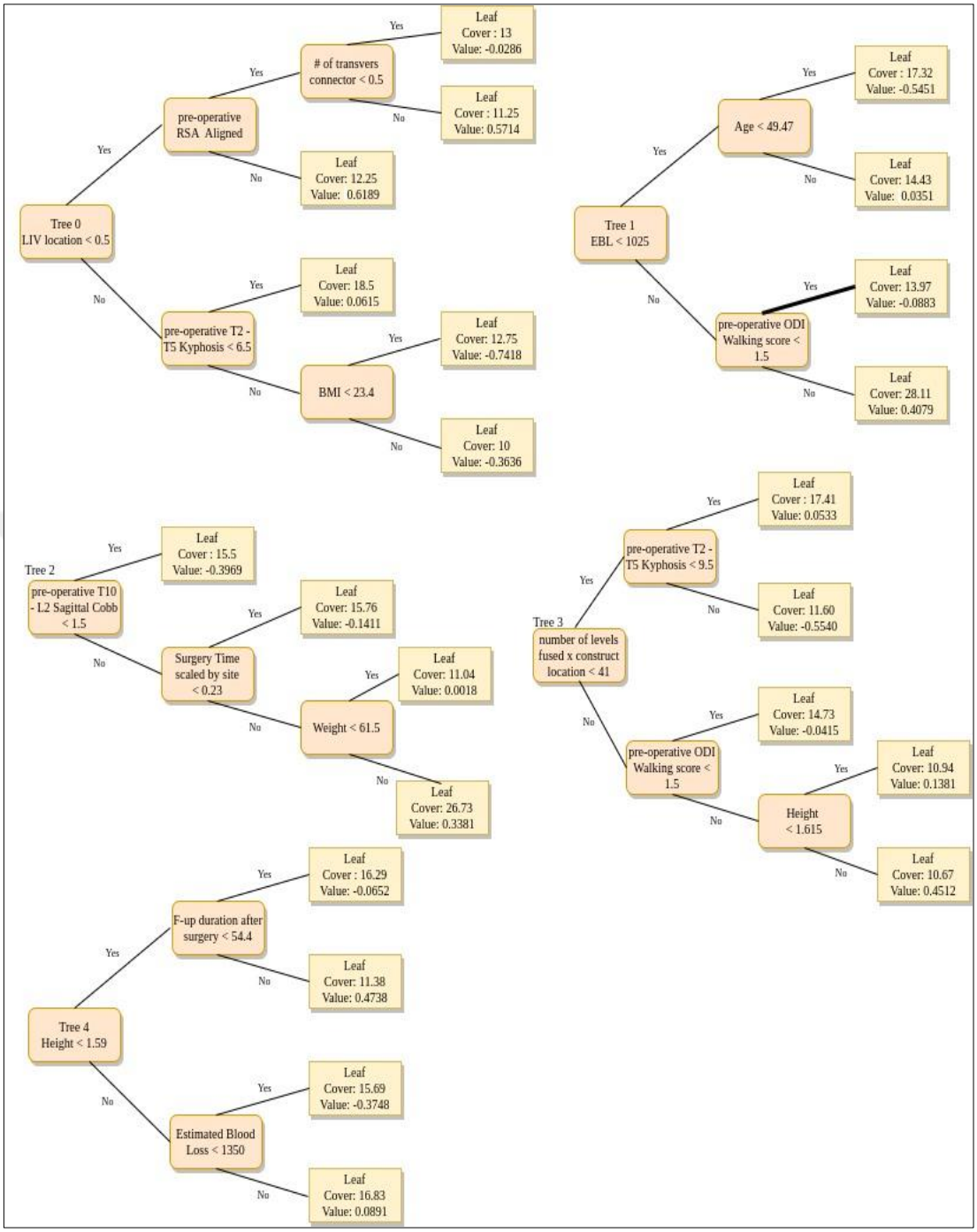

Figure 2: Unguided XGBoost model

Table 6: Confusion matrices of the expert-driven model-building

	Reference	Healthy	Mech Comp		Reference	Healthy	Mech Comp
Training Set Prediction	**Healthy**	109	4	**Test Set Prediction**	**Healthy**	37	2
	Mech Comp	42	28		**Mech Comp**	14	8

Table 7: Expert-driven ensemble leaf nodes

Leaf Information	# of Patients	Mech Comp's	Fold Change	p-value
Influential leaves to predict MCs				
The lowest instrumented vertebra location is the pelvis, or S1- S2 or L5, and the multiplication of the number of levels fused and construct location is greater than or equal to 41. (Tree 1, leaf 2)	76	39%	4.5-fold increase	<0.0001
The estimated blood loss is greater than or equal to 1350 ml and the ODI walking score is greater than 1. (Tree 2, leaf 4)	44	50%	4-fold increase	<0.0001
The lowest instrumented vertebra location is the pelvis and pre-operative ODI Walking score is greater than 2. (Tree 0, leaf 2)	41	46%	3.1-fold increase	<0.0001
Influential leaves to predict healthy patients				
The estimated blood loss is less than 1025 ml, and the age is less than 49.5 years. (Tree 3, leaf 1)	93	2%	0.92-fold decrease	<0.0001
The lowest instrumented vertebra location is not the pelvis, and BMI is less than 24. (Tree 0, leaf 3)	102	3%	0.89-fold decrease	<0.0001
The lowest instrumented vertebra location is not pelvis, or S1- S2, or L5, and the age is less than 35.2 years. (Tree 1, leaf 3)	88	3%	0.86-fold decrease	<0.0001
The estimated blood loss is less than 1350 ml and follow-up duration after surgery is less than 55.8 months. (Tree 2, leaf 1)	126	5%	0.84-fold decrease	<0.0001

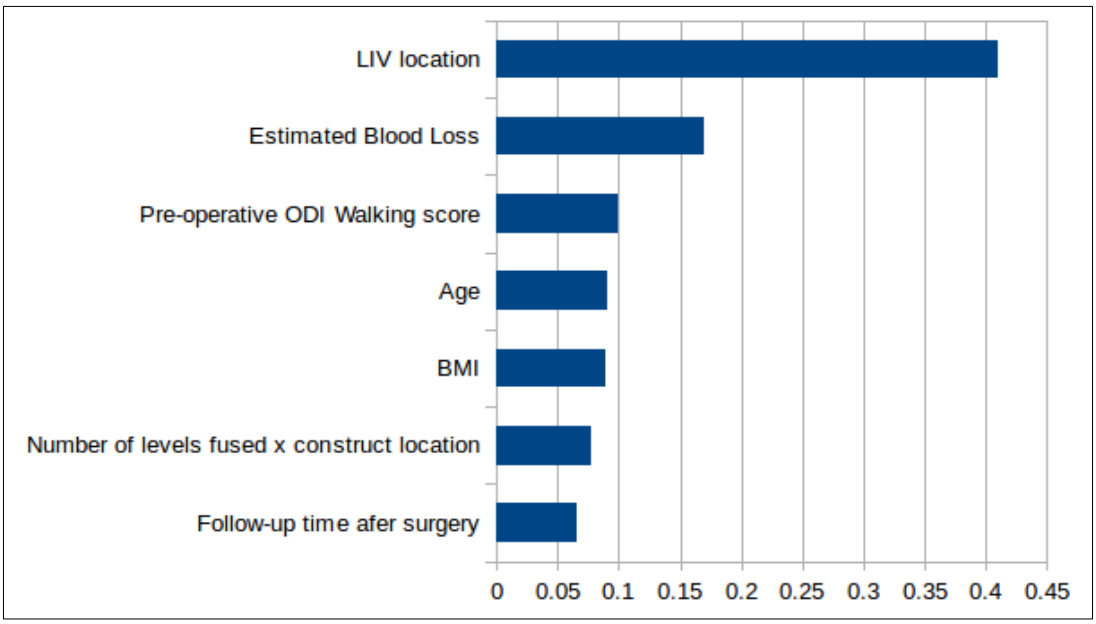

Figure 3: Feature importance of the expert-driven model constructed with gain values.

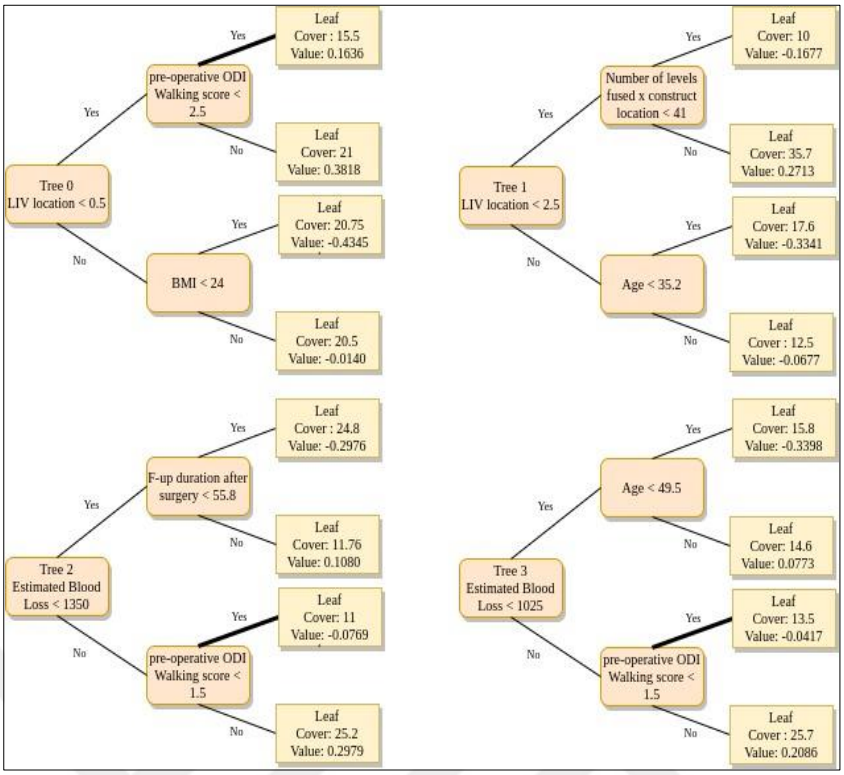

Figure 4: Expert-driven XGBoost model

4.2 Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm

Our training dataset included 295 patients, comprising 237 females and 58 males. During a follow-up period of at least two years, 34% of these patients, or 100 individuals, experienced at least one MC. To address this, we developed a Random Forest model of 500 trees. We also established a rule-based method using a simplified tree ensemble learner to simplify the predictive process. This method generated eight rules, each utilizing between one and three features, to estimate the likelihood of MCs in these patients.

4.2.1 Descriptive statistics

We provided detailed descriptive statistics for the most significant features identified through permutation feature importance analysis and those included in the rule-based decision approach. These statistics were arranged into two tables. Table **8**

displayed the pre-operation variables, illustrating the correlation of each with the presence of MCs among the patient group. If there wasn't any missing values, that line was omitted. Similarly, Table **9** detailed the operative and post-operative features, again correlating these with the occurrence of complications. Both tables included comprehensive statistics such as mean, median, and the range of values for each variable, highlighting their distribution across the patient samples.

Table 8: Descriptive statistics of pre-operation variables

Features	Complication (N=100)	Healthy (N=195)	Total (N=295)	P value
Age (years)				< 0.001
Mean (SD)	58.0 (15.4)	45.2 (18.9)	49.6 (18.8)	
Median (Q1, Q3)	63.4 (51.9, 67.3)	44.1 (26.1, 62.0)	54.1 (31.9, 65.4)	
Min - Max	18.6 - 82.7	17.8 - 82.4	17.8 - 82.7	
BMI (kg/ m^2)				< 0.001
Mean (SD)	26.2 (3.9)	24.1 (4.3)	24.8 (4.3)	
Median (Q1, Q3)	63.4 (51.9, 67.3)	44.1 (26.1, 62.0)	54.1 (31.9, 65.4)	
Min - Max	18.6 - 82.7	17.8 - 82.4	17.8 - 82.7	
BMI (kg/ m^2)				< 0.001
Mean (SD)	26.2 (3.9)	24.1 (4.3)	24.8 (4.3)	
Median (Q1, Q3)	26.0 (23.5, 28.8)	23.6 (20.7, 26.7)	24.1 (21.7, 27.4)	
Min - Max	17.7 - 36.0	16.1 - 38.6	16.1 - 38.6	
GAP Score				< 0.001
Mean (SD)	7.7 (4.1)	5.0 (4.0)	5.9 (4.2)	
Median (Q1, Q3)	9.0 (4.0, 12.0)	4.0 (2.0, 8.0)	5.0 (3.0, 10.0)	
Min - Max	0.0 - 13.0	0.0 - 13.0	0.0 - 13.0	
TL L A Translation to CSVL (°)				0.05
Mean (SD)	22.5 (20.7)	28.3 (22.7)	26.3 (22.2)	
Median (Q1, Q3)	16.3 (5.1, 38.5)	24.9 (7.2, 43.1)	21.8 (6.7, 42.2)	
Min - Max	0.0 - 81.8	0.0 - 82.6	0.0 - 82.6	
Missing	9	13	22	
T10 L2 Kyphosis (°)				< 0.001
Mean (SD)	16.2 (21.6)	8.2 (19.1)	10.9 (20.3)	
Median (Q1, Q3)	14.5 (4.0, 25.0)	6.0 (-6.0, 20.5)	10.0 (-2.0, 23.0)	
Min - Max	-39.0 - 93.0	-34.0 - 70.0	-39.0 - 93.0	
T2 T5 Kyphosis (°)				0.22
Mean (SD)	9.5 (10.8)	10.9 (11.0)	10.4 (11.0)	
Median (Q1, Q3)	7.5 (3.8, 14.2)	10.0 (4.0, 16.0)	9.0 (4.0, 15.5)	
Min - Max	-29.0 - 42.0	-18.0 - 71.0	-29.0 - 71.0	

Table 8: Descriptive statistics of pre-operation variables (Continued)

Features	Complication (N=100)	Healthy (N=195)	Total (N=295)	P value
Major Cobb (°)				0.12
Mean (SD)	35.9 (24.4)	43.3 (23.7)	40.8 (24.1)	
Median (Q1, Q3)	36.5 (15.8, 53.2)	45.0 (24.0, 58.3)	42.0 (21.0, 57.0)	
Min - Max	0.0 - 105.0	0.0 – 96.0	0.0 – 105.0	
Coronal Balance (°)				0.315
Mean (SD)	25.7 (25.4)	22.4 (22.2)	23.5 (23.3)	
Median (Q1, Q3)	18.9 (9.7, 31.6)	16.3 (8.0, 28.5)	17.7 (8.1, 29.4)	
Min - Max	0.0 - 136.5	0.0 - 150.4	0.0 - 150.4	
ODI score				< 0.001
Mean (SD)	46.4 (19.0)	33.9 (20.0)	38.2 (20.5)	
Median (Q1, Q3)	46.0 (32.0, 60.0)	34.0 (16.0, 48.0)	40.0 (23.5, 52.2)	
Min - Max	0.0 - 90.0	0.0 - 80.0	0.0 - 90.0	
Missing	5	14	19	
SRS22 Subtotal				<0.001
Mean (SD)	2.5 (0.6)	3.0 (0.7)	2.8 (0.7)	
Median (Q1, Q3)	2.5 (2.1, 2.9)	2.9 (2.5, 3.4)	2.8 (2.4, 3.3)	
Min - Max	1.4 - 4.2	1.5 - 4.5	1.4 - 4.5	
Missing	4	14	18	
ODI Personal Care				0.01
Mean (SD)	1.3 (1.2)	0.9 (1.1)	1.1 (1.2)	
Median (Q1, Q3)	1.0 (0.0, 2.0)	1.0 (0.0, 2.0)	1.0 (0.0, 2.0)	
Min - Max	0.0 - 4.0	0.0 - 5.0	0.0 - 5.0	
Missing	6	15	21	

Table 9: Descriptive statistics of operation and post-operation variables

Features	Complication (N=100)	Healthy (N=195)	Total (N=295)	p value
Op. LIV location				< 0.001
Mean (SD)	1.0 (1.6)	2.4 (1.8)	1.9 (1.9)	
Median (Q1, Q3)	0.0 (0.0, 2.0)	3.0 (0.0, 4.0)	2.0 (0.0, 4.0)	
Min - Max	0.0 - 5.0	0.0 - 6.0	0.0 - 6.0	
Op. Sacro iliac fixation				< 0.001
Mean (SD)	0.7 (0.4)	0.4 (0.5)	0.5 (0.5)	
Median (Q1, Q3)	1.0 (0.0, 1.0)	0.0 (0.0, 1.0)	0.0 (0.0, 1.0)	
Min - Max	0.0 - 1.0	0.0 - 1.0	0.0 - 1.0	

Table 9: Descriptive statistics of operation and post-operation variables (Continued)

Features	Complication (N=100)	Healthy (N=195)	Total (N=295)	p value
Op. Sacro iliac fix. or S2IA				< 0.001
Mean (SD)	0.8 (0.6)	0.3 (0.5)	0.5 (0.6)	
Median (Q1, Q3)	1.0 (0.0, 1.0)	0.0 (0.0, 1.0)	0.0 (0.0, 1.0)	
Min - Max	0.0 - 2.0	0.0 - 2.0	0.0 - 2.0	
Op. Construct location				< 0.001
Mean (SD)	4.5 (1.0)	3.5 (1.1)	3.8 (1.1)	
Median (Q1, Q3)	5.0 (3.0, 5.0)	3.0 (3.0, 5.0)	3.0 (3.0, 5.0)	
Min - Max	1.0 - 8.0	1.0 - 6.0	1.0 - 8.0	
Op. Delta Lumbar Lordosis (°)				< 0.001
Mean (SD)	15.1 (20.9)	6.5 (17.0)	9.4 (18.8)	
Median (Q1, Q3)	16.0 (0.0, 32.2)	4.0 (-5.0, 16.5)	7.0 (-5.0, 22.0)	
Min - Max	-25.0 - 70.0	-44.0 - 68.0	-44.0 - 70.0	
Op. Delta TK (°)				0.177
Mean (SD)	5.1 (17.3)	2.6 (15.0)	3.4 (15.8)	
Median (Q1, Q3)	5.5 (-6.2, 19.2)	3.0 (-6.0, 13.0)	4.0 (-6.0, 15.0)	
Min - Max	-41.0 - 42.0	-75.0 - 39.0	-75.0 - 42.0	
Op. Number of levels fused				0.003
Mean (SD)	11.8 (3.7)	10.4 (3.6)	10.9 (3.7)	
Median (Q1, Q3)	11.0 (9.0, 15.0)	10.0 (7.0, 13.0)	10.0 (8.0, 13.0)	
Min - Max	4.0 - 20.0	4.0 - 18.0	4.0 - 20.0	
Op. Estimated blood loss (ml)				< 0.001
Mean (SD)	1763.3 (1175.4)	1227.4 (1173.5)	1409.1 (1199.4)	
Median (Q1, Q3)	1475.0 (900.0, 2300.0)	900.0 (500.0, 1625.0)	1000.0 (600.0, 1862.5)	
Min - Max	229.0 - 5380.0	40.0 - 7900.0	40.0 - 7900.0	
Op. Osteotomy Detailed				< 0.001
Mean (SD)	1.1 (1.1)	0.7 (1.0)	0.8 (1.1)	
Median (Q1, Q3)	1.0 (0.0, 2.0)	0.0 (0.0, 1.0)	1.0 (0.0, 1.0)	
Min - Max	0.0 - 5.0	0.0 - 5.0	0.0 - 5.0	
Op. Osteotomy Posterior Column Number				0.098
Mean (SD)	1.1 (1.7)	0.9 (1.9)	1.0 (1.8)	
Median (Q1, Q3)	0.0 (0.0, 2.0)	0.0 (0.0, 1.0)	0.0 (0.0, 1.0)	
Min - Max	0.0 - 7.0	0.0 - 9.0	0.0 - 9.0	
Op. Accessory linked vs satellite unlinked				0.27
Mean (SD)	0.3 (0.6)	0.2 (0.5)	0.2 (0.5)	
Median (Q1, Q3)	0.0 (0.0, 0.0)	0.0 (0.0, 0.0)	0.0 (0.0, 0.0)	
Min - Max	0.0 - 3.0	0.0 - 2.0	0.0 - 3.0	
Op. Hospitalization time (days)				< 0.001
Mean (SD)	14.9 (9.7)	11.1 (6.9)	12.4 (8.2)	
Median (Q1, Q3)	12.0 (9.8, 16.2)	10.0 (7.5, 12.5)	10.0 (8.0, 14.0)	
Min - Max	4.0 - 59.0	4.0 - 54.0	4.0 - 59.0	
Post GAP score				< 0.001
Mean (SD)	3.5 (1.9)	2.0 (1.9)	2.5 (2.0)	
Median (Q1, Q3)	4.0 (2.0, 5.0)	1.0 (0.0, 3.0)	2.0 (1.0, 4.0)	

Table 9: Descriptive statistics of operation and post-operation variables (Continued)

Features	Complication (N=100)	Healthy (N=195)	Total (N=295)	p value
Min - Max	0.0 - 6.0	0.0 - 6.0	0.0 - 6.0	
Post RSA (°)				0.002
Mean (SD)	0.6 (0.7)	0.4 (0.6)	0.4 (0.6)	
Median (Q1, Q3)	0.5 (0.0, 1.0)	0.0 (0.0, 1.0)	0.0 (0.0, 1.0)	
Min - Max	0.0 - 3.0	0.0 - 3.0	0.0 - 3.0	
Post T2 T5 Kyphosis (°)				0.072
Mean (SD)	12.3 (9.0)	13.7 (8.8)	13.2 (8.8)	
Median (Q1, Q3)	12.0 (6.0, 17.0)	13.0 (8.0, 20.0)	13.0 (7.0, 19.0)	
Min - Max	-6.0 - 49.0	-11.0 - 43.0	-11.0 - 49.0	
Post TL L Cobb (°)				0.78
Mean (SD)	15.7 (15.0)	15.9 (13.4)	15.8 (14.0)	
Median (Q1, Q3)	13.0 (2.0, 25.1)	14.3 (5.0, 26.0)	14.0 (3.0, 26.0)	
Min - Max	0.0 - 66.0	0.0 - 63.0	0.0 - 66.0	
Missing	5	21	26	
Follow-up duration (weeks)				< 0.001
Mean (SD)	52.5 (19.0)	41.7 (17.1)	45.3 (18.4)	
Median (Q1, Q3)	55.9 (35.3, 63.7)	35.9 (26.2, 52.9)	40.9 (27.9, 60.4)	
Min - Max	24.1 - 101.4	24.0 - 100.0	24.0 - 101.4	

4.2.2 Cross-validation results

During our analysis, we conducted nine sets of three-fold CV. Detailed results from each grid search were documented in

Table **10**, showcasing the performance metrics associated with each set of parameters found in cv analysis. This table outlines the F-scores, AUC, sensitivity, specificity, AUPRC, and precision metrics, providing a detailed view of the outcomes for various parameter configurations evaluated during the grid search processes.

In the first round of our grid search, we evaluated 42 different parameter combinations involving seven mtry values ranging from 13 to 150 and six different tree sizes ranging from 10 to 1000 trees. We set node size and max nodes to their default values to establish a baseline for future models. The best parameter combinations based on the highest mean F-score were identified and recorded for this initial search.

Table 10: Cross-validation results

Grid Search	Selected Parameter Set	F-score	AUC	Sensitivity	Specificity	AUPRC	Precision
1	10, 25, 1, *null*	0.51 (0.08)	0.72 (0.05)	0.45 (0.09)	0.84 (0.05)	0.53 (0.06)	0.60 (0.09)
2	250, 50, 25, 8	0.62 (0.03)	0.78 (0.04)	0.80 (0.06)	0.61 (0.06)	0.61(0.06)	0.52 (0.03)
3	500, 55, 19, 6	0.63 (0.03)	0.79 (0.04)	0.84 (0.07)	0.59 (0.05)	0.63 (0.06)	0.51 (0.02)

For the second grid search, we significantly expanded our exploration to 840 parameter combinations, still spanning the same mtry range of 13 to 150 but limiting the tree sizes to between 10 and 500. Additionally, we varied the node size across six values from 1 to 45 and max nodes from 4 to 16. The outcomes of this search were visually represented in Figure 5, where baseline parameters for subsequent searches were determined. In the corresponding plot footnotes, details about the subgroup of models are provided. Models yielding lower mean F-scores were discarded before progressing to the next set of parameters, which included maxnodes, nodesize, mtry, and the number of trees.

In the third grid search, we utilized a comprehensive array of 2916 parameter combinations, including nine mtry values ranging from 30 to 70, nine tree sizes between 100 and 500, six node sizes from 16 to 31, and six max nodes between 5 and 10. The resulting parameter optimization process was illustrated in Figure 6. The initial analysis of the scatter plots led to eliminating models using maxnode parameters of 5, 9, and 10. Subsequent scrutiny of the adjusted plot resulted in the selection of node sizes 16 and 19 for further evaluation. Further analysis focused on mtry values, narrowing the choices to 50 and 55.

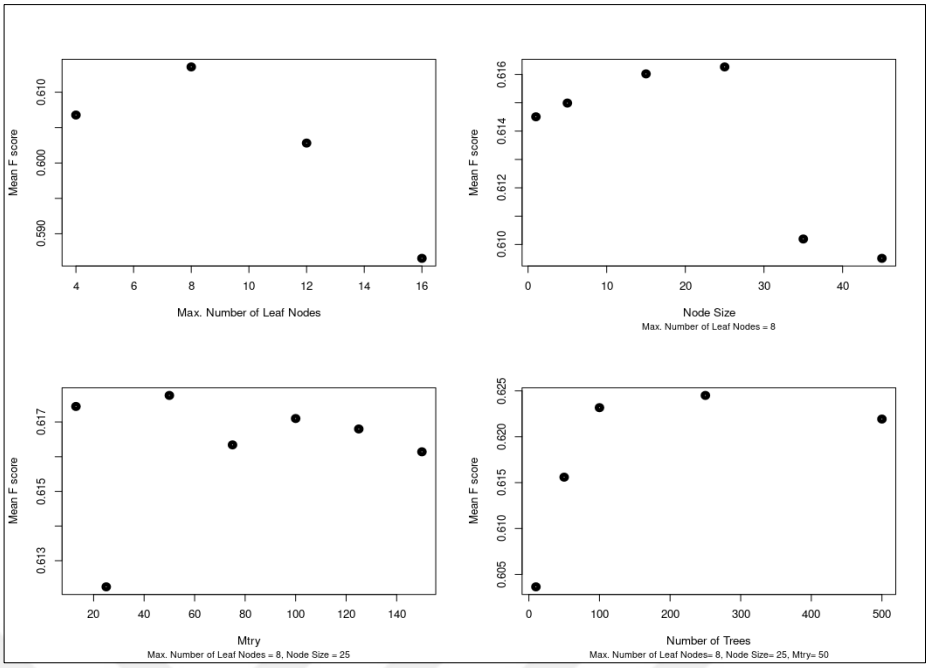

Figure 5: Mean F score vs. RF parameters in the second grid search

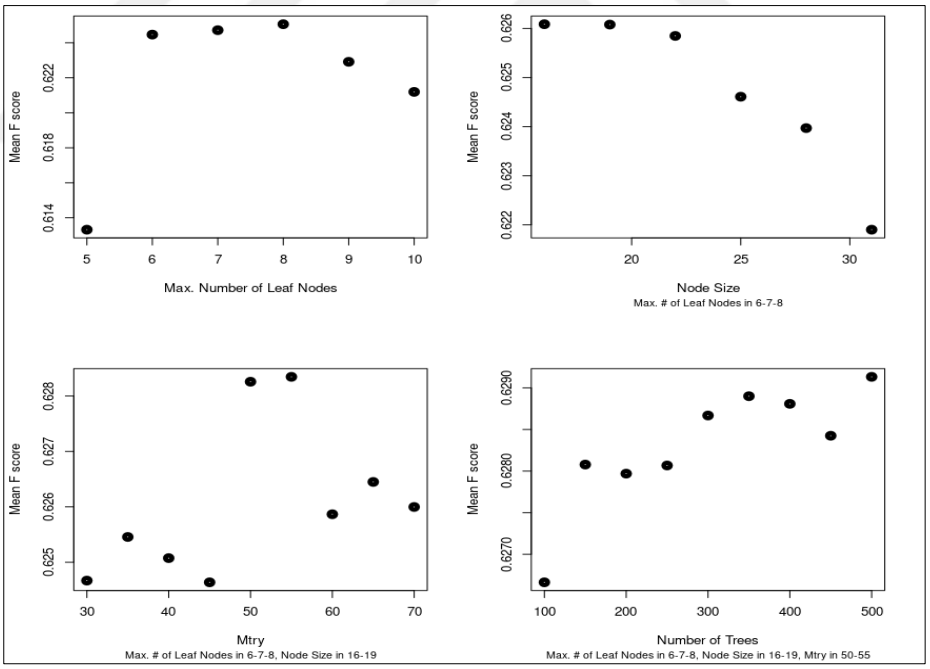

Figure 6: Mean F score vs. RF parameters in the third grid search.

Ultimately, we fixed the number of trees at 500 based on the ongoing positive trend in mean scores despite the potential for further increases with more trees. However, to avoid overcomplicating the model, we chose not to pursue additional tree

counts, as shown by the cumulative error rate graph in Figure 7. Finally, we refined the selection to the parameter sets that yielded the highest F-scores, forming the basis for the final Random Forest model configuration.

4.2.3 RF model with the whole training set

The last version of our random forest model was constructed using the entire training dataset and the optimal parameters identified from the third grid search: 500 trees, an mtry of 55, a nodesize of 19, and maxnodes set to 6. Out of 153 features, the model did not utilize 24, accounting for 16% of the total. The relationship between error rate and tree size is depicted in Figure 7, while the performance metrics for the training, out-of-bag, and test datasets are detailed in Table 11.

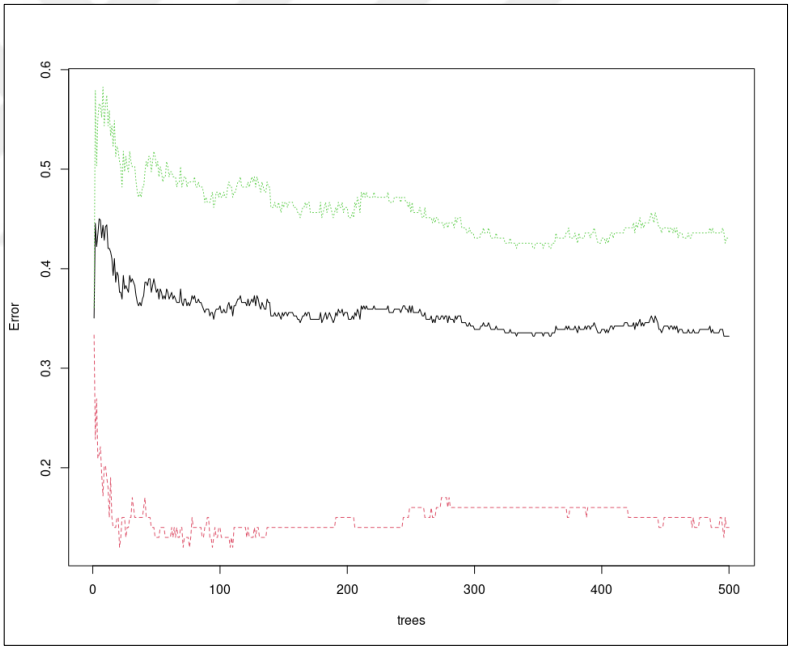

Figure 7: Cumulative error rate of the RF model

In the test dataset, the final random forest model achieved a sensitivity of 91%, which is vital for our goal of providing preventative healthcare to patients at risk of MCs. The model fails to detect only 9% of potential MCs.

Table 11: Performance metrics of the RF model

Data subgroups	Training Set	Out of Bag Set	Test Set
Accuracy	75%	67%	72%
F-score	73%	64%	68%
Sensitivity	99%	86%	91%
Specificity	63%	57%	64%
Precision	58%	51%	55%
NPR	99%	89%	93%
False Positive Rate	37%	43%	36%
False Discovery Rate	42%	49%	45%
False Negative Rate	1%	14%	9%
Matthews Corr. Coef.	59%	41%	51%

Notably, the performance metrics for the training set were substantially higher than those for the out-of-bag and test sets. This discrepancy is due to approximately two-thirds of the training data being used to construct the model. Comparing the results from the out-of-bag and test sets, which were not involved in model training, can make a more meaningful evaluation.

The feature importance matrix is displayed in Figure 8, which shows the mean decrease in Gini values averaged across all trees. The top 30 features are listed, with each feature name prefixed to indicate whether it relates to the pre-operation (pre), operation (op), or post-operation (post) stage. The top five most significant features identified by the random forest model were the post-operation GAP score, the location of the construct, the position of the LIV location, EBL, and the patient's age.

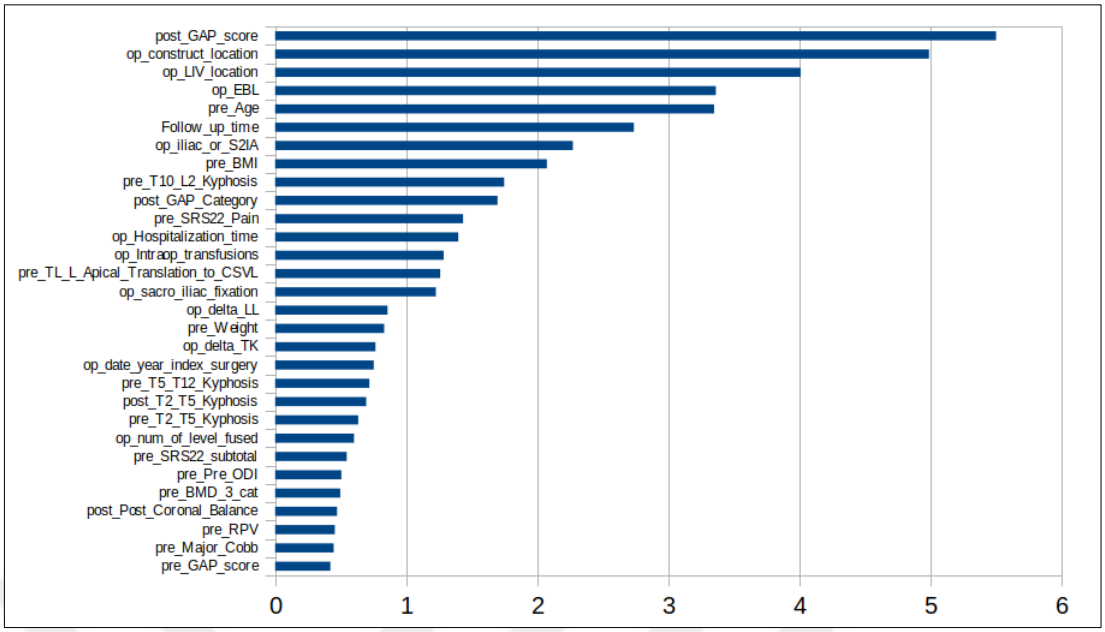

Figure 8: Built-in feature importance graph of the RF model

4.2.4 InTrees and STEL results

We generated frequent patterns using the default parameters of the "getFreqPattern" function, and we displayed these in Table **12**. We arranged the table according to support values, noting that none of the frequent patterns with two or more features had a support exceeding 0.235.

Table 12: Frequent patterns

Condition	Support	Confidence	Prediction
Construct location <= 4.5	0.082	0.626	Healthy
Construct location > 4.5	0.055	0.735	Complication
LIV location <= 0.5	0.047	0.622	Complication
LIV location > 0.5	0.041	0.693	Healthy
Iliac fixation or S2IA <= 0.5	0.039	0.628	Healthy
Post GAP score <= 2.5	0.031	0.614	Healthy
Post GAP score category <= 0.5	0.031	0.662	Healthy
Iliac fixation or S2IA > 0.5	0.024	0.643	Complication

We extracted 2,796 rules from 500 trees and oversampled training data. As only 100 out of 295 training data patients had experienced MCs, we sampled these patients twice to address the imbalance during rule extraction. We pruned the rules using a

"maxDecay" of 0.03 and selected the rule set using the regularized random forests via the "selectRuleRRF" function. The STEL algorithm then used these output rules, filtering for a minimum frequency of 0.1, to create a simplified rule set for predicting MCs. This final rule set is shown in Table **13**. For predicting new instances, we apply each rule from the set sequentially, from top to bottom, until a rule that fits the new instance is found. The performance metrics were in Table **14**.

Table 13: The rule set

Rule (Condition)	Rule Len.	Prediction	Freq. of the rule	Error
Estimated blood loss <= 550 mL & Osteotomy detailed<=1.5 & pre operation ODI Personal Care<=3.5	3	Healthy	0.11	0.00
Hospitalization time > 14.5 days & pre operation TL L Apical Translation to CSVL <= 14.28°	2	Comp.	0.13	0.10
Pre operation T2 T5 Kyphosis > 6.5° & post GAP score <=4.5 & construct location <= 4.5	3	Healthy	0.11	0.06
LIV location <=0.5 & post operation TL L Cobb>19.735°	2	Comp.	0.21	0.10
LIV location <=0.5 & operation accessory linked vs satellite unlinked <= 0.5	2	Comp.	0.14	0.17
Follow-up time <= 57.2 months & operation delta Lumbar Lordosis > -16.5° & operation iliac fixation or S2IA <=0.5	3	Healthy	0.11	0.23
Post Gap Score > 1.5	1	Comp.	0.16	0.19
Else	1	Comp.	0.04	0.32

Table 14: Rule set performance metrics

Data subgroups	Training Set	Test Set
Accuracy	81%	74%
F-score	78%	68%
Sensitivity	95%	81%
Specificity	74%	71%
Precision	66%	58%
Negative Predictive Value	97%	89%
False Positive Rate	26%	29%
False Discovery Rate	34%	42%
False Negative Rate	5%	19%
Matthews Corr. Coef.	66%	49%

4.2.5 Permutation feature importance results

Before identifying key features, we conducted hierarchical clustering to organize highly correlated variables into groups. In the final random forest model, which utilized 129 features, we formed 80 clusters with a maximum of six features in the largest cluster and only one in the smallest. The clusters deemed significant in the feature importance analysis are displayed in Table **15**. Each feature name is prefixed with an encoding to indicate its stage: pre (pre-operation), op (operation), or post (post-operation).

Table 15: Clustered variables

Variable	Clustered Variables
Op LIV location	Op sacro iliac fixation, op construct location, op iliac or S2IA
Post GAP score	Post GAP category, post RPV
BMI	Weight
Age	Body Mass Density features
Pre TL L Apical Translation to CSVL	Pre TL L Cobb, post TL L Cobb, post TL L Apical Translation to CSVL
Pre ODI score	Pre ODI walking and standing scores
Pre GAP score	Pre GAP category, op delta LL, pre RSA, pre RLL, pre RPV
Follow-up time	Year of the surgery
Op EBL	Op Intraop transfusions
Pre SRS22 Subtotal	Pre NRS back, pre ODI Pain Intensity, pre SF36 PCS, pre SRS22 F, pre SRS22 Pain
Op biggest Osteotomy type	Op osteotomy none or post or ant, op osteotomy detailed, op 3column osteotomy location simple, op 3column osteotomy location
Pre Major Cobb	Pre MT Cobb, post Major Cobb, post MT cobb
Op Number of levels fused	Op UIV location anatomical
Op Osteotomy posterior column number	Op Osteotomy posterior column number cat1, op Osteotomy posterior column number cat2

The results of the permutation feature importance, showing increases in error rates for the test set, are displayed in Figure 9, Figure 10, and Figure 11, while the results for the training set are available in (Appendix 1, Table 30). We only highlighted variables in these figures where the permutation resulted in an error rate increase of more than 0.5%.

In Figures 9 to 11, we detailed the mean increases in error rates for performance metrics, as outlined in the methods section. The figures use red to represent measurements from the random forest model and blue for those from the rule set. It's important to note that the feature names depicted may represent clusters of features, not just individual ones (refer to Table **15** for details on clustered variables).

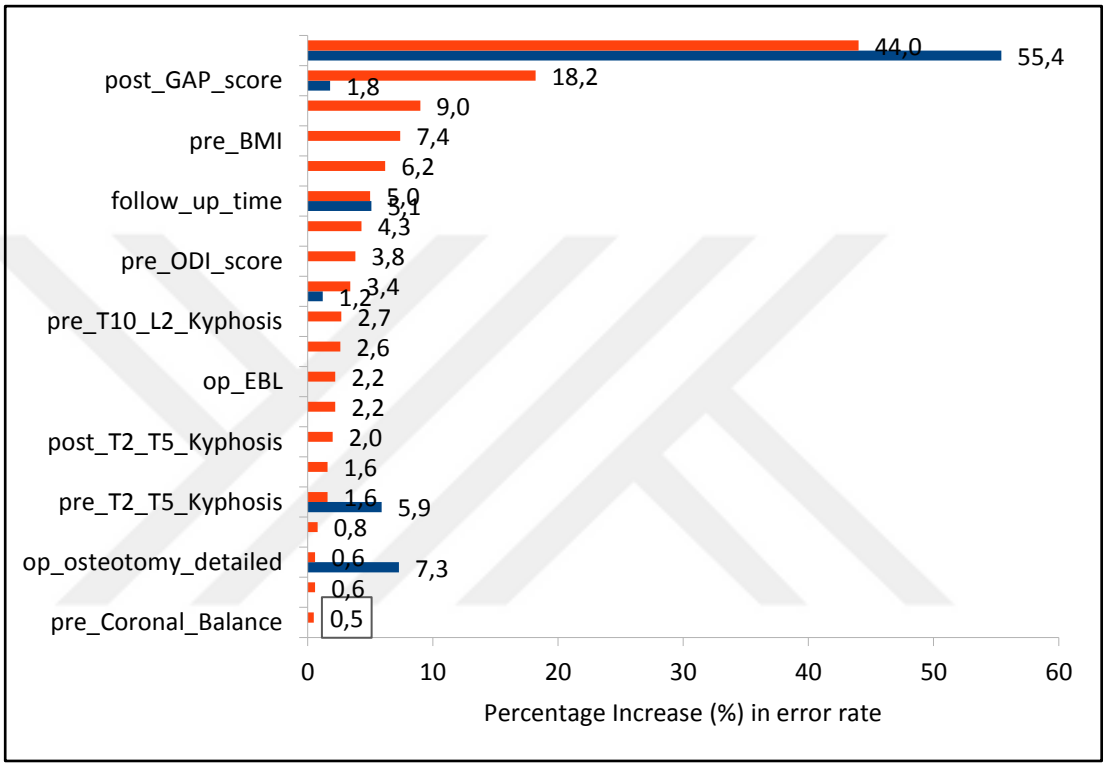

Figure 9: F-score - permutation feature importance graph

In the permutation feature importance analysis, a significant increase, p-value < 0.05, in the error rate of the F-score was observed for 24 clusters in the test set and 25 in the training set for the random forest model. For the rule set, this increase was significant for 5 clusters in the test set and 9 in the training set. Similarly, a significant increase in the error rate of the accuracy metric was noted for 24 clusters in the test set, 25 in the training set for the random forest model, 7 clusters in the test set, and 9 in the training set for the rule set. The error rate of the sensitivity metric showed a significant increase for 19 feature groups in both the test and training sets for the random forest model, for 4 in the test set and 7 in the training set for the rule set.

When comparing the test and training results, the rankings of important features were consistent across all performance metrics except the sensitivity metric for the random forest model. Here, the rankings for the fourth and fifth most important features and the seventh and eighth were different.

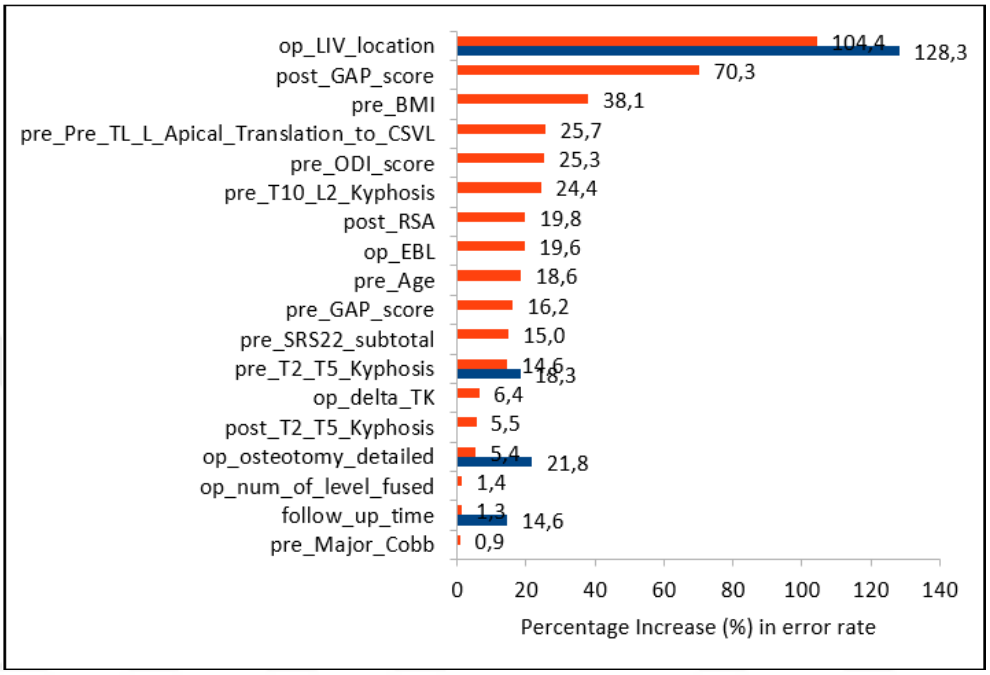

Figure 10: Accuracy – permutation feature importance graph

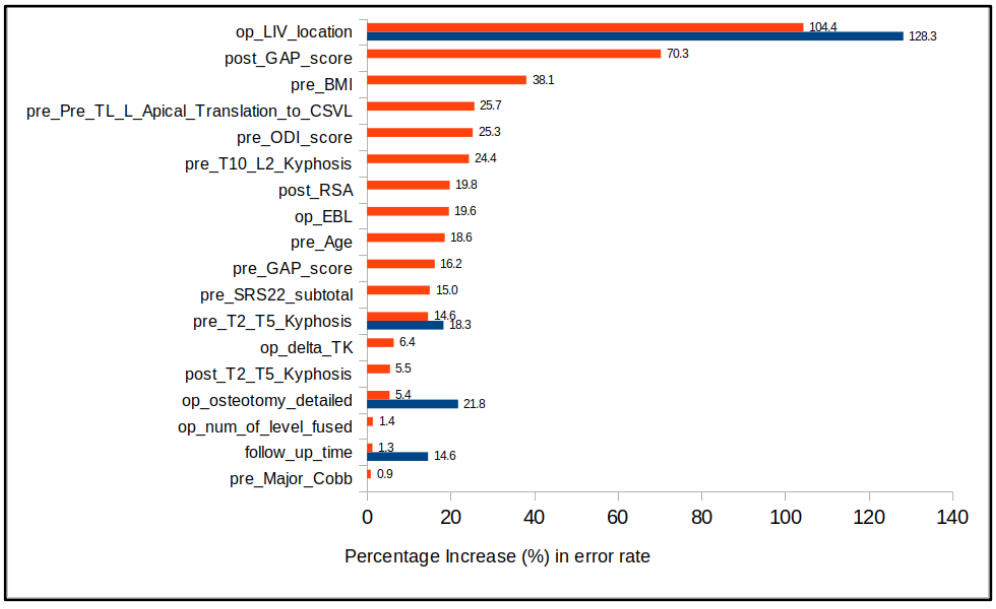

Figure 11: Sensitivity – permutation feature importance graph

4.3 The Security Analysis of Privacy-Preserving XGBoost Inference

4.3.1 Synthetic Data Descriptive Statistics

The mean values, standard deviations (SD), medians, and interquartile ranges of the encoded substitute model training sets were detailed in the first three columns of Table 17 and 18. Descriptive statistics for the associated synthetic datasets for parts 2, 3, and 4 were similarly listed in the subsequent columns of Table **16**, including the results from the two-sample discrete Kolmogorov-Smirnov tests, shown as p-values. Features included in these tables were limited to those found in the target models, ordered by their importance in those models. In scenario A, the original attacker training set contained 53 patient records, while in scenario B, it contained 66. Since all p-values were 0.4 or higher, the null hypotheses that the original and synthetic datasets derived from the same distribution were not dismissed in all scenario parts.

Table 16: Descriptive Statistics of Important Features in Attack Scenarios

	Original attacker training set		Summary of 50 synthetic data			Summary of 150 synthetic data			Summary of 600 synthetic data		
Encoded Features	Mean (SD)	Q1-Q2-Q3	Mean (SD)	Q1-Q2-Q3	KS test p-value	Mean (SD)	Q1-Q2-Q3	KS test p-value	Mean (SD)	Q1-Q2-Q3	KS test p-value
Scenario A - EBL	3.5 (4.0)	0/2/4	3.9 (4.0)	1/2/5.5	0.54	3.6 (4.0)	0/2/4	0.94	3.6 (4.1)	0/2/4	1
F-upTime	5.5 (4.2)	2/4/11	5.9 (3.9)	4/4/10.3	0.40	5.5 (3.9)	2/4/8	0.97	5.5 (4.2)	2/4/8	1
LIVLoc	1.4 (2.0)	0/0/3	1.5 (2.0)	0/0/3	0.93	1.4 (1.9)	0/0/3	0.96	1.4 (2.0)	0/0/3	1
ODI-ws	1.6 (1.5)	0/1/3	1.7 (1.4)	1/1/3	0.61	1.6 (1.5)	0/1/3	0.98	1.6 (1.5)	0/1/3	1
BMI	5.2 (2.8)	3 /5/6	5.5 (2.7)	4/5/6.75	0.71	5.2 (2.7)	3/5/6	0.99	5.2 (2.9)	3/5/6	0.99
Scenario B - EBL	5.5 (4.4)	2/4/9	5.9 (4.2)	2/4/9	0.57	5.7 (4.3)	2/6/8	0.81	5.5 (4.5)	2/4/9	1
Age	5.5 (4.8)	2/4/10.8	6.0 (4.6)	2/4/10.5	0.74	5.7 (4.7)	2/4/9	0.98	5.6 (4.8)	2/4/10	1
FusebyConsLoc	7.3 (3.3)	5/7/9	7.6 (3.2)	6.3/7/9.5	0.49	7.4 (3.3)	5/7/8	0.98	7.3 (3.5)	5/7/9	1
BMI	4.6 (4.9)	1/3/8.8	4.7 (4.8)	1/3/8.8	0.75	4.5 (4.8)	1/3/8	1	4.6 (5.0)	1/3/8.3	1
ODI/ws	1.6 (1.6)	0/1/3	1.9 (1.6)	1 – 2 – 3	0.43	1.7 (1.6)	0/2/3	0.71	1.7 (1.7)	0/1/3	1
F-upTime	6.1 (4.8)	1/5/11	6.8 (4.6)	3/8/11	0.61	6.4 (4.7)	2/8/11	0.84	6.1 (4.8)	1/5/11	1

4.3.2 Target model CV results

We executed a single grid search involving 1500 parameter combinations to select the best parameters, focusing on features such as the LIVLoc, EBL, ODI-ws, age, BMI, FusebyConsLoc, and F-upTime. The search included five variations in the number of trees between 3 and 7, two maximum depths, 2 or 3, five learning rates between 0.1 and 0.5, five gamma values between 1 and 5, and six ranges of minimum child weights between 5 and 10. We set the favorable scale ratio at 5, in alignment with the practices described in Balaban et al.'s work (28). The results of this extensive CV are outlined in Table 17 summarized with mean and standard deviations in parenthesis.

Table 17: Cross-validation results of target model

Scenario	AUC (%)	Sensitivity (%)	Specificity (%)	AUPRC (%)	Precision (%)	F-score (%)
A	72 (9)	53 (19)	79 (7)	0.21 (5)	0.28 (10)	37(9)
B	73 (7)	58 (15)	79 (9)	0.32 (12)	0.39 (16)	45 (13)

For scenario A, we determined the ideal parameters to be six trees, a maximum depth of 2, a learning rate of 0.4, a gamma of 4, and a minimum child weight of 8. In scenario B, the best settings were five trees, a maximum depth of 2, a learning rate of 0.4, a gamma of 1, and a minimum child weight of 6.

4.3.3 Performance metrics of the target model

The target models, trained using the identified parameters and the entire training dataset, resulted in six trees each having 20 leaves for scenario A, and five trees with 20 leaves for scenario B. The confusion matrices for both the training and test datasets were displayed in Table 18, while the related performance metrics were in Table 19.

Table 18: Confusion matrices

		Training set		Test Set - Target		Test Set - Attacker	
Scenario	**Pred Real**	**Healthy**	**Mech. Comp.**	**Healthy**	**Mech. Comp.**	**Healthy**	**Mech. Comp.**
A	**Healthy**	102	2	33	1	9	1
	Mech. Comp.	13	15	4	4	3	4
B	**Healthy**	71	2	23	1	19	1
	Mech. Comp.	21	17	7	6	6	4

Table 19: Target model performance metrics

Scenario	**Set**	**ACC (%)**	**Sensitivity (%)**	**Specificity (%)**	**Precision (%)**	**NPR (%)**	**F-score (%)**
A	**Training**	89	88	89	54	98	67
	Test-Target	88	80	89	50	97	62
	Test-Attacker	76	80	75	57	90	67
B	**Training**	79	89	77	45	97	60
	Test-Target	78	86	77	46	96	60
	Test-Attacker	77	80	76	40	95	53

We presented feature importance graphs in Figure 12, which clearly demonstrated that EBL was the most significant feature in both scenarios. For scenario A, the assembled tree ensemble included five EBL nodes, four of which were unique, along with four F-upTime nodes, two unique, two LIVLoc nodes, one unique, two unique ODI-ws nodes, and a single unique BMI split point. In scenario B, the model featured five EBL nodes, four unique, three age nodes, two unique, one unique Fuse-byConsLoc split point, two unique BMI nodes, two unique ODI-ws nodes, and two unique F-upTime nodes.

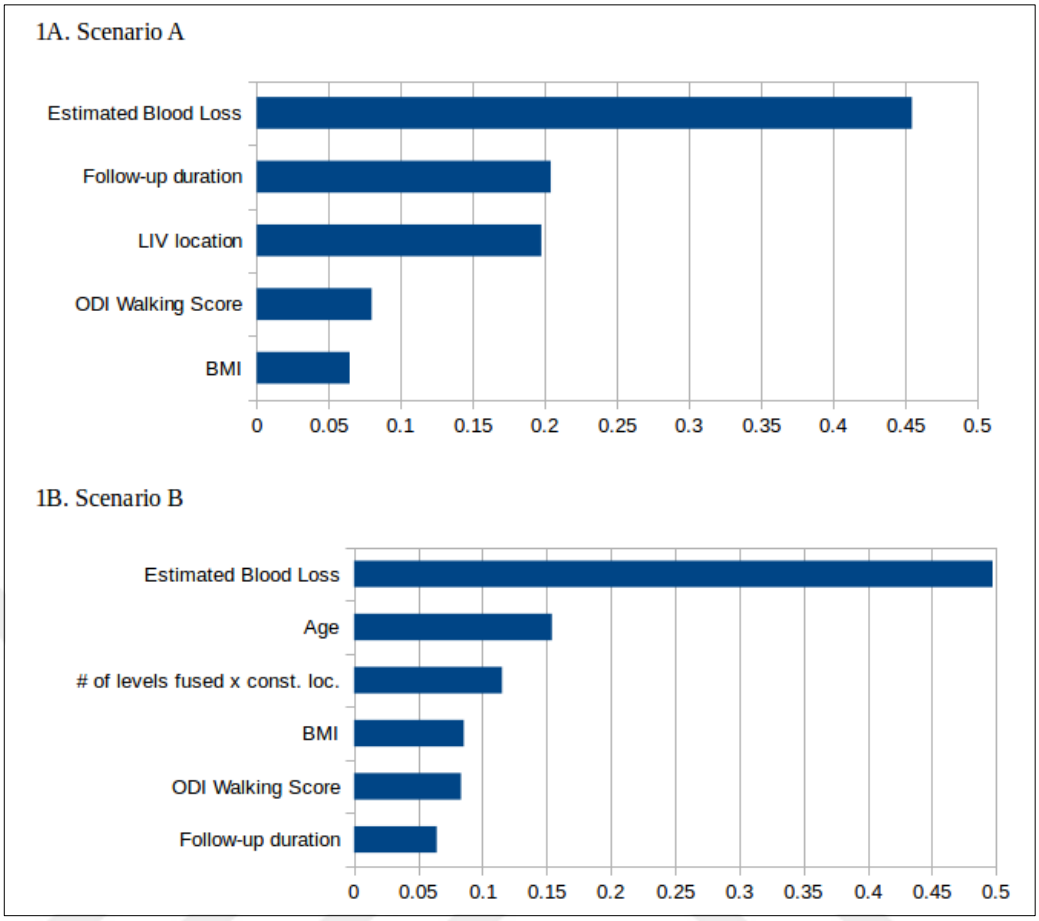

Figure 12: Target model feature importance graphs

In the following three subsections, we explored the substitute models developed by attackers, who employed both real and synthetic datasets to query the target model across both scenarios.

4.3.4 Cross-validation results of the substitute models

We conducted grid searches for each part twice and displayed the performance metrics in Table 20, with the accompanying parameter sets shown in Table 21. Table 20 presents the CV metrics as means and SDs. Both tables categorize each scenario into four sub-scenarios to indicate whether synthetic data was used alongside real attack data. The number appended to the scenario letter, e.g., A1, represents the number of synthetic data points included in the substitute model construction. The

numbers 2, 3, and 4 corresponded to 50, 150, and 600 synthetic data points, respectively, while the number 1 signifies that only real data was utilized.

Table 20: Cross-validation results of the substitute models

Scenario	AUC	TPR	TNR	AUPRC	Precision	F score
A1-1	0.87 (0.09)	0.70 (0.23)	0.86 (0.08)	0.57 (0.15)	0.70 (0.11)	0.68 (0.15)
A1-2	0.87 (0.08)	0.74 (0.21)	0.80 (0.14)	0.52 (0.13)	0.65 (0.17)	0.68 (0.16)
A2-1	0.90 (0.06)	0.86 (0.13)	0.83 (0.08)	0.70 (0.12)	0.78 (0.08)	0.81 (0.08)
A2-2	0.91 (0.05)	0.90 (0.07)	0.83 (0.07)	0.63 (0.11)	0.79 (0.06)	0.84 (0.05)
A3-1	0.97 (0.02)	0.90 (0.09)	0.92 (0.05)	0.81 (0.15)	0.89 (0.06)	0.90 (0.06)
A3-2	0.97 (0.03)	0.90 (0.09)	0.92 (0.05)	0.77 (0.21)	0.90 (0.06)	0.90 (0.06)
A4-1	0.99 (0.00)	0.98 (0.02)	0.97 (0.04)	0.90 (0.03)	0.96 (0.04)	0.97 (0.02)
A4-2	0.99 (0.01)	0.95 (0.06)	0.97 (0.03)	0.76 (0.26)	0.97 (0.03)	0.96(0.03)
B1-1	0.99 (0.01)	0.98 (0.06)	0.92(0.09)	0.29 (0.23)	0.95 (0.06)	0.95 (0.06)
B1-2	0.99 (0.02)	0.99 (0.05)	0.99(0.05)	0.21 (0.20)	0.92 (0.08)	0.95 (0.05)
B2-1	0.94 (0.04)	0.98 (0.04)	0.75(0.15)	0.75 (0.09)	0.90 (0.06)	0.93 (0.04)
B2-2	0.93 (0.04)	0.98 (0.04)	0.76(0.13)	0.45 (0.19)	0.90 (0.05)	0.94 (0.03)
B3-1	0.96 (0.05)	0.95 (0.04)	0.77 (0.13)	0.81 (0.09)	0.91 (0.04)	0.93 (0.03)
B3-2	0.95 (0.03)	0.95 (0.04)	0.74 (0.15)	0.70 (0.14)	0.90 (0.05)	0.92 (0.03)
B4-1	0.99 (0.01)	0.98 (0.01)	0.90 (0.07)	0.97 (0.03)	0.97 (0.02)	0.98 (0.01)
B4-2	0.99 (0.01)	0.98 (0.01)	0.90 (0.07)	0.94 (0.05)	0.97 (0.02)	0.98 (0.01)

During the initial grid search, termed GS1, our evaluation covered 486 combinations of parameters, which comprised choices of having 5, 10, or 15 trees, maximum tree depths of 2 or 3, learning rates set at 0.1, 0.3, or 0.5, column subsample ratios fixed at 0.6, 0.8, or 1, and minimum child weights at 1, 4, or 8. For the subsequent grid search, GS2, we proceeded within a narrowed parameter space derived from GS1's outcomes. We delved into an additional 675 combinations.

Table 21: Parameter set finalized with cross-validation analysis

Scenario	Scale pos. weight	Number of trees	Max. depth	Learning rate	Gamma	Column sub-sample	Min. child weight
A1-1	2.12	5	2	0.5	0	0.6	1
A1-2	2.12	5	2	0.5	0	0.5	3
A2-1	1.45	5	2	0.5	4	1	1
A2-2	1.45	5	2	0.6	3	0.8	2
A3-1	1.36	5	2	0.5	0	1	1
A3-2	1.36	4	2	0.6	2	0.9	1
A4-1	1.11	5	2	0.5	4	1	1
A4-2	1.11	4	2	0.5	2	0.8	4
B1-1	1	5	2	0.1	0	0.6	4
B1-2	1	4	2	0.1	2	0.7	4
B2-1	1	5	2	0.5	0	0.8	1
B2-2	1	3	2	0.5	3	0.7	1
B3-1	1	5	3	0.5	4	1	1
B3-2	1	4	3	0.5	5	0.9	1
B4-1	1	5	3	0.5	0	1	1
B4-2	1	4	3	0.5	0	1	1

In Table 21, we utilized the scale positive weight to adjust the classification threshold, setting it according to the ratio of the number of negative instances to positive instances. We employed a parameter set with an AUC value that was not necessarily the highest to maintain model simplicity. We chose parameter sets whose accuracy could be as low as achieved when one patient is removed from the set that yields the best accuracy. Additionally, we selected parameter sets that result in fewer trees, higher gamma values, and greater minimum child weights, using these criteria in the given order for selection. For example, in scenario B2, the parameter set selected in GS2 was simpler than that in GS1 because it had fewer trees, even though both had acceptably high AUC values.

4.3.5 Performance metrics of substitute models

Figure 13 and Figure 14 displayed the performance metrics for substitute models with a validation accuracy greater than 85%. The counts of substitute models plotted for Scenarios A1 to A4 were 36, 125, 240, and 269, respectively, while Scenarios B1 to B4 included 265, 231, 198, and 27 substitute models, respectively. In these figures, the stars indicated the performance metrics of the target models on the attacker test data, showcasing all eight scenarios with varying numbers of substitute models. In scenario B4, we selected a column subsample ratio parameter of one, as determined by the CV results. Thus, the same ensemble was constructed for each seed, yielding 27 models instead of the anticipated 270. The most effective attack in scenario A involved querying the target model using only real attacking data. The mean values [with standard deviations] for accuracy, sensitivity, precision, and F-score were 0.49 [0.12], 0.62 [0.25], 0.32 [0.099, and 0.41 [0.10], respectively. When synthetic data were incorporated in scenario A, both precision and accuracy decreased.

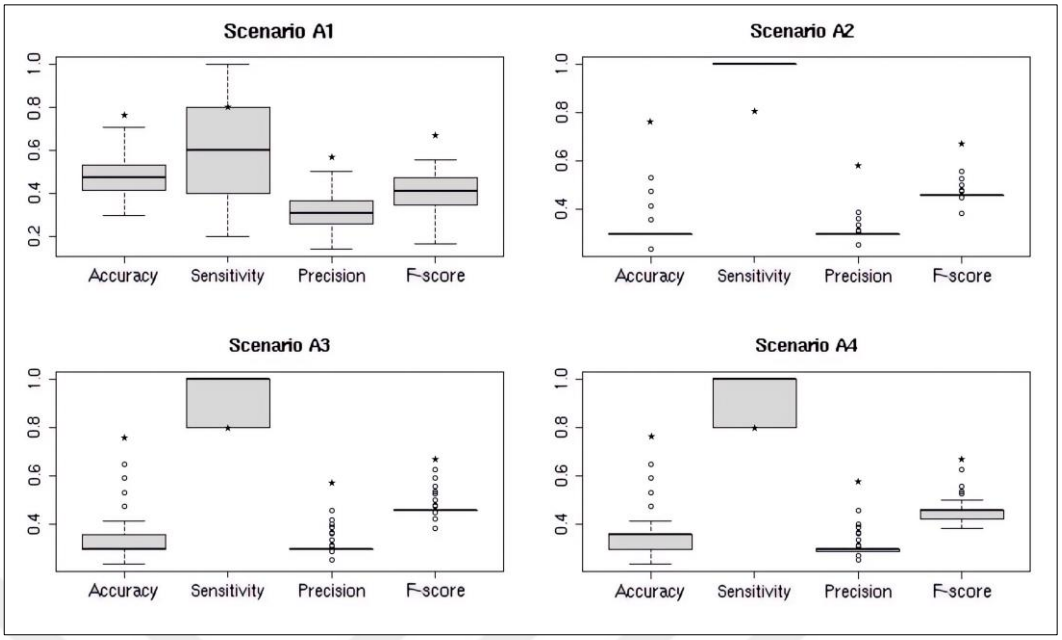

Figure 13: Performance metrics of the substitute models - scenario A

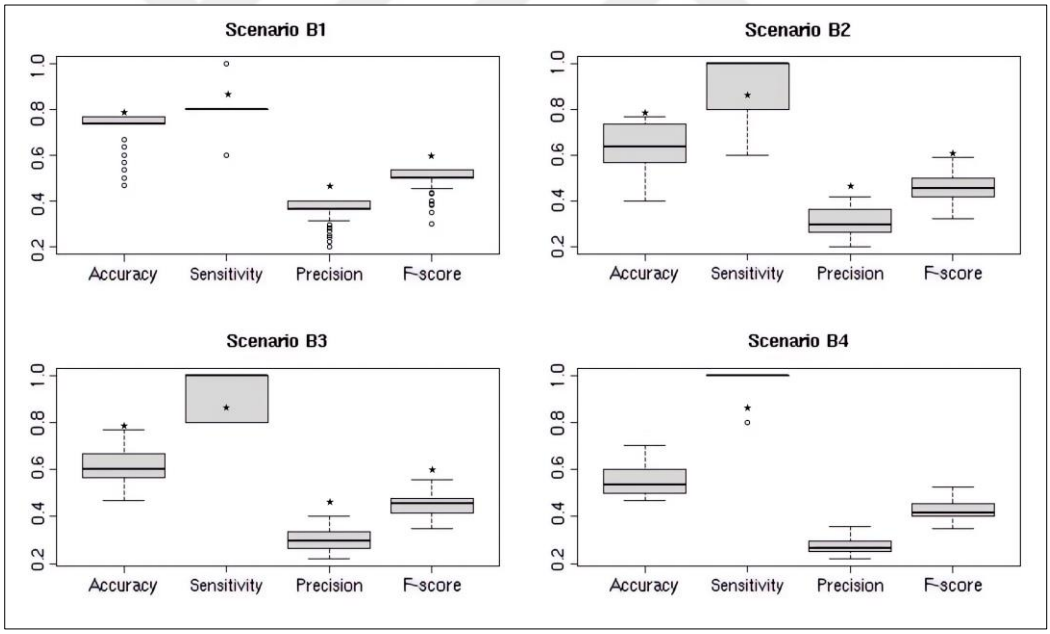

Figure 14: Performance metrics of the substitute models – scenario B

The mean values [with standard deviations] for accuracy were 0.31 [0.06], 0.36 [0.09], and 0.37 [0.11] for scenarios A2-A4, respectively. In contrast, the most effective attack in scenario B, B1, involved the attacker querying the model with real data points. The means [standard deviations] for accuracy, sensitivity, precision, and

F-score in this scenario were 0.73 [0.06], 0.80 [0.05], 0.36 [0.04], and 0.50 [0.05], respectively. The mean accuracy values with standard deviations for scenarios B2-B4 were 0.63 [0.09], 0.61 [0.07], and 0.55 [0.06].

Variations in performance indicators between the optimal substitute models and the baseline target models showed differences of 0.27, 0.18, 0.25, and 0.26 for accuracy, sensitivity, precision, and F-score, respectively, in scenario A. In contrast, scenario B recorded more minor variances of 0.04 for accuracy, 0.00 for sensitivity, 0.04 for precision, and 0.03 for F-score. The effective performance in scenario B is analyzed in detail in Section 7. Additionally, the mismatch in accuracy between the target model and the mean accuracies across all scenarios tested with the substitute models was substantial, at 0.38 for scenario A and 0.14 for scenario B, demonstrating the substitute models' inability to replicate the target model's performance in both scenarios.

4.3.6 Variable importance and leaked split points analysis

To evaluate the similarity between the feature importance of the target and substitute models, we provided box plots of the gain values of the features used by the target model in Figure 15 and Figure 16. Although the accuracy values reported in the previous subsection indicated that incorporating synthetic data into queries did not enhance the performance of substitute models, the variable importance graphs might suggest a different conclusion. As we worked with shallow ensembles and a low number of trees, it proved advantageous for an attacker to identify the features utilized by the target model. We included substitute models that achieved a validation accuracy greater than 85% in this analysis, and the stars in the graphs represented the gain values of the target model.

We computed two metrics for each scenario to assess the results of the feature importance comparison. The first metric calculates the average number of features standard to both target and substitute models, while the second metric averages the sum of gain values for the features utilized in both the target and substitute models.

For scenario A, the first metric yielded results of 1.03 [21%], 3.10 [62%], 4.47 [89%], and 4.06 [81%] for parts one to four, respectively, with the percentages in brackets representing the proportion of this metric relative to the number of features used in the target model. The corresponding readings for the second metric in scenario A were 0.22, 0.83, 0.93, and 0.99 for parts one through four.

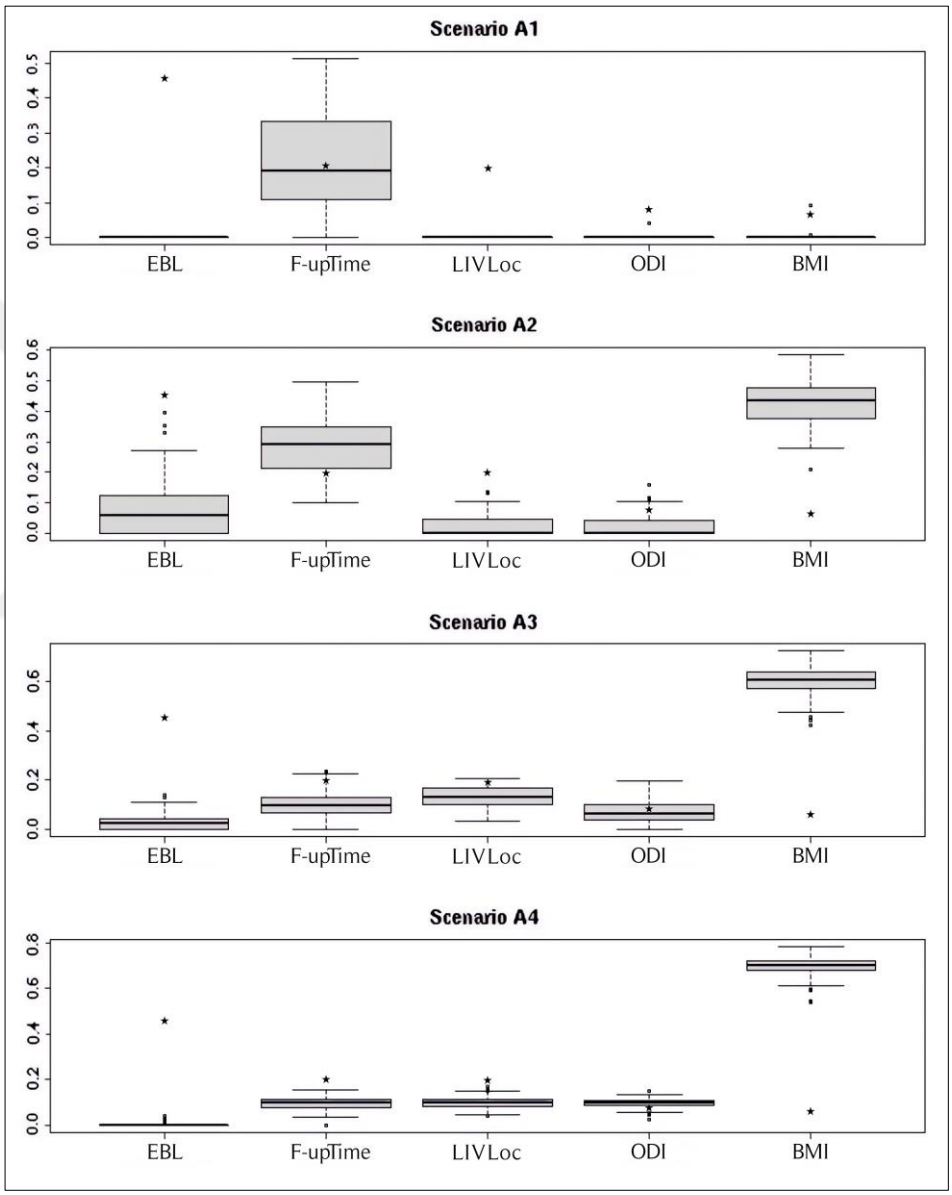

Figure 15: Substitute model gain values - scenario A

In scenario B, the values for the first metric were 1.51 [25%], 2.46 [41%], 4.76 [79%], and 4.56 [76%] for parts one through four, respectively. The values for the second metric were 0.92, 0.61, 0.71, and 0.89 for parts one to four in scenario B.

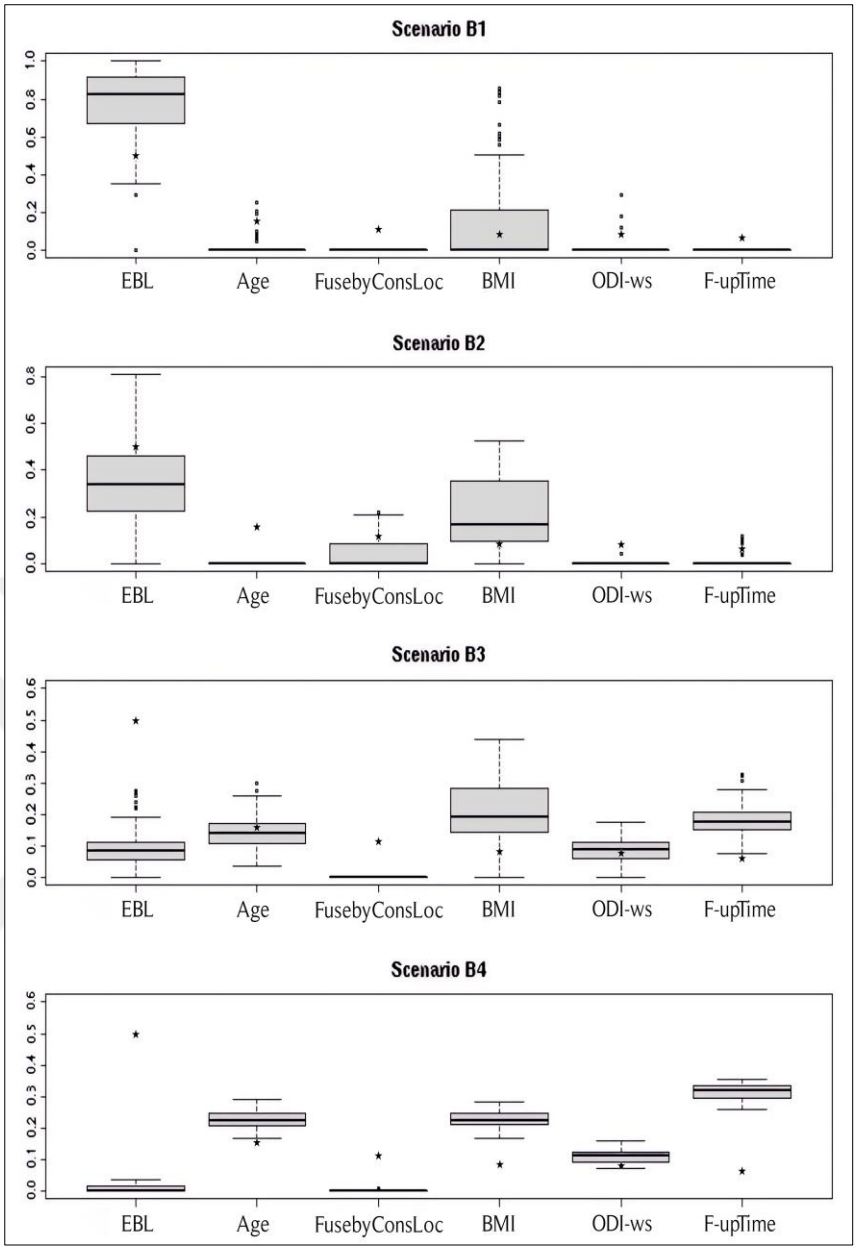

Figure 16: Substitute model gain values - scenario B

To further analyze the accurate usage of split points in the substitute models, we included heat maps in Figure 17. In Figure 17, the validation accuracy thresholds were displayed in the second column, determined through the unique usage of splits. In scenario B4, the column subsample parameter chosen during CV was 1. Consequently, three models met the threshold from the 27 final substitute models.

Scenario	Validation Accuracy Threshold	Number of models	EBL	F-upTime	LIVLoc	ODI-ws	BMI
A1	0.85	36	0.00	0.01	0.00	0.00	0.00
A2	0.90	31	0.01	0.18	0.06	0.03	0.77
A3	0.95	45	0.04	0.06	1.00	0.50	0.78
A4	0.99	27	0.00	0.00	1.00	0.50	0.80

Scenario	Validation Accuracy Threshold	Number of models	EBL	Age	Fuseby ConsLoc	BMI	ODI-ws	F-up Time
B1	0.99	64	0.19	0.00	0.00	0.11	0.00	0.00
B2	0.95	43	0.26	0.00	0.09	0.22	0.00	0.01
B3	0.95	25	0.23	0.26	0.00	0.28	0.40	0.48
B4	0.98	3*	0.33	0.50	0.00	0.33	0.50	0.50

Figure 17: Heat maps of the target model splits used in substitute models

We detailed the accurate usage of split points in the substitute models. In scenario A3, on average, 4.6 target model features were employed. All substitute models accurately used the target model's only LIVLoc split point and one of the two ODI-ws split points. The substitute models correctly used the target model's only BMI split point 35 times, one of the four EBL split points nine times, and one of the two F-upTime split points five times. In scenario A4, on average, four target model features were employed. All substitute models accurately used the target model's only LIVLoc split point and one of the two ODI-ws split points. The substitute models correctly used the target model's only BMI split point 21 times. However, none of the substitute models accurately used the target model's EBL and F-upTime split points.

In scenario B3, the substitute models, on average, incorporated 4.9 features from the target model. They accurately utilized one of the two available split points for F-upTime 24 times, one out of four for EBL 23 times, one of the two for ODI-ws 20 times, one of the two for BMI 14 times, and one of the two for age 13 times. In scenario B4, the substitute models averaged using five features from the target model. During the final phase of substitute model construction in this scenario, a column subsample parameter of one was set, leading to the generation of identical models across different seeds. These models accurately used one of the four EBL split points twice and another two times. They consistently applied one of the two split points for age and ODI-ws. Additionally, the models correctly identified one of the two split points for BMI and one of the two for F-upTime, each on two occasions.

4.3.7 File sizes and execution times of the PPML inference

We evaluated the target models' PPML inferences by comparing their prediction outcomes to those from an equivalent public model. To verify the accuracies of both the public and the privacy-preserving XGBoost models, we utilized the training datasets from the substitute models, which contained fifty-three and sixty-six patients in scenarios A and B, respectively. In both scenarios, the predictions from the privacy-preserving XGBoost models were the same as those from the original models.

In scenario A, the average time to prepare a single encoded query was recorded at 2.68 seconds, with a standard deviation of 0.09 seconds. The average encryption duration, including key generation, was measured at 4.50 seconds with a standard deviation of 0.05 seconds. This encryption process was followed by three calculation steps before the final output: node comparison value calculation, tree score calculation, and final masking of the value. The average duration for these steps was 0.17 seconds, 0.12 seconds, and 0.07 seconds, with standard deviations of 0.01, 0.01, and 0.01, respectively. The average duration for decryption in scenario A was 0.02 [0.001] seconds, and the total end-to-end duration for the entire process averaged 7.56 [0.11] seconds.

In scenario B, the average time needed to prepare a single encoded query was 2.70 seconds [0.06]. The average duration for the encryption process, including key generation, came to 4.48 seconds [0.05]. The average times for node comparison calculation, tree score calculation, and final masking were 0.17, 0.12, and 0.07 seconds, with standard deviations of 0.01, 0.02, and 0.004, respectively. The average decryption time was 0.02 seconds [0.001], with the total end-to-end process duration averaging 7.56 seconds [0.08]. The file sizes for the encoded input and model outputs were consistent across both scenarios, measuring 1.605 MB for the input and 191 kB for the output for each query.

4.3.8 RF attack with default parameters

We also constructed an alternative attack scenario using RF as model to built and reported the performance metrics in Table 22 for scenario A and Table 23 for scenario B, and we summarized them with box plots in Figure 18 and Figure 19. We hypothesized no prior ML knowledge and used the default parameter set. ACC, SENS,SPEC,PREC, and NPR stood for accuracy, sensitivity, specificity, precision, and negative precision rate, respectively. Means and standard deviations were presented. Blue lines in the figures represented the target model performance. All the training datasets used for this attack were 110 patient records in total.

Table 22: Attack with RF model with default parameters – scenario A

Tree Count	ACC-Mean	ACC-SD	SENS-Mean	SENS-SD	SPEC-Mean	SPEC-SD	PREC-Mean	PREC-SD	NPR-Mean	NPR-SD	F score Mean	F score-SD
25	0.63	0.08	0.24	0.17	0.79	0.09	0.30	0.18	0.71	0.06	0.26	0.16
50	0.65	0.07	0.21	0.13	0.83	0.08	0.32	0.21	0.71	0.04	0.25	0.16
100	0.63	0.05	0.21	0.12	0.81	0.07	0.31	0.16	0.71	0.03	0.25	0.12
150	0.66	0.04	0.23	0.11	0.84	0.05	0.36	0.15	0.72	0.03	0.27	0.12
200	0.66	0.04	0.22	0.10	0.85	0.06	0.38	0.15	0.72	0.03	0.27	0.11
250	0.65	0.04	0.24	0.08	0.83	0.05	0.37	0.10	0.72	0.02	0.29	0.08
300	0.66	0.05	0.21	0.07	0.84	0.06	0.38	0.13	0.72	0.03	0.27	0.08
350	0.66	0.05	0.22	0.06	0.85	0.06	0.39	0.11	0.72	0.02	0.28	0.07
400	0.66	0.04	0.21	0.04	0.86	0.05	0.39	0.09	0.72	0.01	0.27	0.04
450	0.66	0.04	0.23	0.08	0.84	0.05	0.39	0.09	0.73	0.02	0.29	0.07
500	0.66	0.04	0.21	0.05	0.84	0.06	0.38	0.11	0.72	0.02	0.27	0.06

Table 23: Attack with RF model with default parameters – scenario B

Tree Count	ACC-Mean	ACC-SD	SENS-Mean	SENS-SD	SPEC-Mean	SPEC-SD	PREC-Mean	PREC-SD	NPR-Mean	NPR-SD	F score Mean	F score-SD
25	0.66	0.05	0.71	0.10	0.65	0.07	0.29	0.05	0.92	0.03	0.41	0.06
50	0.66	0.04	0.68	0.10	0.65	0.05	0.28	0.03	0.91	0.02	0.40	0.04
100	0.65	0.04	0.66	0.09	0.65	0.05	0.27	0.03	0.91	0.02	0.39	0.04
150	0.65	0.04	0.67	0.10	0.65	0.04	0.28	0.04	0.91	0.03	0.39	0.05
200	0.66	0.04	0.67	0.10	0.66	0.05	0.28	0.03	0.91	0.02	0.39	0.04
250	0.64	0.03	0.65	0.09	0.64	0.03	0.26	0.03	0.90	0.02	0.37	0.04
300	0.65	0.03	0.67	0.10	0.65	0.04	0.28	0.03	0.91	0.02	0.39	0.05
350	0.64	0.03	0.63	0.08	0.64	0.03	0.26	0.03	0.90	0.02	0.37	0.04
400	0.65	0.03	0.70	0.10	0.65	0.03	0.28	0.03	0.92	0.03	0.40	0.05
450	0.65	0.02	0.65	0.09	0.64	0.02	0.27	0.03	0.90	0.02	0.38	0.04
500	0.65	0.02	0.67	0.10	0.64	0.02	0.27	0.03	0.91	0.03	0.39	0.05

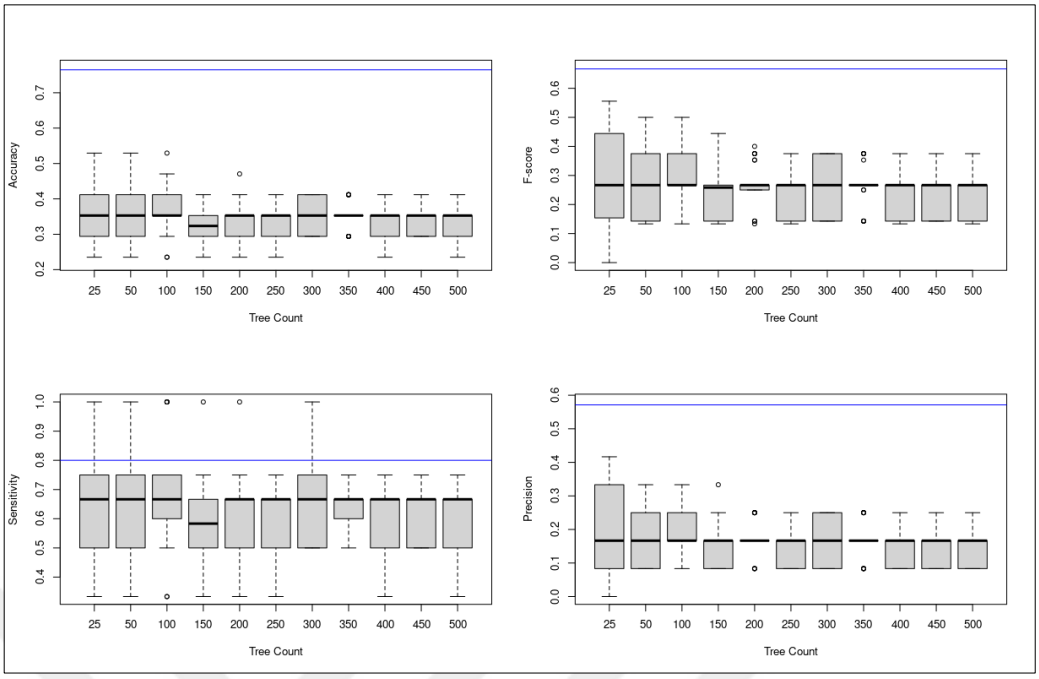

Figure 18: Box plots of RF attack performance metrics – scenario A

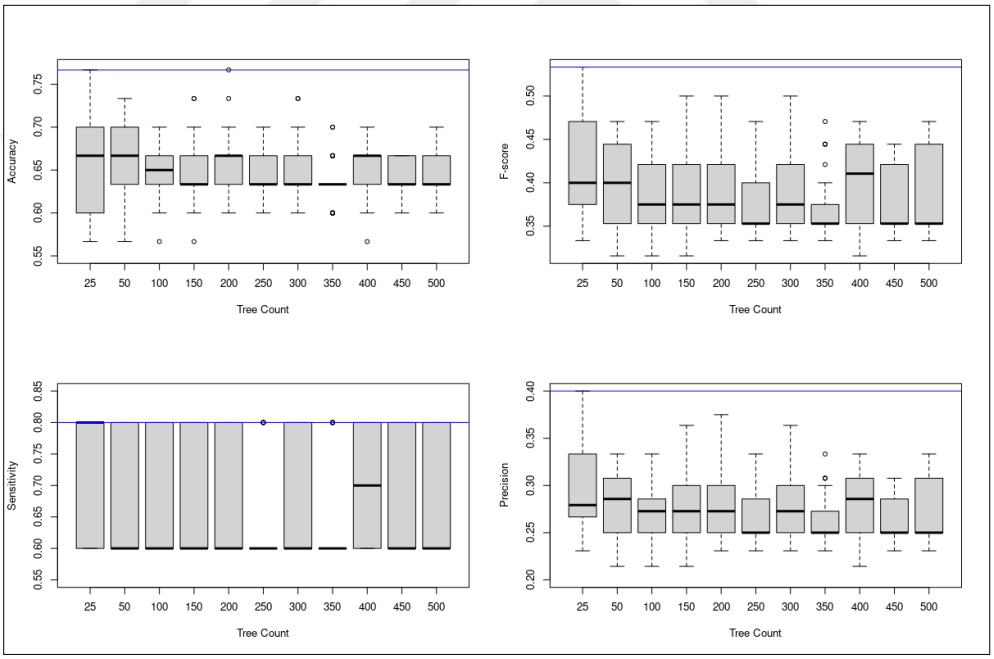

Figure 19: Box plots of RF attack performance metrics – scenario B

4.4 The Security Analysis of Privacy-Preserving RF Inference

4.4.1 Performance metrics of the target model

The target models were trained using parameters found in the RF modeling section except for the node size parameter, which was configured to 12 for both scenarios with a decrease proportioned to the decrease in the number of training patient records. The confusion matrices for both the training and test datasets were displayed in Table 24 and 25, while the related performance metrics were in Table 26.

Table 24: Confusion matrices – Scenario A

		Training set		Test Set (Target)		Test Set (Attacker)	
Scenario	**Pred Real**	**Healthy**	**Mech. Comp.**	**Healthy**	**Mech. Comp.**	**Healthy**	**Mech. Comp.**
A	**Healthy**	103	0	29	2	12	0
	Mech. C.	30	50	14	16	11	16

Table 25: Confusion matrices – Scenario B

		Training set		Test Set (Target)		Test Set (Attacker)	
Scenario	**Pred Real**	**Healthy**	**Mech. Comp.**	**Healthy**	**Scenario**	**Pred Real**	**Healthy**
B	**Healthy**	84	1	23	3	16	2
	Mech. Comp.	40	64	18	21	7	9

Table 26: Target model performance metrics

Scenario	**Set**	**ACC**	**TPR**	**TNR**	**Precision**	**NPR**	**F-score**
A	**Training**	0.84	1.00	0.77	0.63	1.00	0.77
	Test-Target	0.74	0.89	0.67	0.53	0.94	0.67
	Test-Attacker	0.72	1.00	0.52	0.59	1.00	0.74
B	**Training**	0.78	0.98	0.68	0.62	0.99	0.76
	Test-Target	0.68	0.88	0.56	0.54	0.89	0.67
	Test-Attacker	0.74	0.82	0.70	0.56	0.89	0.67

We reported the feature importance graph in Figure 20 for scenario A and in Figure 21 for scenario B.

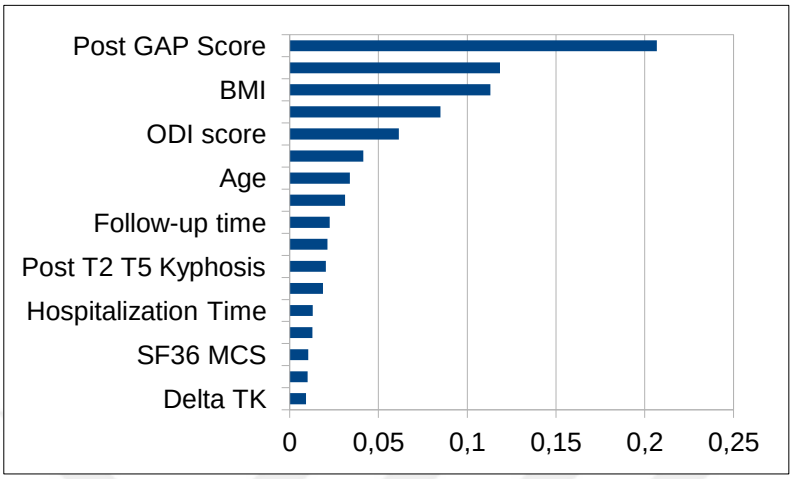

Figure 20: Feature importance graph of target model scenario A

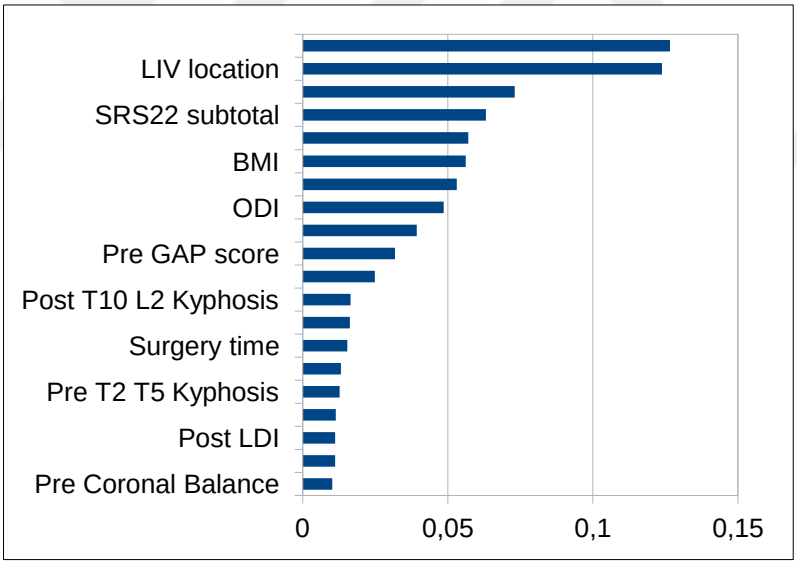

Figure 21: Feature importance graph of target model scenario B

4.4.2 Performance metrics of substitute models

In Figure 22 and Figure 23, the default parameters were used to construct substitute models with different numbers of trees in the ensemble. The blue lines indicated the performance metrics of the target model on the attacker test set.

For scenario A, at the smallest tree count of 25, the average accuracy was 0.70, with a standard deviation of 0.03. This accuracy remained consistent, peaking at 0.72 as the tree count increased, demonstrating the stability of the model across different configurations. Sensitivity started at 0.98 for the smallest tree count. It reached a perfect score of 1.00 from 100 trees onward, indicating that the model became better at identifying true positives as more trees were used. The precision and F-score metrics started at 0.58 and 0.73, respectively, for 25 trees and showed minor improvements as the number of trees increased, stabilizing at 0.59 for precision and 0.74 for F-score at higher tree counts. This suggested that the model's ability to avoid false positives improved slightly with more trees. A similar observation was seen in scenario B, which had slightly worse performance metrics except for sensitivity. We observed that attackers have an advantage in decrypting the model using even default parameters if they are allowed to query their complete data set.

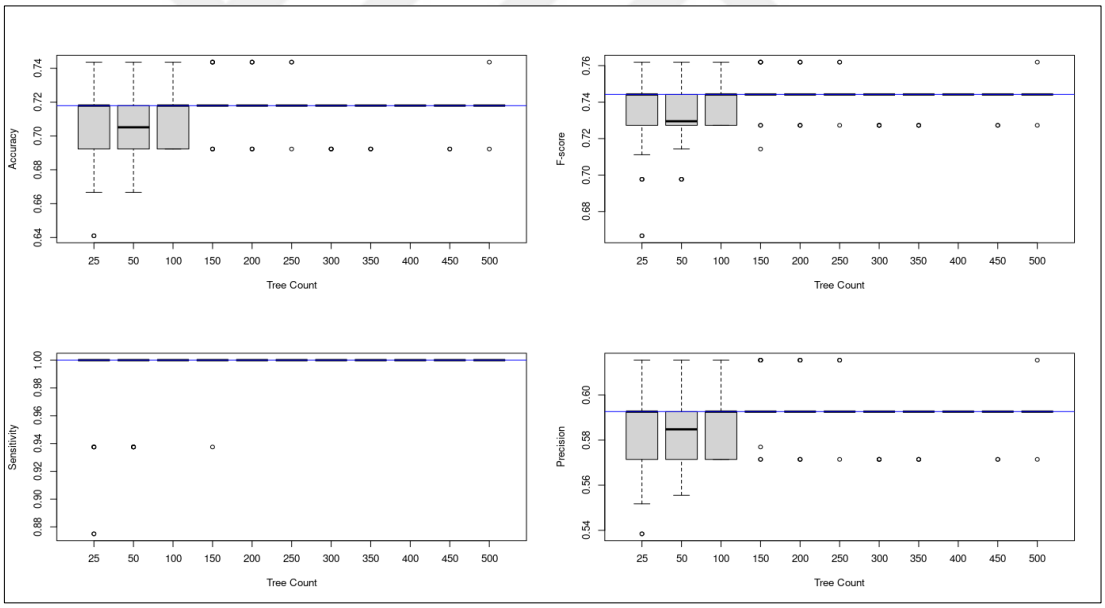

Figure 22: Performance metrics of substitute models with default parameters – scenario A

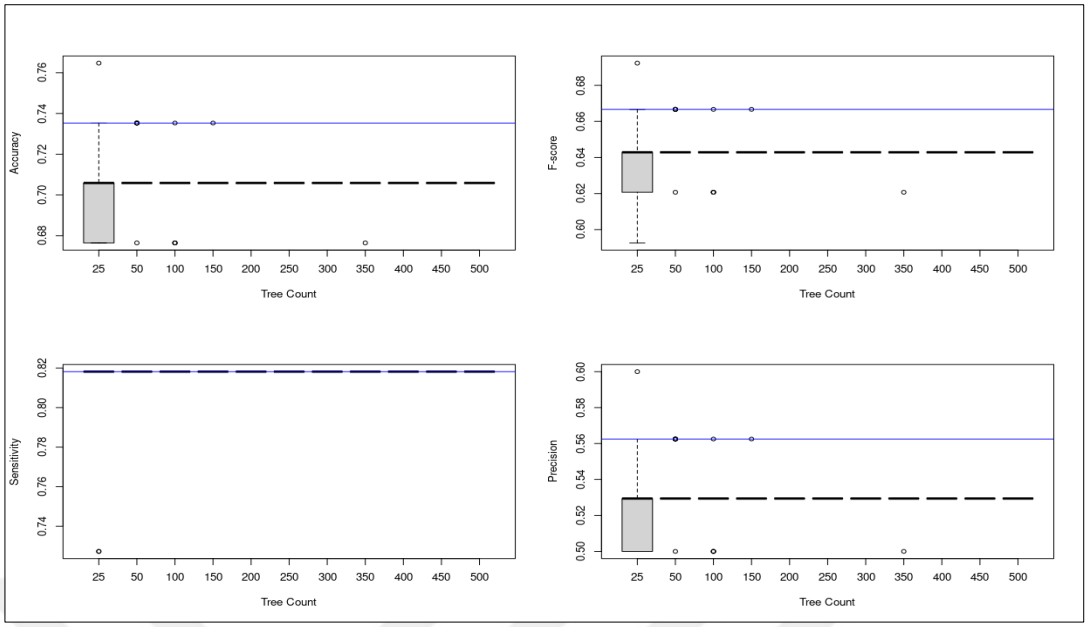

Figure 23: Performance metrics of substitute models with default parameters – scenario B

As explained in the methods section, we also compare feature importances. For scenario A and five hundred trees, we observed that the post-GAP score held the highest target importance among all features, contributing 20.7% to model predictions and ranking first overall, despite being estimated fifth in attacker importance ranking. LIV location, which was ranked first, was second in its contribution at 11.8%. BMI and EBL also had signals that the attacker missed. Conversely, the SRS22 subtotal was ranked second in attacker's importance at 10.8% but dropped to sixth place with a target importance of 4.1%. The ranking of all feature importance was given in Table 27.

Table 27: Feature importance ranking of substitute model with default parameters – scenario A

Rank	Feature	Importance Mean	Target Importance	Target Rank
1	LIV location	12.4%	11.8%	2
2	SRS22 subtotal	10.8%	4.1%	6
3	Pre GAP score	5.6%	2.1%	10
4	Age	5.0%	3.4%	7
5	Post GAP Score	5.0%	20.7%	1
6	ODI score	4.9%	6.2%	5
7	BMI	4.3%	11.3%	3
8	EBL	4.0%	8.5%	4
9	TL_L Apical Translation to CSVL	2.6%	3.1%	8
10	Hospitalization Time	2.3%	1.3%	13
11	SF36 MCS	1.6%	1.0%	15
12	Follow-up time	1.4%	2.3%	9
13	Occupation	0.9%	1.0%	16
14	Delta TK	0.8%	0.9%	17
15	Pre Coronal Balance	0.7%	1.9%	12
16	Pre Leg Length Discrepancy	0.7%	1.3%	14
17	Post T2 T5 Kyphosis	0.7%	2.0%	11

In our analysis of the substitute model with default parameters for scenario B, we observed significant variances between the perceived and actual importance of various features. Notably, LIV location, which emerged as the most important feature in the substitute model with a mean importance of 19.7%, was actually ranked second in the target model with 12.4% importance. Conversely, age, which was the most critical feature in the target model with 12.7% importance, was ranked second in the substitute model, reflecting a slight underestimation at 12.3%. The third-ranked feature, SRS22 subtotal, with a mean importance of 9.6% in the substitute model, was originally fourth in the target model at 6.3%, indicating a higher perceived value in the substitute model. Pre GAP score, notably ranked fourth in the substitute model at 7.6%, had been significantly underestimated in the target model where it was tenth with only 3.2% importance. Further discrepancies were evident in other features such as BMI and EBL, which were ranked fifth and sixth in the substitute model but held higher importance in the target model rankings of sixth and third, respectively. Moreover, the Post GAP score, which was notably seventh in importance in the target model at 5.3%, dramatically dropped to fifteenth in the substitute model with a mere 0.7% importance.

Next, in Figure 24 and Figure 25, the default parameters, as well as a balanced dataset and F-score performance metric were used to construct substitute models with different numbers of trees in the ensemble. The blue lines indicated the performance metrics of the target model on the attacker test set. We observed that the performance metrics in scenario A dropped a little except the sensitivity metric, and no change in scenario B. Finally, we presented the performance metrics of the best ten substitute models sorted with F-score and the mean values of 100 random parameter trials in Figure 26 and Figure 27.

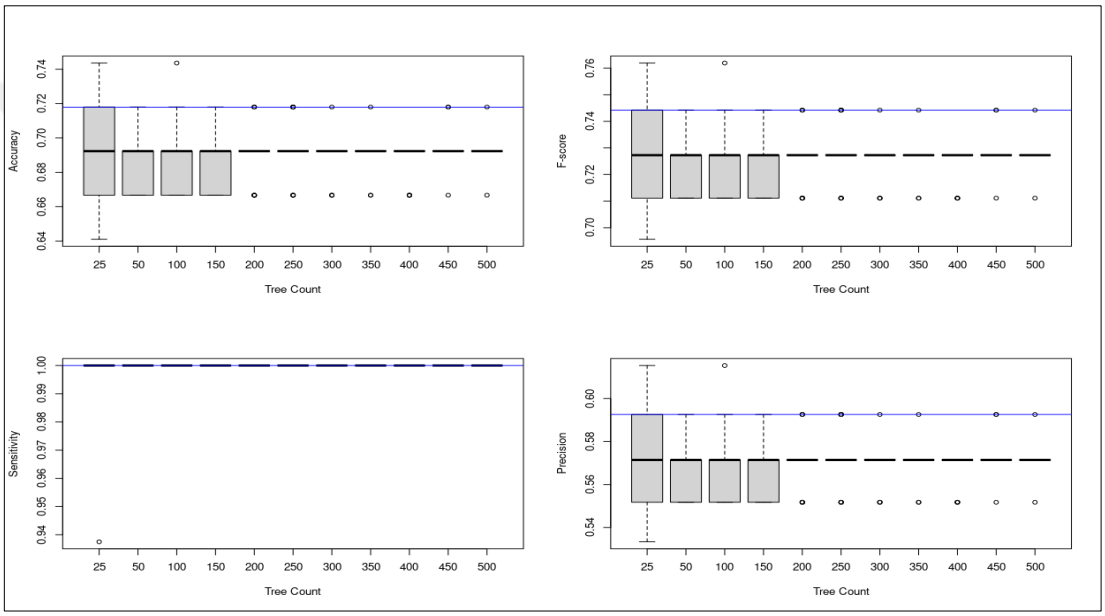

Figure 24: Performance metrics of substitute models with improved default parameters – scenario A

It was visible that 10% of the times the models performed outstanding scores, even better than target model predictions in scenario A. In scenario A, mean values of accuracies were ranging 0.74 to 0.75, sensitivities were ranging 0.99 to 1.00, precision ranging 0.61 to 0.62 and F-score ranging 0.76 to 0.77. In scenario B, mean values of accuracies were ranging 0.71 to 0.74, sensitivities were 0.82, precision ranging 0.53 to 0.56 and F-score ranging 0.65 to 0.67. The parameters of the models were presented in Table 28 and Table 29.

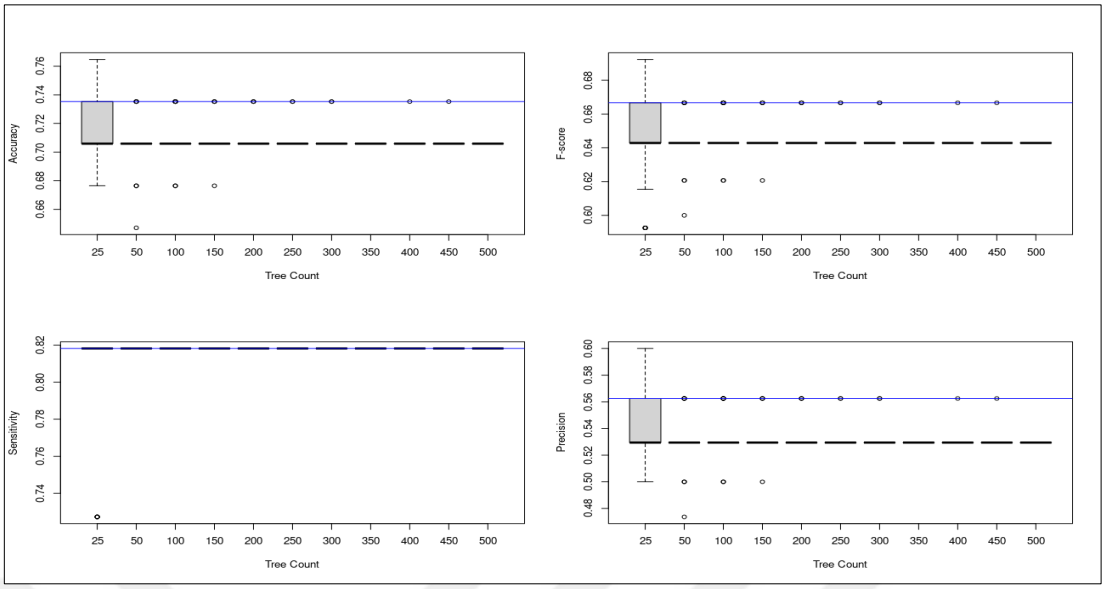

Figure 25: Performance metrics of substitute models with improved default parameters – scenario B

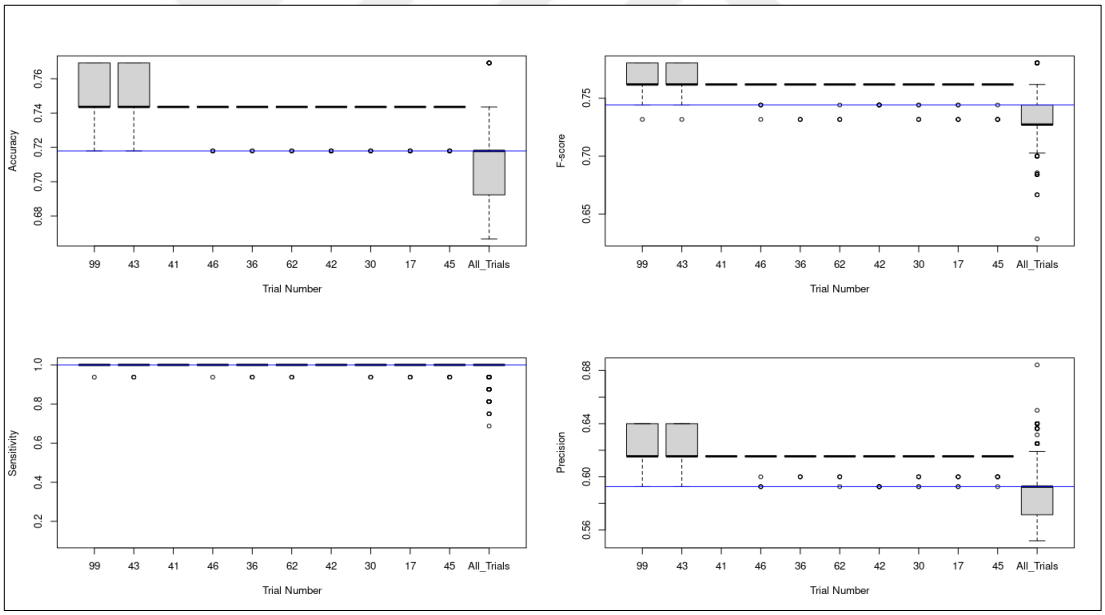

Figure 26: Performance metrics of the best ten substitute models with random parameters – scenario A

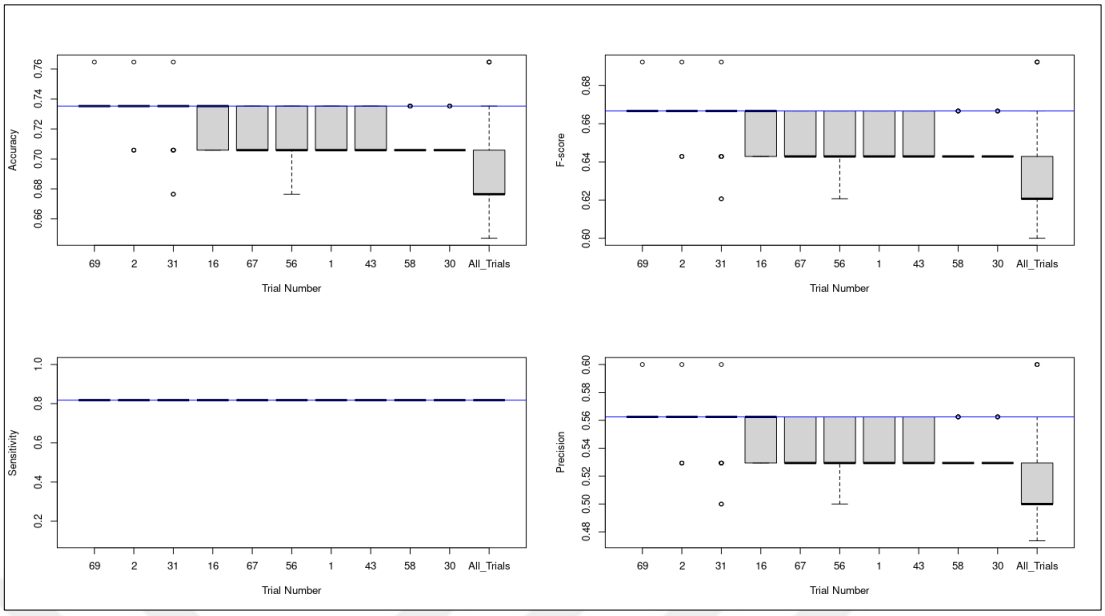

Figure 27: Performance metrics of the best ten substitute models with random parameters – scenario B

Table 28: Parameters of the best-performing models – scenario A

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	99	617	33	5	15	F	balanced
2	43	185	35	5	9	F	balanced
3	41	952	20	18	20	F	balanced
4	46	495	13	20	16	F	balanced
5	36	700	13	14	17	AUC	balanced
6	62	445	21	18	20	AUC	balanced
7	42	720	26	19	20	AUC	balanced
8	30	469	16	16	18	AUC	balanced
9	17	347	17	12	18	F	none
10	45	354	15	14	19	AUC	balanced

Table 29: Parameters of the best-performing models – scenario B

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	69	356	46	2	19	AUC	none
2	2	905	45	2	8	AUC	balanced
3	31	252	33	2	9	AUC	balanced
4	16	609	59	16	3	AUC	balanced
5	67	114	59	13	1	F	none
6	56	917	58	20	10	F	balanced
7	1	726	54	6	4	F	balanced
8	43	208	57	9	5	AUC	none
9	58	397	59	6	2	AUC	none
10	30	692	53	9	7	AUC	balanced

4.4.3 Finding a safe threshold to protect the privacy of the target model

We concluded that privacy was susceptible to leaks in both scenario models. Thus, we gradually decreased the number of queries that attackers send to the target models. We conducted the same procedures in the previous section with a partial of the original attacker training set. In this section, we presented three different numbers of queries: 30, 40, and 50. Since feature importances differed in the previous section, we excluded feature importance analysis in this section.

In Figure 28, Figure 29, and Figure 30, the box plots of the performance metrics of the substitute models built with the default parameters and with different numbers of trees in the ensemble for scenario A were shown for 30, 40, and 50 allowed queries. We observed the performance metrics across a range of tree counts from 25 to 500 in our predictive models. The blue lines indicated the performance metrics of the target model on the attacker test set.

With 30 queried data, initially, at a tree count of 25, the accuracy averaged 0.45 with a standard deviation of 0.04. This accuracy declined slightly, stabilizing at 0.41 by the tree count 500. The sensitivity, however, remained exceptionally high across all tree counts, achieving a perfect score of 1.00 from 100 trees onward. The specificity started at a notably low value of 0.08 for 25 trees and decreased further as the number of trees increased, reaching 0.00 for tree counts of 450 and 500. The F-score, which balances precision and sensitivity, was consistently around 0.58 to 0.60. Except for good sensitivity results, substitute models failed to achieve target model performance metrics, as the blue lines indicated.

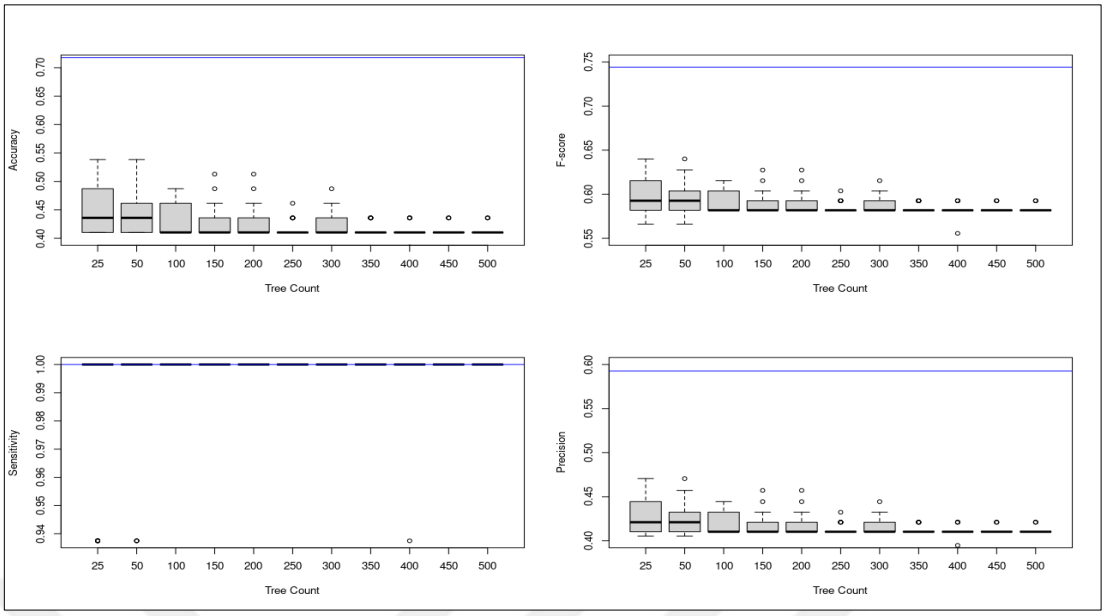

Figure 28: Performance metrics of substitute models with default parameters and 30 queried data – scenario A

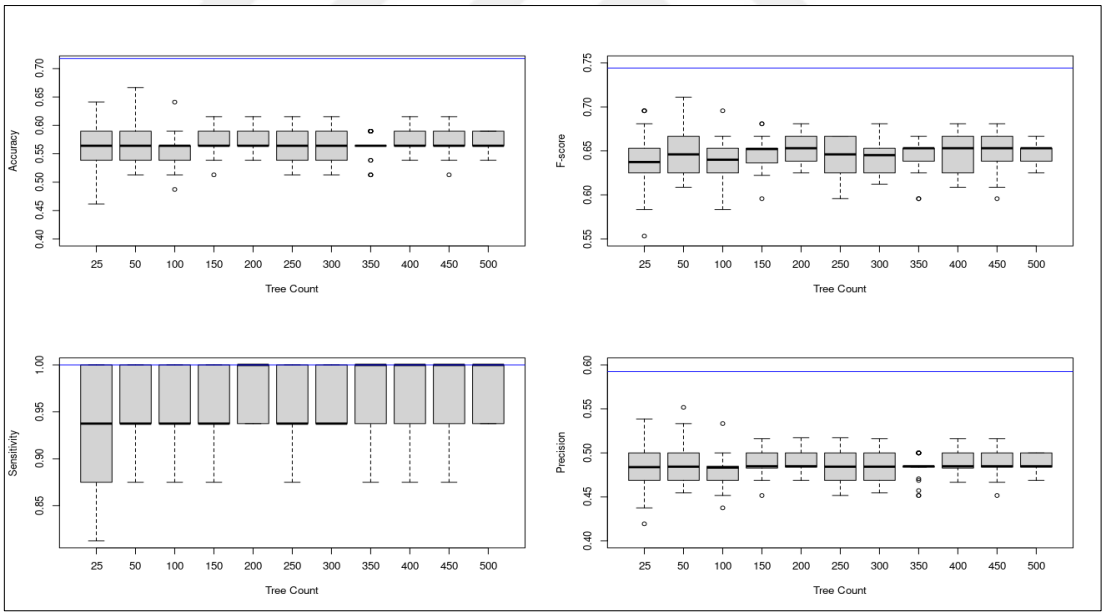

Figure 29: Performance metrics of substitute models with default parameters and 40 queried data – scenario A

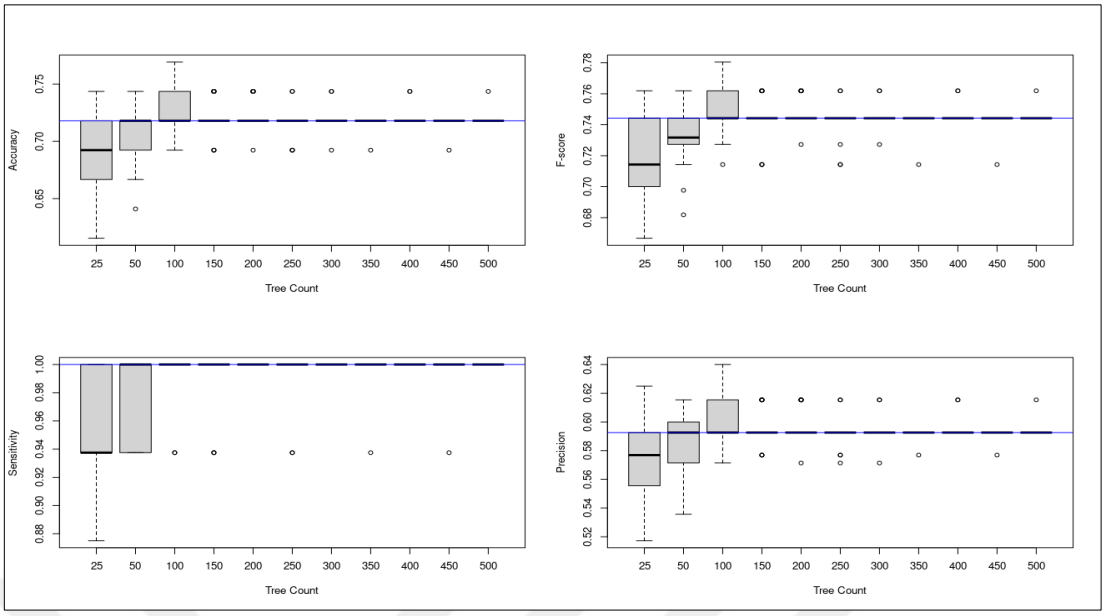

Figure 30: Performance metrics of substitute models with default parameters and 50 queried data – scenario A

For the 40-query-attacker-set, initially, at 25 trees, the average accuracy was recorded at 0.57 with a standard deviation of 0.05. As the tree count increased, the accuracy remained relatively stable, consistently hovering around 0.57. The sensitivity, starting at 0.93 for 25 trees, improved slightly as more trees were added, reaching 0.98 for 350 and 500 tree counts. This suggested that the model became better at identifying true positives as the number of trees and queries increased. Specificity began at 0.31 for 25 trees and showed slight fluctuations, generally decreasing slightly to 0.28 by the 500-tree mark, highlighting a consistent challenge in correctly identifying negative cases across different tree configurations. Precision remained steady around 0.49 across all tree counts, suggesting a moderate ability to avoid false positives that did not significantly improve with more trees. The negative predictive rate showed slight improvement from 0.87 at 25 trees to 0.95 at 350 and 500 trees, reflecting better performance in predicting negative outcomes as the tree count increased. The F-score, which balances the precision and sensitivity, was initially 0.64 at 25 trees and slightly improved to 0.65 at higher tree counts. Compared to the 30-query-attacker-set, the substitute models had higher accuracy, F-score, and precision metrics observed, but not as good as the target model.

For the 50-query-attacker-set, we observed all the performance metrics achieved by the target model were also achieved by the substitute models as the blue lines indicated. Refer to subsection 4.4.1 for the target model performance metrics.

In Figure 31, Figure 32, and Figure 33, the performance metric box plots of the substitute models built with default parameters and with different numbers of trees in the ensemble for scenario B were shown for 30,40, and 50 allowed queries. We observed the performance metrics across a range of tree counts from 25 to 500 in our predictive models. The blue lines indicated the target model metrics on the attacker test set.

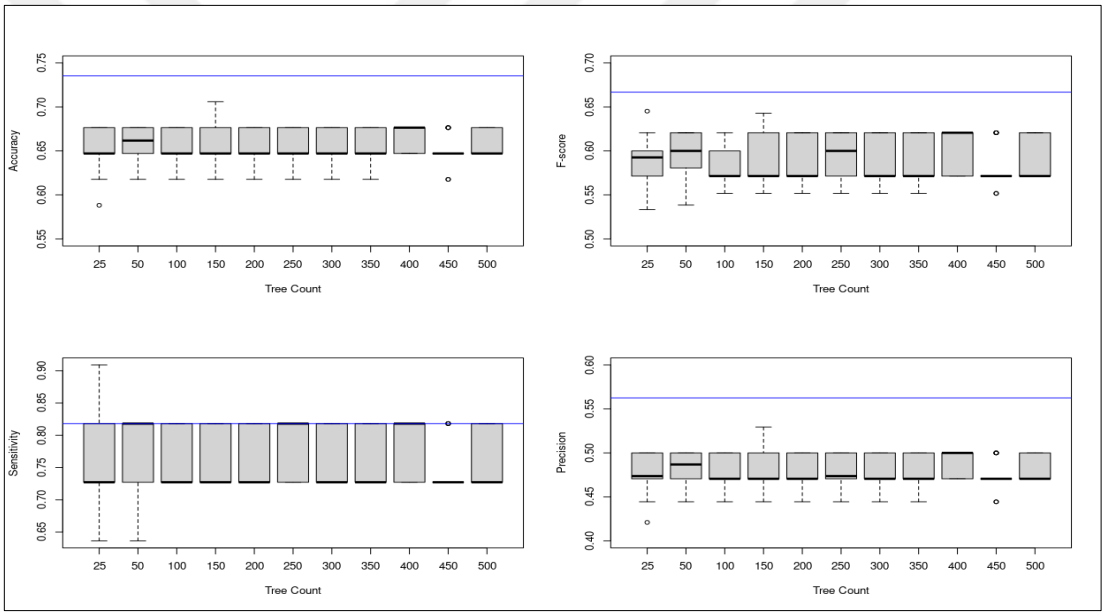

Figure 31: Performance metrics of substitute models with default parameters and 30 queried data – scenario B

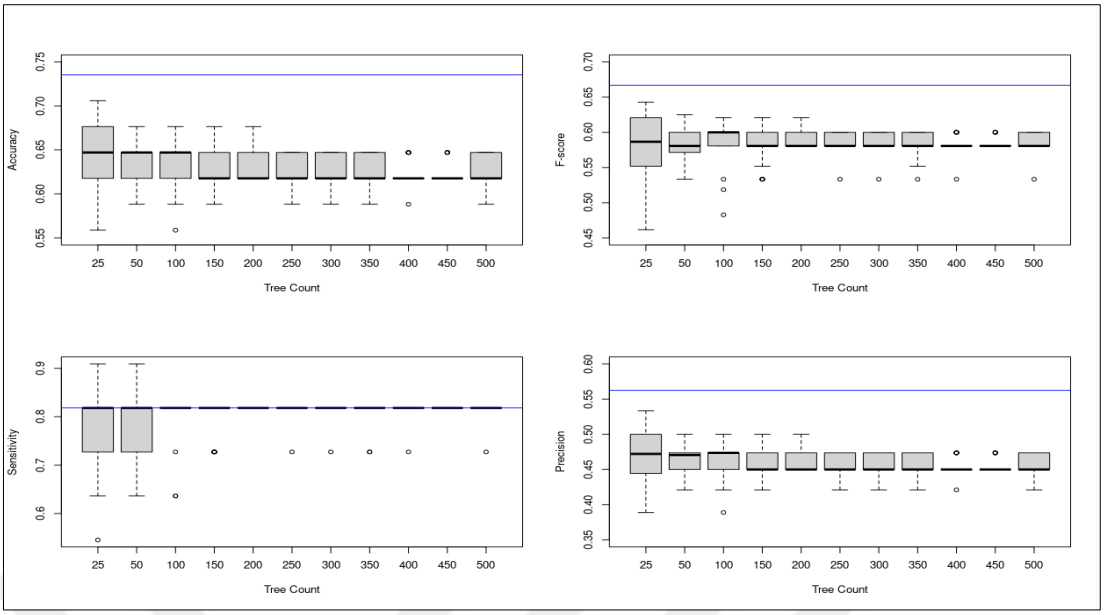

Figure 32: Performance metrics of substitute models with default parameters and 40 queried data – scenario B

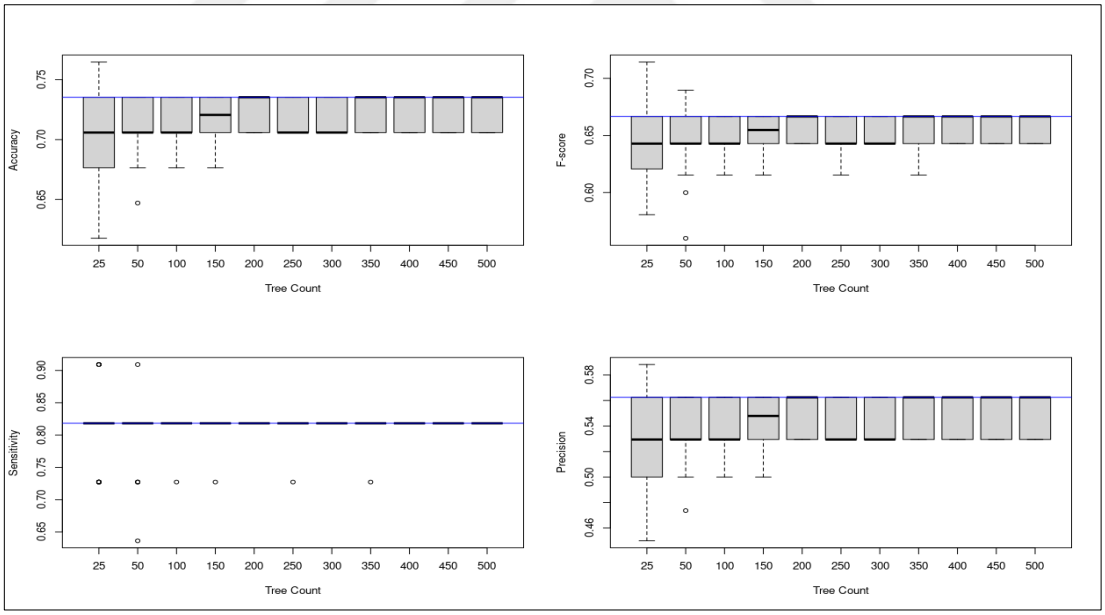

Figure 33: Performance metrics of substitute models with default parameters and 50 queried data – scenario B

In scenario B for the 30-query-attacker-set, the average accuracy remained consistent at approximately 0.66, demonstrating the model's stable performance irrespective of tree count increases. Sensitivity varied slightly around 0.76 to 0.79,

reflecting a fairly consistent ability to identify true positives, with a peak at 400 trees showing a sensitivity of 0.78. Specificity consistently hovered around 0.61, indicating a stable actual negative identification rate across all tree configurations. The precision, staying near 0.48, suggested a modest capability of the model to minimize false positives, with a slight improvement to 0.49 observed at 400 trees. The negative predictive rate showed minimal variation, generally around 0.84 to 0.86, signifying consistent predictiveness for negative outcomes. The F-score was maintained at approximately 0.59, slightly increasing to 0.60 at 50 and 400 trees. Compared to scenario A, substitute models achieved better performances but were still not as high as the target model performance metrics, as indicated with blue lines.

In scenario B for the 40-query-attacker-set, we examined the accuracy mean consistently hovered around 0.63, demonstrating a stable performance across different tree configurations. Sensitivity gradually improved from 0.77 at 25 trees to 0.82 at 200 trees. It remained consistent thereafter, indicating that the model became increasingly proficient at identifying true positives as the number of trees increased. Specificity started at 0.58 and slightly decreased to 0.53, suggesting a modest decline in the model's ability to identify true negatives as more trees were used. Precision was relatively stable, around 0.46 to 0.47. The negative predictive rate was relatively stable, mostly around 0.86. The F-score remained around 0.58 to 0.59. The attack produced similar results compared to the 30-query-attacker-set attack. Furthermore, we rejected the null hypothesis that the F-score of the substitute models was greater than or equal to the target model's F-score with a p-value of 0.

In scenario B, for the 50-query-attacker set, we observed that all the performance metrics achieved by the target model were also achieved by the substitute models, especially when we increased the tree count to 350 and above, as the blue lines indicated. Refer to subsection 4.4.1 for the target model performance metrics.

Next, we checked the performance metrics of the substitute models if we built the models with default parameters but with a balanced class weight and F-score as a performance metric. In Figure 34, Figure 35, and Figure 36, the performance metrics

box plots of the substitute models for scenario A were shown for 30,40, and 50 allowed queries. We observed the performance metrics across a range of tree counts from 25 to 500 in our predictive models.

In scenario A with 30-query-attacker-data, as the tree count increased from 25 to 500, the average accuracy remained relatively constant at approximately 0.42 to 0.43, indicating stable performance across different model complexities. The sensitivity achieved a perfect score of 1.00 from 100 trees onward, demonstrating the model's consistent ability to identify true positives as more trees were added correctly. Specificity was notably low, hovering around 0.01 to 0.03, suggesting that while the model was good at identifying true positives, it struggled significantly with false positives. Precision also remained consistent, around 0.41 to 0.42. The NPR maintained a perfect score of 1.00 from 100 trees onward. The F-score was consistent around 0.58 to 0.59. Except for good sensitivity results, substitute models failed to achieve target model performance metrics, as blue lines indicated in Figure 34.

In scenario A with 40-query-attacker-data, the average accuracy slightly improved as the tree count increased from 0.61 at 25 trees to 0.63 at 500 trees. Sensitivity and NPR reached perfection at 1.00 from 100 trees onward. Specificity saw slight fluctuations but generally hovered around 0.36 to 0.38, suggesting the model had moderate success in correctly identifying true negatives, with a minor improvement as the tree count increased. Precision slightly improved as well, stabilizing at 0.53 in the latter counts. The F-score remained consistently around 0.68 to 0.69, indicating a good balance between sensitivity and precision across all tree counts.

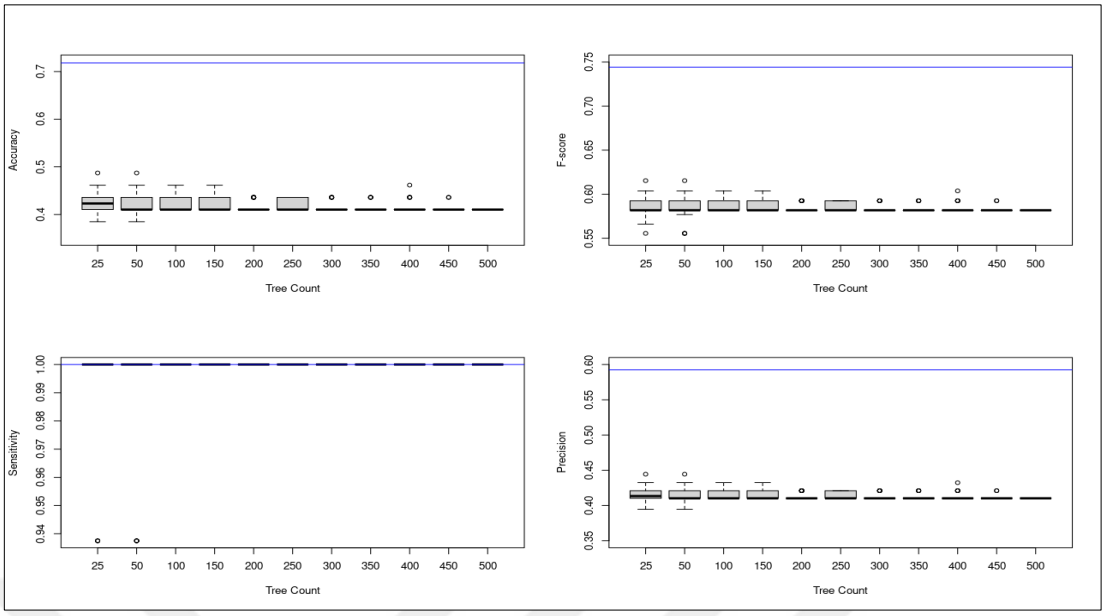

Figure 34: Performance metrics of substitute models with altered default parameters and 30 queried data – scenario A

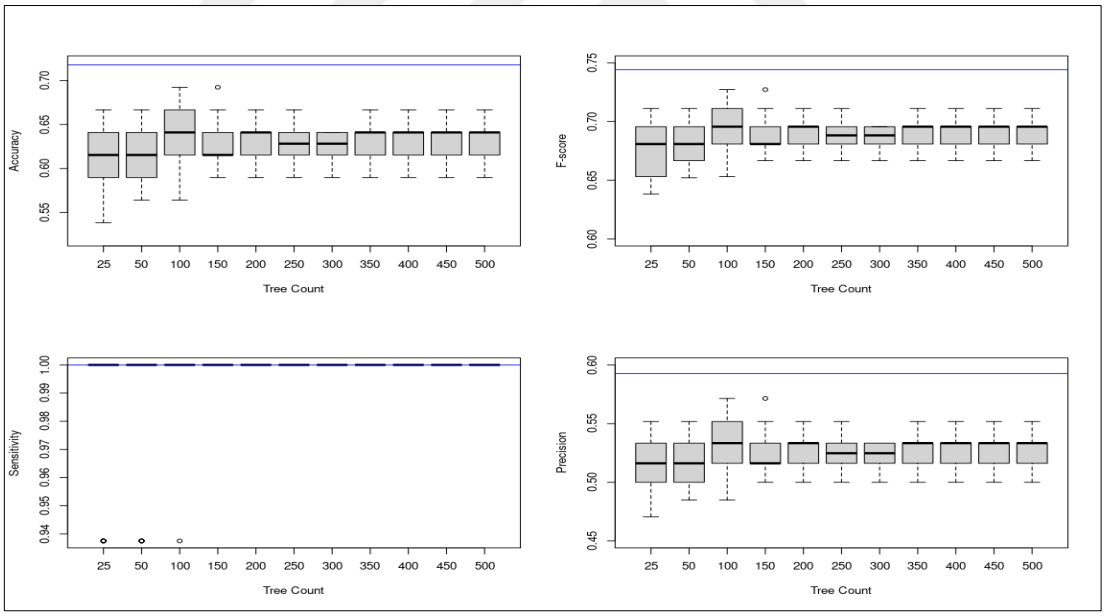

Figure 35: Performance metrics of substitute models with altered default parameters and 40 queried data – scenario A

Even though performance metrics slightly improved compared to the 30-query-attacker-data, substitute models failed to achieve target model performance metrics as blue lines indicated in Figure 35. Furthermore, we rejected the null hypothesis that F-

score of the substitute models were greater than or equal to target model F-score with a p-value 0.

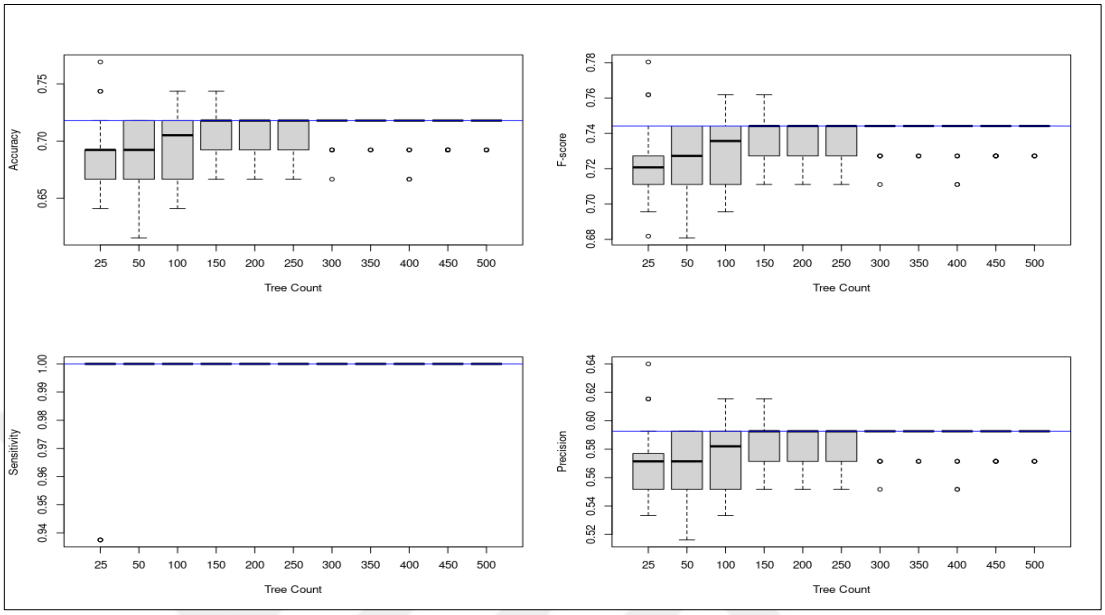

Figure 36: Performance metrics of substitute models with altered default parameters and 50 queried data – scenario A

In scenario A for the 50-query-attacker-set, we observed all the performance metrics achieved by the target model were also achieved by the substitute models, especially when we increased the tree count to 150 and above as seen in Figure 36, as the blue lines indicated. Refer to subsection 4.4.1 for the target model performance metrics.

In Figure 37, Figure 38, and Figure 39, substitute models performance metrics box plots for scenario B were shown for 30,40, and 50 allowed queries. We observed the performance metrics across a range of tree counts from 25 to 500 in our predictive models.

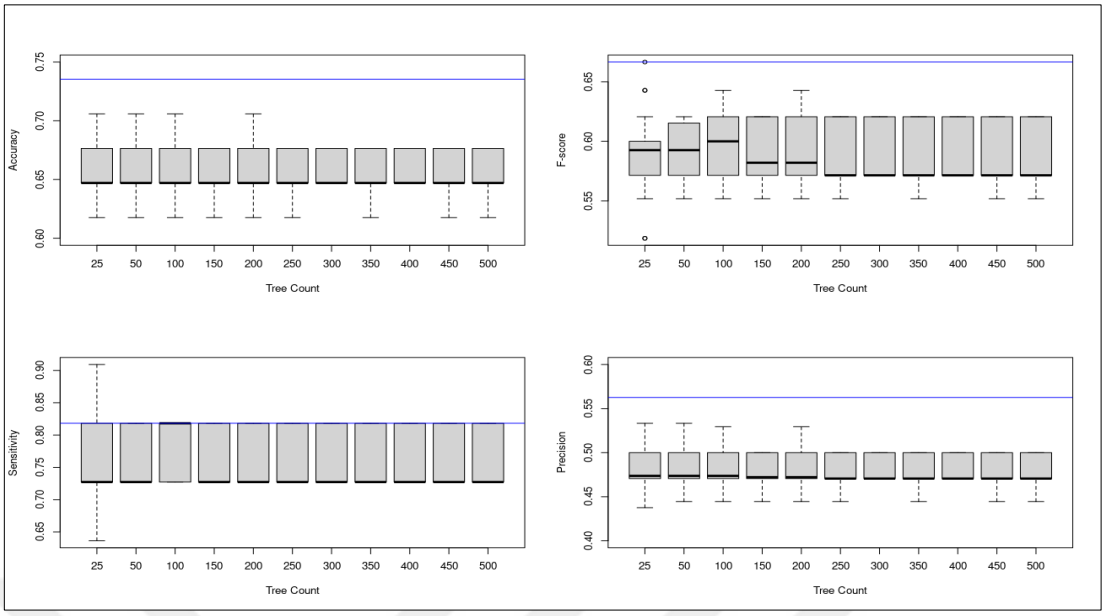

Figure 37: Performance metrics of substitute models with altered default parameters and 30 queried data – scenario B

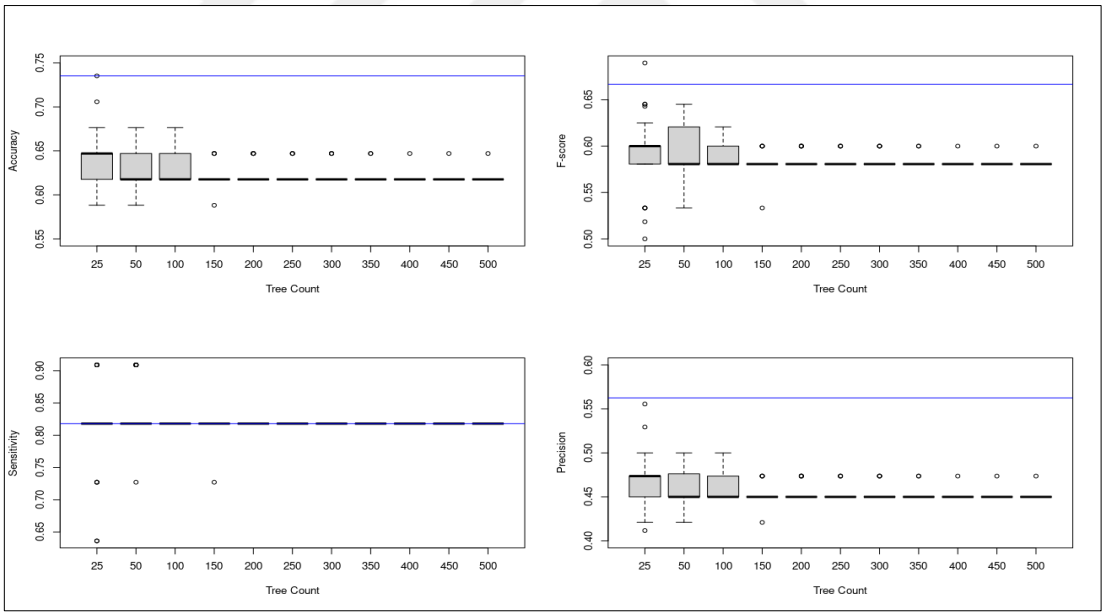

Figure 38: Performance metrics of substitute models with altered default parameters and 40 queried data – scenario B

In scenario B with 30-query-attacker-data, the average accuracy remained consistent around 0.66, with slight variations from 0.65 to 0.66 as the tree count increased from 25 to 500. Sensitivity showed slight variation, mostly staying between

0.75 and 0.78. The specificity also showed stability, mainly fluctuating around 0.60 to 0.61. Precision maintained a level around 0.48, while the NPR consistently reported scores around 0.84. The F-score remained relatively stable, around 0.59. Compared to scenario A, substitute models achieved better performances, but still not as high as the target model performance metrics as indicated with blue lines.

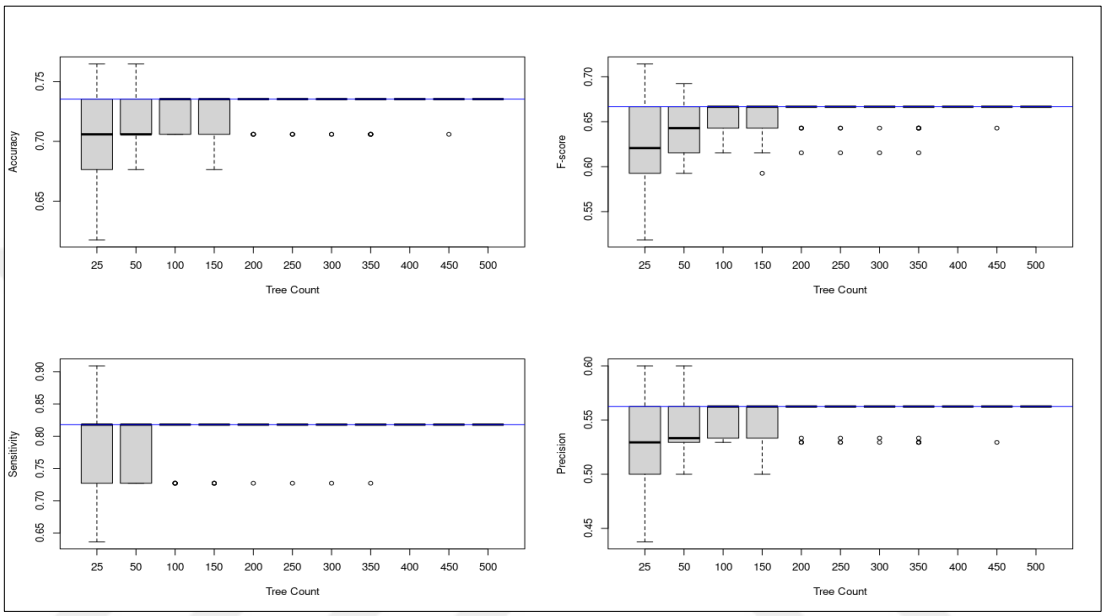

Figure 39: Performance metrics of substitute models with altered default parameters and 50 queried data – scenario B

In scenario B with 40-query-attacker-data, the substitute models presented a consistent performance across various tree counts for a predictive model. The accuracy, sensitivity, and specificity metrics remained largely stable across the tree counts, around 0.62, 0.82, and approximately 0.52, respectively. Precision and the NPR also showed slight variation, maintaining values around 0.45 and 0.86, respectively. The F-score across all tree counts was consistently around 0.58. Even though performance metrics slightly improved compared to 30-query-attacker-data, substitute models failed to achieve target model performance metrics except for sensitivity, as blue lines indicated in Figure 38. Moreover, the two-tailed and left tailed t-tests resulted with 0 p-value.

In scenario B for the 50-query-attacker-set, we observed all the performance metrics achieved by the target model were also achieved by the substitute models, especially when we increased the tree count to 150 and above as seen in Figure 39, as the blue lines indicated. Refer to subsection 4.4.1 for the target model performance metrics.

In the final analysis, we presented the best ten substitute models and the mean values of 100 random parameter trials in scenario A sorted with F-score built with random parameters in Figure 40, Figure 41, Figure 42.

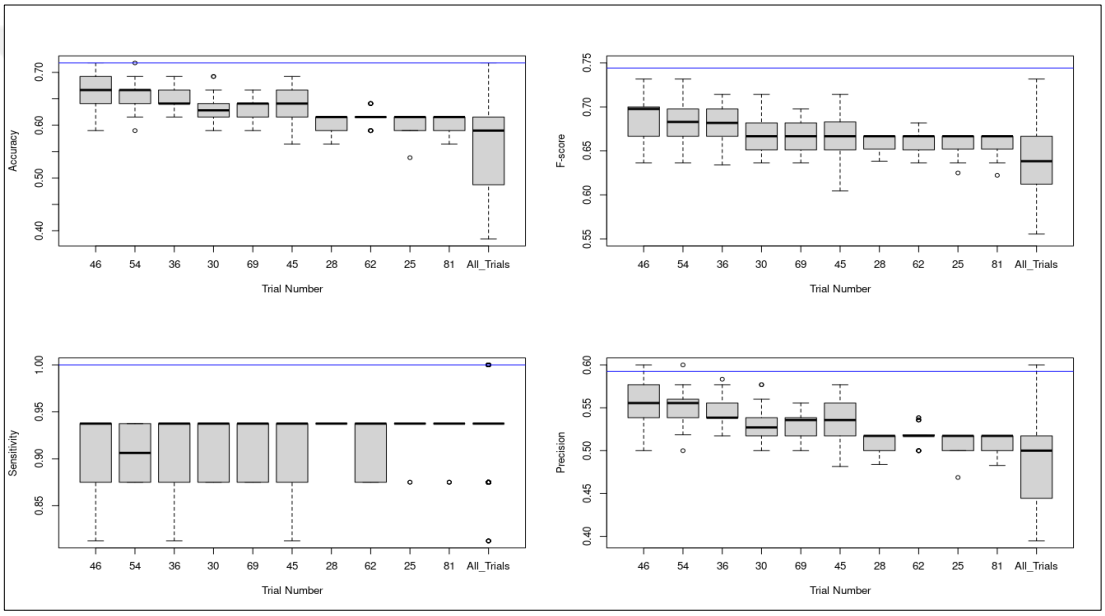

Figure 40: Performance metrics of the best ten substitute models with random parameters and 30 queried data – scenario A

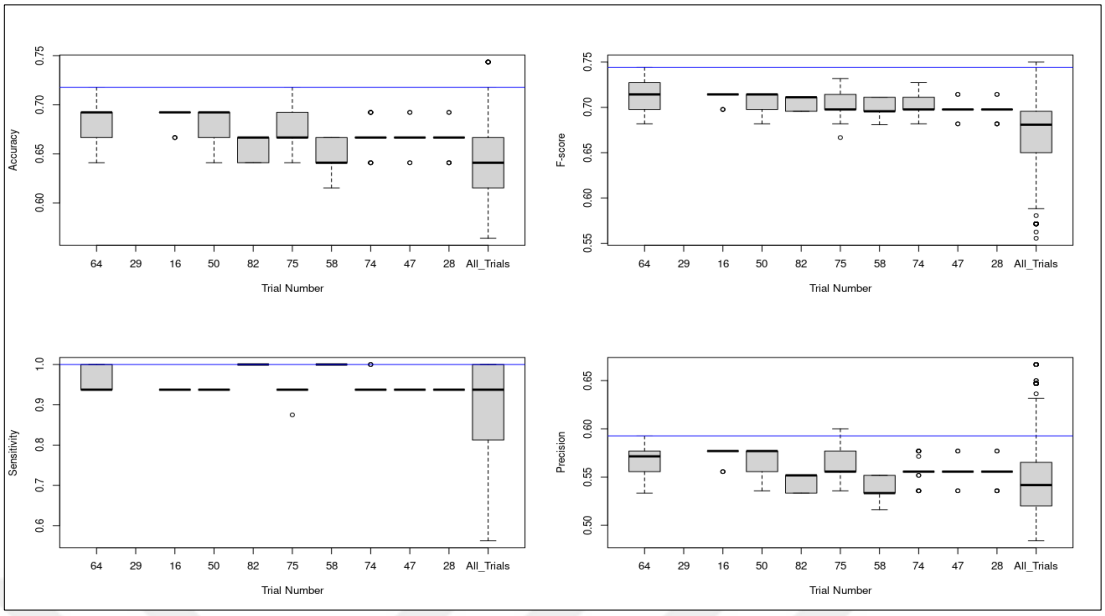

Figure 41: Performance metrics of the best ten substitute models with random parameters and 40 queried data – scenario A

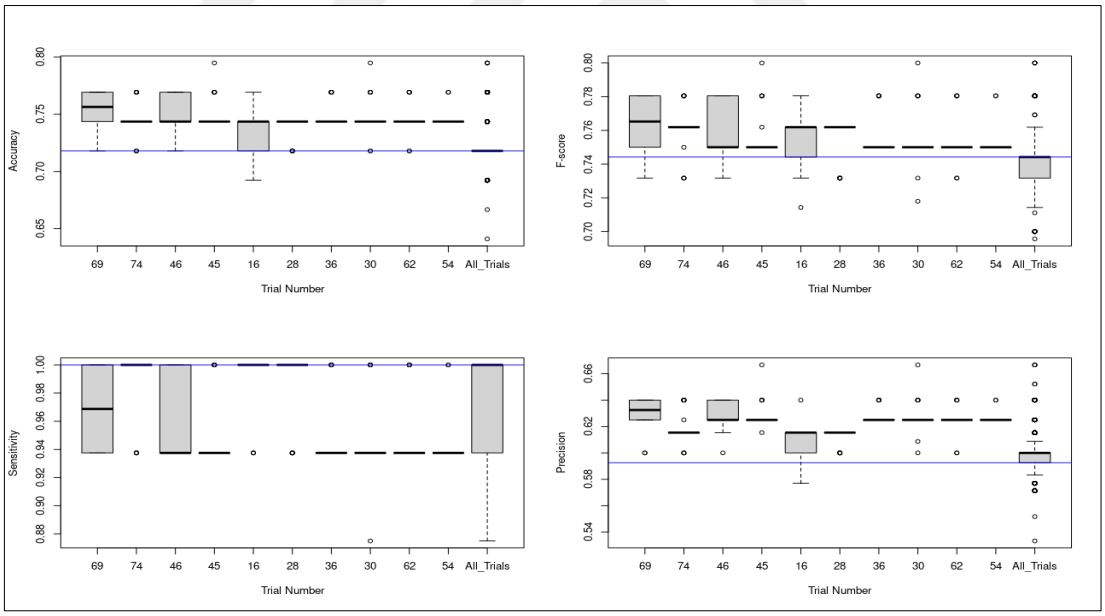

Figure 42: Performance metrics of the best ten substitute models with random parameters and 50 queried data – scenario A

The best ten substitute models built with a 30-query data set achieved mean accuracies of 0.60 to 0.65, whereas the F-score ranged between 0.66 and 0.69. Even though the sensitivity ranged between 0.91 and 0.94, the precision values remained

low, ranging from 0.51 to 0.55. The parameters of the models were presented in (Appendix 2, Table 31). We observed that none of the performance metrics hit the target model performance with 30-query data, as indicated with blue lines.

Second, the best ten substitute models built with a 40-query-data set achieved mean accuracies of 0.65 to 0.69, whereas the F-score ranged between 0.70 to 0.71. Even though the sensitivity ranged between 0.94 and 1, the precision values remained low, ranging from 0.54 to 0.58. The parameters of the models were presented in (Appendix 2, Table 32). Compared to the 30-query dataset, all metrics were increased, even though performance metrics of the target model were still not achieved except sensitivity, which was indicated by blue lines. Furthermore, we rejected the null hypothesis that the F-score of the substitute models was greater than or equal to the target model's F-score with a p-value of 0.

Third, the best ten substitute models built with a 50-query dataset achieved mean accuracies of 0.65 to 0.69, whereas the F-score ranged between 0.74 and 0.75. The sensitivity ranged between 0.94 and 0.99; the precision ranged between 0.61 and 0.63. The parameters of the models were presented in (Appendix 2, Table 33). Compared to the 40-query dataset, all metrics were increased and exceeded the prediction performance of our target model.

Next, we presented the performance metrics of the best ten substitute models and the mean values of 100 random parameter trials in scenario B sorted with F-score built with random parameters in Figure 43, Figure 44, and Figure 45.

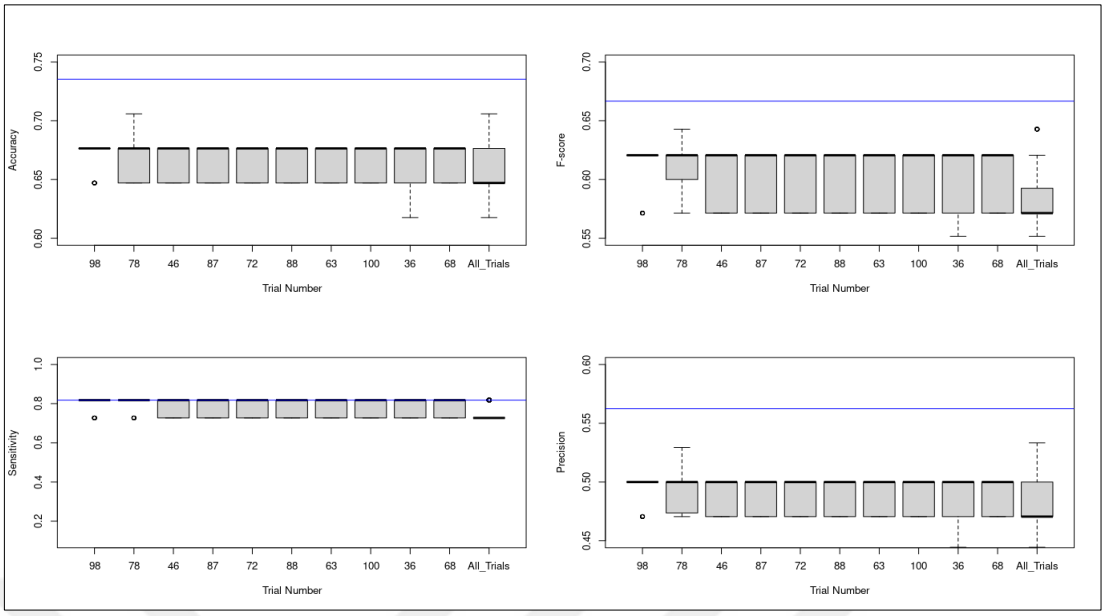

Figure 43: Performance metrics of the best ten substitute models with random parameters and 30 queried data – scenario B

The best ten substitute models built with a 30-query data set achieved mean accuracies of 0.66 to 0.67, whereas the F-score ranged between 0.60 and 0.61. Even though the sensitivity ranged between 0.78 and 0.80, the precision values remained low at 0.49. This analysis had a similar outcome to scenario B, with slightly better accuracy and a worse F-score. The parameters of the models were presented in (Appendix 2, Table 34). We observed that none of the performance metrics, except the sensitivity, hit the target model performance with 30-query data, as indicated by the blue lines.

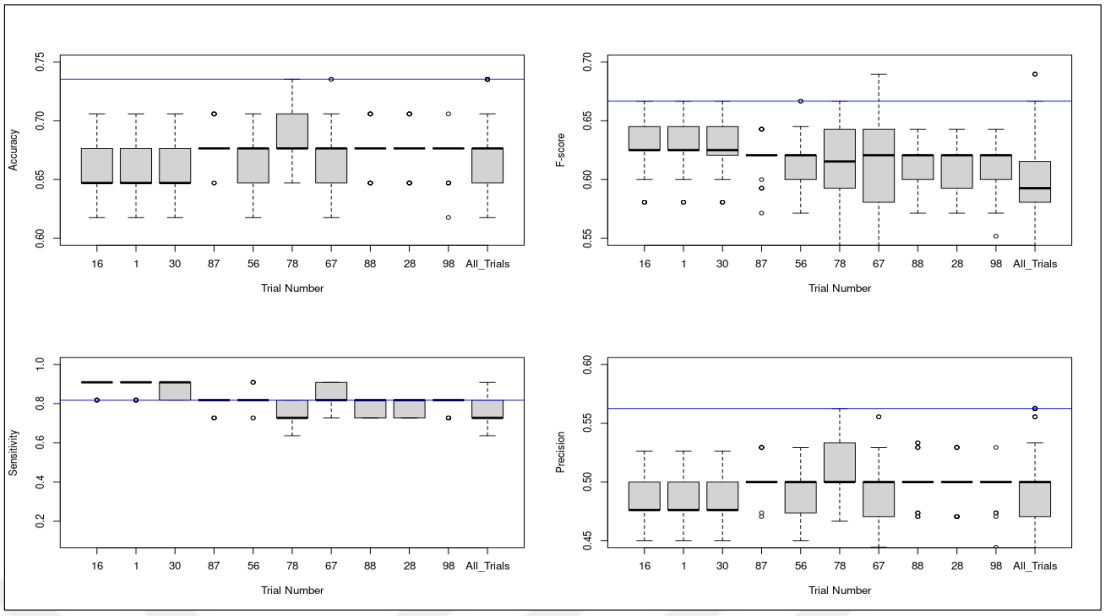

Figure 44: Performance metrics of the best ten substitute models with random parameters and 40 queried data – scenario B

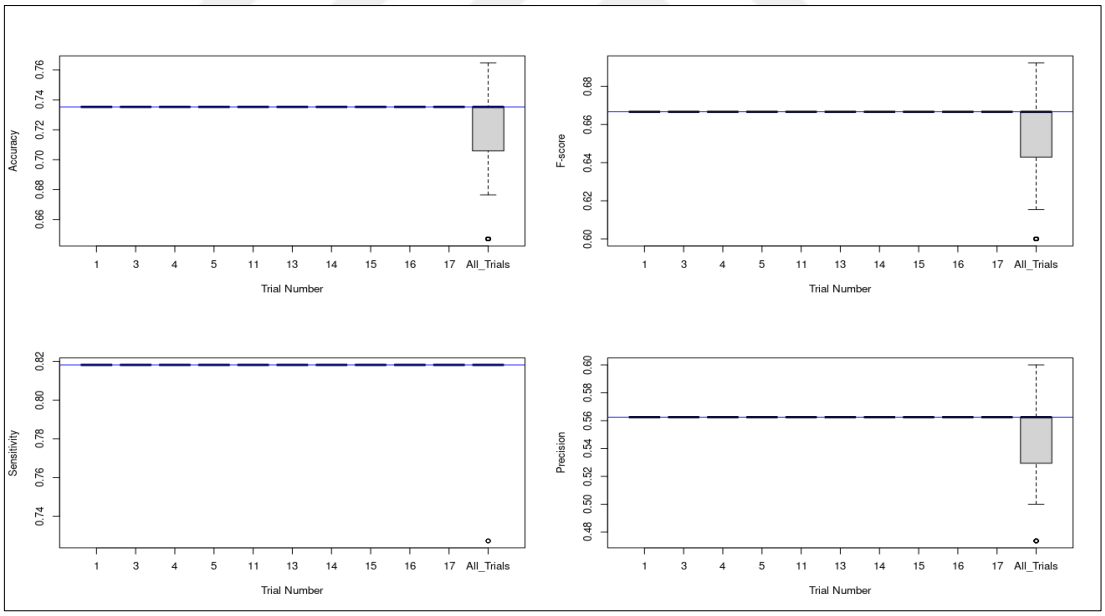

Figure 45: Performance metrics of the best ten substitute models with random parameters and 50 queried data – scenario B

Second, the best ten substitute models built with a 40-query-data set achieved mean accuracies of 0.66 to 0.69, whereas the F-score ranged between 0.61 to 0.63. Even though the sensitivity ranged between 0.76 and 0.89, the precision values

remained low, ranging from 0.49 to 0.51. The parameters of the models were presented in (Appendix 3, Table 35). All metrics were increased compared to the 30-query dataset, even though the target model's performance metrics were still not achieved except sensitivity, which was indicated by blue lines. Furthermore, we rejected the null hypothesis that the substitute models' F-scores were greater than or equal to the target model's F-score with a p-value of 0.

Third, the best ten substitute models built with a 50-query data set achieved all the performance metrics of the target model, as indicated with blue lines. The parameters of the models were presented in (Appendix 2, Table 36).

4.4.4 File sizes and execution times of the PPML inference

We evaluated the differences in prediction outcomes of PPML inferences and their public model counterparts. To verify that the prediction outcomes of both the public and the privacy-preserving RF models were the same, we utilized the training datasets from the substitute models, which contained 105 and 110 patients. In both scenarios, the predictions from the privacy-preserving RF models were the same as those from the original models.

In scenario A, the average time to prepare a single encoded query was recorded at 4.21 seconds, with a standard deviation of 0.13 seconds. The average duration for encryption, which includes key generation, measured at 9.58 seconds with a standard deviation of 0.09 seconds. This encryption process was followed by three calculation steps before the final output: node comparison value calculation, tree score calculation, and final masking of the value. The average duration for these steps was 0.37 seconds, 0.26 seconds, and 0.12 seconds, with standard deviations of 0.02, 0.02, and 0.01, respectively. The average duration for decryption in scenario A was 0.03 [0.001] seconds, and the total end-to-end duration for the entire process averaged 14.58 [0.17] seconds.

In scenario B, the average time needed to prepare a single encoded query was 4.20 seconds [0.12]. The average duration for the encryption process, including key generation, was 9.58 seconds [0.06]. The average times for node comparison calculation, tree score calculation, and final masking were 0.37, 0.26, and 0.12 seconds with standard deviations of 0.01, 0.02, and 0.01, respectively. The average decryption time was 0.03 [0.001] seconds, with the total end-to-end process duration averaging 9.73 [0.09] seconds. The file sizes for the encoded input and model outputs were consistent across both scenarios, measuring 2.506 MB for the input and 291 kB for the output for each query.

Compared to the XGBoost scenario attack (30), the file input size increase was proportioned to the increase in split points and the decrease in the number of variables. In contrast, the output file size did not change since we did not change the ring dimensions. Furthermore, the increase in the timing results of homomorphic encryption was approximately doubled, which was at the expected levels discussed in Magara's thesis (50), as we increased the tree depth, and the calculations needed to reach the leaves of each tree doubled.

5 DISCUSSION

5.1 Building Clinically Actionable Models for Predicting Mechanical Complications in Postoperatively Proportioned ASD Patients Using XGBoost Algorithm

Our joint initiative between clinicians and bioinformaticians has created a model that effectively evaluates clinical outcomes at an individual level while maintaining robust overall predictive performance. Developed using an expert-modified XGBoost algorithm and data from 244 patient records, the model demonstrates a satisfactory accuracy of 74% and an impressive sensitivity of 80%. Given our objective to accurately identify MCs, the achieved precision of 36% is deemed suitable for implementing preventive measures for at-risk patients.

Previously, machine learning has proven valuable in predicting complications related to adult spinal deformity. For example, Yagi et al. (85) documented a 92% accuracy in predicting significant complications within two years in a study of 195 patients. Similarly, Scheer et al. (86) reported an 86% accuracy in identifying a particular MC. Additionally, models derived from combined data from two study groups have effectively stratified risk for complications and readmissions, achieving AUC values between 0.67 and 0.92, comparable to our study outcomes.

Our study identifies critical predictors for MCs in patients with optimal post-operative alignment. These include the specific positioning of the lower-instrumented vertebra, the volume of estimated blood loss, pre-surgery ODI walking scores, patient age, and BMI. The interaction between the number of surgical levels fused and the construct location, as well as the length of follow-up post-surgery, is also significant.

Research has consistently shown that the position of the lowest instrumented vertebra is crucial for predicting outcomes (82, 86, 87). Other important factors include the extent of estimated blood loss, and the number of spinal levels fused, which are linked to higher rates of complications (88). Age (14, 86, 87, 89) and BMI (90, 91)

also play significant roles. Our data's most influential patient-reported measure was the ODI Walking score, consistent with previous literature (87). More extended follow-up periods significantly enhance the ability to predict complications, a finding supported by the longitudinal study results from Gupta et al. (92).

Contrasting with the initial model, which consisted of 5 trees and 23 leaves using 14 features, the refined expert-guided model is more streamlined, incorporating just four trees with 16 leaves and focusing on seven key features. This refinement simplifies the model and focuses on the most actionable and essential data, transforming the predictive process from an opaque algorithm to a transparent, interpretable tool that integrates seamlessly into clinical practice.

In choosing the appropriate algorithm, we avoided regression due to its vulnerability to multicollinearity (93). We ruled out algorithms compromised by large input spaces, such as k-nearest neighbors, which risk becoming ineffectual with increased feature counts (94). While principal component analysis was considered for reducing dimensionality (95), it would compromise the model's interpretability, which is crucial for clinical application. Instead, we opted for ensemble tree models known for their efficacy in smaller datasets and their ability to enhance predictive accuracy by boosting weak learners (96, 97, 98).

Our hyperparameter tuning strategy was meticulously designed to keep the model straightforward and clinically relevant without excessive complexity. We intentionally limited the number of trees and avoided extensive subsampling to ensure that each model component contributed meaningfully without overcomplicating the analysis.

We carefully balanced sensitivity and precision within the model to ensure it effectively identifies at-risk patients without overwhelming false positives. This sensitivity-focused approach, supported by the model's F score settings, allows for a detailed examination of predictive outcomes, facilitating more straightforward interpretation and clinical application.

The retrospective nature of our study and the relatively small size of our patient cohort are limitations. However, the dataset is one of the largest and most comprehensive for this condition. To enhance the generalizability and validity of our findings, further studies with more extensive and diverse patient populations are necessary. Moreover, the deliberate exclusion of bagging techniques that might increase accuracy was chosen to maintain the model's practicality and interpretability in clinical settings (99).

5.2 Predicting Mechanical Complications in Postoperatively Proportioned and Moderately Disproportioned ASD Patients Using RF Algorithm

In our research, we utilized a training dataset of 295 individual health records to develop a random forest model with 500 trees and a set of 8 rules. Our random forest model achieved 72% accuracy and 91% sensitivity, with a precision of 55%, based on tests conducted with 98 patient records. Conversely, the rule set recorded 74% accuracy and 81% sensitivity, with a precision of 58%. Despite the relatively low precision for both models, this was considered acceptable because our study's main objective was accurately identifying patients at risk of MCs. We prioritized sensitivity over precision by using a class weighting of 2:1 and selecting F-score as our performance metric. Consequently, using these models to guide preventive treatments could lead to unnecessary interventions. This makes the application of costly preventive treatments impractical.

In our random forest model, the most effective features for predicting MCs were the location of the lowest instrumented vertebra (LIV) during surgery, post-operation GAP score, BMI, age, and pre-operation apical translation to CSVL, listed in order of significance. When comparing the key features of the random forest model and the rule set based on the importance of the permutation feature, we observed that the rule set's performance depended on just four to six features, simplifying its complexity. In addition to LIV location and post-operation GAP score, features such as osteotomy, pre-operation T2 T5 Kyphosis, and follow-up time were crucial for the rule set's effectiveness. Furthermore, all features utilized in the rule set were deemed significant

in the permutation importance analysis, or an associated feature was significant, except the accessory linked versus satellite unlinked categorical variable. Although this variable did not statistically predict MCs, it enhanced the rule set's predictive capability. The previous discussion subsection explained the importance of such variables in the literature. We did not use permutation feature importance analysis in the previous subsection as the ensembles had only a few trees, and default feature importance was adequate for our analysis.

While the retrospective nature of this study might have resulted in the omission of key data on MCs, it is important to recognize that these limitations are typical of retrospective research. Although the size of our cohort limited our ability to explore specific conditions, this does not undermine the study's value, given that the ESSG dataset is among the largest and most comprehensive datasets for ASD. However, the reliance on data from only six sites might introduce biases that could potentially influence the broader applicability of our findings.

5.3 Tree-based Modeling for Interpretability

In this study, we opted to use tree-based modeling techniques rather than neural networks to align with the clinicians' expressed preference for interpretable models. While neural networks have demonstrated remarkable predictive performance across a variety of clinical tasks, their inherent complexity and "black-box" nature often limit their practical adoption in healthcare settings where transparency and explainability are paramount.

Tree-based models, such as decision trees, random forests, and gradient boosting machines, offer a more transparent approach to modeling by providing clear, rule-based structures that can be easily visualized and understood. This interpretability is crucial for clinicians who need to trust and validate the model's predictions before integrating them into clinical decision-making. By using tree-based models, we ensured that the rationale behind each prediction could be traced back to specific

features and decision rules, facilitating clinical insight and fostering confidence in the model's outputs.

Moreover, the interpretability of tree-based models supports regulatory compliance and ethical considerations, as it enables the identification of potential biases and errors in the decision process. This transparency is essential in clinical environments where patient safety and accountability are critical. Although neural networks may offer marginally higher accuracy in some cases, the trade-off between performance and interpretability guided our choice, prioritizing models that clinicians can readily understand and trust.

5.4 The Security Analysis of Privacy-Preserving XGBoost Inference

Our research focused on evaluating the commercial viability of the ESSG's modeling work. We aimed to determine a suitable query threshold for additional ASD surgery sites interested in utilizing the model while maintaining the security of the model and the data queries. In our implementation, we used the CKKS scheme, which adheres to the IND-CPA standard, thereby protecting the privacy of the input data. Nonetheless, each query and the inherent nature of the encoding method inadvertently revealed some details about the model via the results.

Recent advancements in algorithm development have focused on 'extracting' models using query data, particularly targeting complex models like neural networks. Examples include the Copycat CNN (100) and Knockoff Nets (101), which are both designed to build neural networks. In our research, we exclusively used the naive XGBoost algorithm to conduct attacks, attempting to replicate the less complex target ensembles in the attack phase.

We identified two vulnerabilities in our study that we wish to discuss. Firstly, although we split the attacker data before constructing a model to be attacked (target), the seven variables used in the modeling phase were derived from the dataset described in Balaban et al.'s work (28). This approach might have increased the likelihood of a

successful attack, as the attacker data contributed to the variable selection method of the target model-building phase. Secondly, we were compelled to include synthetic data in the validation sets for parts two to four of each scenario, in addition to the original dataset. This inclusion may have led to a signal loss when comparing parts 3 and 4 in both scenarios. The average number of variables used by target and attacker models decreased from 4.47 to 4.06 and from 4.76 to 4.56 in scenarios A and B, parts 3 and 4, respectively. In Figure 16, we observed some lost signals, although a trend was visible where the use of the target model's split points increased as the query size was augmented with synthetic data.

Given that the HEAAN library relies on approximation techniques, we initially expected potential variations in accuracy between the PPML and public models. Nonetheless, our ensembles were relatively shallow; ultimately, we observed no differences. The homoscedastic two-sample t-test used to compare timing results across both scenarios did not show significant differences in overall timing means (p-value 0.9). However, there was a notable difference in the average encryption times (p-value 0.004), with encryption taking about 0.2 seconds longer on average in scenario A. This variation could be attributed to an additional tree in the ensemble. As suggested in Magara et al.'s work (49), further analysis is required to clarify the cause of this variation. One potential solution could be standardizing the number of trees by including dummy trees until reaching a pre-set maximum count.

For patients aligned proportionally according to the GAP scoring system, proactively detecting MCs is particularly important. Therefore, we developed highly sensitive models, resulting in low-precision target models. This method reduced the ratio of healthy patients to patients with MCs from 4.9 in the complete dataset to 2.1 for scenario A and 1 for scenario B in the original attacker training datasets. Introducing synthetic data to the queries altered this ratio in favor of more MC predictions, reflected in the reduction of the scale positive weight parameter in Table 5, which denotes the ratio. Despite attacker cv mean precisions ranging from 65% to 97%, the attacker models could not replicate these precisions in the test sets due to this ratio adjustment in scenarios parts 2-4. Although adding more synthetic data did not

enhance the performance metrics of the attacks, logically, more queries led to more significant information leakage, as demonstrated in our analysis of the gain values in the attacker models.

We noted a significant difference between parts 2 and 3 in both scenarios regarding the average target model variables used in attacker models, which increased from 62% to 89% in scenario A and 41% to 79% in scenario B, respectively. Using the results from part 3, attackers might develop new data generation methods by modifying those variables, potentially increasing the information leakage from the queried models. The heat maps in Figure 17 support our observation as they show the correct usage of split points when 150 synthetic data points are added to the original dataset.

At last, we concluded that allowing 100 queries would be safe based on the evaluations discussed. In the second part of both scenarios, queries exceeded 100, where 50 synthetic data points were combined with the original attacker dataset. In scenario A, the ensemble was queried 103 times, and 116 times in scenario B. Given that the participating sites had a maximum of 70 patients over ten years, the query limit protects the model for 14 years based on the data from the mentioned site, which is appropriate. Furthermore, the timings and file sizes for a single query to the privacy-preserving model inference were reasonably suitable for practical use.

5.5 The Security Analysis of Privacy-Preserving RF Inference

Our research focused on assessing the feasibility of commercializing the ESSG's modeling initiatives. We endeavored to determine a suitable query limit for other ASD surgery locations interested in leveraging the model while maintaining the security of both the model and the data queried. The CKKS scheme, known for its IND-CPA level security, was utilized in our implementation to ensure the privacy of the input data. Nonetheless, every query and the encoding technique unintentionally leaked some details about the model through the outcomes of the queries.

Recent advancements in various algorithms have targeted the extraction of model data using query inputs, predominantly focusing on intricate models like neural networks. Prominent methods such as Copycat CNN (100) and Knockoff Nets (101) are designed to create neural networks. In contrast, our study exclusively utilized a straightforward approach, employing the RF algorithm to simulate attacks, aiming to emulate the target ensembles during the attack stages.

The vulnerability in our research was not using CV techniques when we built the target and substitute models. The reason we did not use such techniques was that even with using default metrics, substitute models were able to achieve performance scores at least as good as the target models. Furthermore, to find an acceptable threshold for queries, we conducted a random parameter analysis instead of tuning parameters with cross-validation to have a broader view of what substitute models can achieve in performance.

Given that the HEAAN library relies on approximation techniques, we initially expected potential variations in accuracy between the PPML and public models. Nonetheless, in our ensembles, we observed no differences. The homoscedastic two-sample t-test used to compare timing results across both scenarios did not show significant differences in timing results, with a p-value of 0.95.

Compared to the security analysis results of clinically actionable privacy-preserving XGBoost models, we concluded that a lesser query limit was fit with only 40 patient records. Given that the participating sites had a maximum of 149 patients over 10 years, the query limit protects the model for 2.7 years based on the data from the mentioned site is appropriate. Given that continual model updating had leverage in healthcare (102, 103, 104), the 2.7-year duration to ensure model privacy was enough to update the model. On the other hand, LIV location was found important by substitute models in both scenarios, and in fact, it was an important variable in both models. This was because it was the most important variable found in the permutation variable importance in the RF modeling section.

Also, the timings and file sizes for a single query to the privacy-preserving RF model inference were reasonably suitable for practical use. However, the timings can be improved further by changing the encoding algorithm as it uses if statements. However, it can also be used with arrays, as the only timing result increased due to the number of split points. Furthermore, as explained in Magara's thesis (50), the number of trees only indirectly affects homomorphic calculation times, only increasing by 14% due to the baching effect. Using 500 trees instead of small-sized trees, the number of split points increased. Hence, the algorithm had to increase homomorphic multiplications to compute the output, as discussed in Magara's thesis (50).

Lastly, we conducted different methodologies to assess the substitute models compared to XGBoost attacks because of the differences in the number of trees of the target models. XGBoost models were clinically actionable by design, thus resulting in a small number of trees.

5.6 Differences in Query Limits of PPML Inferences

There might be two explanations for the differences in query limit in our PPML inferences. For XGBoost modeling, we also administered expert-driven methodologies, where we actively removed the splits that did not have clinical explanations. Besides, we only modeled one subgroup of ASD patients with XGBoost, whereas we modeled two subgroups using the RF algorithm. Furthermore, we built models with fewer trees compared to random forest models. Next, XGBoost models only modeled on one subgroup of ASD patients, whereas RF models used two of the three GAP score subgroups. Due to these reasons, XGBoost target models were secure with a higher query limit as they had more unique signals that attacker data could not learn. On the other hand, using 500 trees in RF modeling balanced out the weaker signals and produced fewer but stronger signals that substitute models could take advantage of.

5.7 Our Contribution to PPML Field

Our study significantly extends the work of Magara et al. (50) by providing a comprehensive security analysis focused on privacy-preserving machine learning (PPML) in realistic adversarial settings. While Magara et al. demonstrated the effectiveness of the privacy-preserving XGBoost algorithm for predicting mechanical complications in ASD patients, our contribution lies in rigorously evaluating the robustness of privacy-preserving methods against substitute model building, including scenarios where adversaries act individually or collude. This nuanced adversarial modeling enables a deeper understanding of potential vulnerabilities that were not fully addressed in previous work.

Furthermore, we advance the practical application of cryptographic protections by implementing homomorphic encryption using open-source libraries, thereby demonstrating the feasibility of deploying such advanced techniques in real-world healthcare environments. This implementation bridges the gap between theoretical cryptographic solutions and their operational use, confirming that sensitive patient data can be secured without sacrificing model utility or scalability.

Another critical insight from our analysis is the challenge of protecting inference models that are inherently simple due to explainability requirements. Unlike more complex models that might obscure internal workings, simpler models are more vulnerable to extraction and inversion attacks. To mitigate this, we propose a methodology for limiting the number of queries allowed to the model, establishing a threshold that balances privacy preservation with practical usability.

Overall, our contributions provide a robust framework for securing PPML systems in sensitive domains, ensuring patient data confidentiality and safeguarding intellectual property without compromising predictive performance. By combining thorough security analysis with practical cryptographic implementation, our work offers valuable guidance for future deployments of privacy-preserving machine

learning in healthcare and beyond, building on and extending the foundation laid by Magara et al.(50).

5.8 Preferring A Blockchain-less Solution

While renowned for its role in user-centric systems that safeguard private data, blockchain often offers solutions that do not directly address the primary challenges of specific applications, such as those in supply chain contexts where failures predominantly occur at the data collection phase—a problem blockchain does not solve. This mismatch of the solution to the problem was a key consideration, suggesting that blockchain might not offer tangible benefits for the core challenges in our PPML ASD data.

Blockchain technology thrives in a decentralized environment, eliminating the need for a central authority. However, in many real-world scenarios, such as health data management and processing, some authority and centralized control are often necessary to ensure data integrity, compliance with health regulations, and efficient system operation. The decentralized nature of blockchain could complicate the governance and operational control required in these contexts.

One of the significant barriers to implementing blockchain in our project was its limited feasibility in handling extensive data, notably encrypted health records like those in our study of ASD. Blockchain's capacity constraints become evident with larger datasets, which can be cumbersome and inefficient to manage on a public blockchain due to increased processing time and potential cost implications.

In summary, although blockchain presents innovative aspects of decentralization, it was deemed unsuitable for our PPML algorithm due to practical challenges in scalability, data handling, and integration with existing technological infrastructure. Thus, a trusted server that aligns with the operational, security, and regulatory needs of handling sensitive health data in our study was chosen as the more appropriate technology.

CONCLUSION

In this study, we developed and validated robust machine-learning models for predicting MCs in ASD patients. These models were built utilizing XGBoost and Random Forest algorithms, enhancing prediction accuracy by integrating clinical insights into their training processes. We focused on two primary patient groups: well-aligned post-operatively and those moderately disproportioned. Our results underscored the effectiveness of these predictive models in clinical settings, where they achieved high sensitivity and specificity, aiding surgeons in planning and executing spinal surgeries with reduced complication risks.

We implemented PPML techniques to secure both the model and the data. The models operated under encryption, ensuring that patient data remained confidential and the model's integrity was protected against unauthorized access. This was particularly crucial given the sensitive nature of the healthcare data involved. By leveraging homomorphic encryption, we ensured that computations could be carried out on encrypted data, maintaining data privacy throughout the process.

Moreover, we explored the potential of blockchain technology to create a transparent, secure framework for sharing and utilizing machine learning models. Even though this technology offered a decentralized approach to managing data transactions and a verifiable and permanent record of data transactions, encrypted file sizes are the biggest problem to overcome, which deviated our direction to use a homomorphic encryption algorithm on a trusted server without losing any privacy.

Future research will focus on several key areas to further enhance the models and their applications. First, we plan to expand our dataset to include a broader demographic to improve the generalizability of the models. By incorporating data from a wider range of patient backgrounds and medical conditions, the models can be trained to predict outcomes more accurately across diverse patient populations.

Another area of future work will involve continuously improving the privacy-preserving techniques employed in this study. While the current implementation

provides a robust framework for protecting data privacy, evolving digital threats and advancing technology will necessitate ongoing enhancements to these security measures. The research will be directed toward developing more efficient cryptographic methods and exploring newer blockchain technologies that may offer improved performance.

We also aim to refine the integration of clinical expertise into the model training process. This will involve deeper collaboration with healthcare professionals to tailor the machine learning algorithms more closely to real-world clinical needs. By enhancing the interpretability of the models, clinicians can gain better insights into the predictive factors and potentially adjust treatment plans more effectively based on model outputs.

Moreover, the practical deployment of these models in clinical settings will be explored. This includes real-time application in surgical planning and post-operative care, where the models can provide dynamic predictions based on the latest patient data. Testing in live clinical environments will help identify practical challenges and opportunities for further refinement.

In conclusion, the research presented has laid a strong foundation for using advanced machine-learning techniques to predict MCs in ASD patients. Through rigorous testing and enhancement, these models have the potential to significantly impact patient care by reducing the risk of complications and improving surgical outcomes. As we move forward, our efforts will continue to focus on improving the accuracy, usability, and security of these models to serve better the needs of the medical community and its patients.

6 REFERENCES

1. Turing AM. Computing machinery and intelligence. Parsing the Turing Test. 2007 Nov 23;23–65. doi:10.1007/978-1-4020-6710-5_3
2. Jordan MI, Mitchell TM. Machine learning: Trends, Perspectives, and prospects. Science. 2015 Jul 17;349(6245):255–60. doi:10.1126/science.aaa8415
3. Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, et al. Dermatologist-level classification of skin cancer with Deep Neural Networks. Nature. 2017 Jan 25;542(7639):115–8. doi:10.1038/nature21056
4. Shourie P, Anand V, Gupta S. An efficient CNN framework for radiologist level pneumonia detection using chest X-ray images. 2023 3rd International Conference on Intelligent Technologies (CONIT). 2023 Jun 23. doi:10.1109/conit59222.2023.10205557
5. Tang J, Liu R, Zhang Y-L, Liu M-Z, Hu Y-F, Shao M-J, et al. Application of machine-learning models to predict tacrolimus stable dose in renal transplant recipients. Scientific Reports. 2017 Feb 8;7(1). doi:10.1038/srep42192
6. Rao SR, DesRoches CM, Donelan K, Campbell EG, Miralles PD, Jha AK. Electronic health records in small physician practices: Availability, use, and perceived benefits. Journal of the American Medical Informatics Association. 2011 May;18(3):271–5. doi:10.1136/amiajnl-2010-000010
7. Habehh H, Gohel S. Machine learning in Healthcare. Current Genomics. 2021 Dec 16;22(4):291–300. doi:10.2174/1389202922666210705124359
8. North America Spinal Surgery Market - size, Growth & statistics, Forecast [Internet]. [cited 2022 May 20]. Available from: https://www.mordorintelligence.com/industry-reports/north-america-spinal-surgery-market-industry
9. Back surgery - types & recovery: Made for this moment [Internet]. [cited 2022 May 20]. Available from: https://www.asahq.org/madeforthismoment/preparing-for-surgery/procedures/back-surgery
10. Daniell JR, Osti OL. Failed back surgery syndrome: A review article. Asian Spine Journal. 2018 Apr 30;12(2):372–9. doi:10.4184/asj.2018.12.2.372
11. Nachemson AL. Evaluation of results in Lumbar spine surgery. Acta Orthopaedica Scandinavica. 1993 Jan;64(sup251):130–3. doi:10.3109/17453679309160143
12. Sebaaly A, Gehrchen M, Silvestre C, Kharrat K, Bari TJ, Kreichati G, et al. Mechanical complications in adult spinal deformity and the effect of restoring the spinal shapes according to the Roussouly classification: A multicentric study. European Spine Journal. 2019 Dec 26;29(4):904–13. doi:10.1007/s00586-019-06253-1
13. Schwab F, Ungar B, Blondel B, Buchowski J, Coe J, Deinlein D, et al. Scoliosis Research Society—schwab adult spinal deformity classification. Spine. 2012 May;37(12):1077–82. doi:10.1097/brs.0b013e31823e15e2

14. Yilgor C, Sogunmez N, Yavuz Y, Boissiere L, Obeid I, Acaroglu E, et al. Global alignment and proportion (GAP) score: Development and validation of a new method of analyzing Spinopelvic alignment to predict mechanical complications after adult spinal deformity surgery. The Spine Journal. 2017 Oct;17(10). doi:10.1016/j.spinee.2017.07.234
15. Bitcoin: A Peer-to-Peer Electronic Cash System [Internet]. [cited 2024 May 20]. Available from: https://bitcoin.org/bitcoin.pdf
16. Akhai S. Healthcare record management for Healthcare 4.0 via Blockchain: A review of current applications, opportunities, challenges, and future potential. Blockchain for Healthcare 40. 2023 Nov 13;211–23. doi:10.1201/9781003408246-11
17. Ethereum: A secure decentralized generalised transaction ledger [Internet]. [cited 2024 May 20]. Available from: https://ethereum.org/en/whitepaper/
18. Atzei N, Bartoletti M, Cimoli T. A survey of attacks on Ethereum Smart Contracts (SOK). Lecture Notes in Computer Science. 2017;164–86. doi:10.1007/978-3-662-54455-6_8
19. Augot D, Chabanne H, Chenevier T, George W, Lambert L. A user-centric system for verified identities on the bitcoin blockchain. Lecture Notes in Computer Science. 2017;390–407. doi:10.1007/978-3-319-67816-0_22
20. Yan B, Chen P, Li X, Wang Y. Nebula: A blockchain based decentralized sharing computing platform. Communications in Computer and Information Science. 2019 Dec 23;715–31. doi:10.1007/978-981-15-2777-7_58
21. Verhoeven P, Sinn F, Herden T. Examples from blockchain implementations in logistics and supply chain management: Exploring the mindful use of a new technology. Logistics. 2018 Sept 11;2(3):20. doi:10.3390/logistics2030020
22. Siqueira A, Conceição AF, Rocha V. User-centric health data using self-sovereign identities. Anais do IV Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain 2021). 2021 Aug 16. doi:10.5753/wblockchain.2021.17135
23. Hawashin D, Jayaraman R, Salah K, Yaqoob I, Simsekler MC, Ellahham S. Blockchain-based management for organ donation and transplantation. IEEE Access. 2022;10:59013–25. doi:10.1109/access.2022.3180008
24. Lauter K. Private AI: Machine learning on encrypted data. SEMA SIMAI Springer Series. 2022;97–113. doi:10.1007/978-3-030-86236-7_6
25. Tiwari K, Shukla S, George JP. A systematic review of challenges and techniques of privacy-preserving machine learning. Data Science and Security. 2021;19–41. doi:10.1007/978-981-16-4486-3_3
26. Breiman L. Random Forests. Machine Learning. 2001;45(1):5–32. doi:10.1023/a:1010933404324
27. Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016 Aug 13. doi:10.1145/2939672.2939785

28. Balaban B, Yilgor C, Yucekul A, Zulemyan T, Obeid I, Pizones J, et al. Building clinically actionable models for predicting mechanical complications in postoperatively well-aligned adult spinal deformity patients using XGBoost algorithm. Informatics in Medicine Unlocked. 2023;37:101191. doi:10.1016/j.imu.2023.101191
29. Ping H, Stoyanovich J, Howe B. Datasynthesizer: Privacy-preserving synthetic datasets. In: Proceedings of the 29th International Conference on Scientific and Statistical Database Management. 2017 Jun 27.
30. Balaban B, Magara SS, Yilgor C, Yucekul A, Obeid I, Pizones J, et al., Privacy-Preserving Machine Learning (PPML) Inference for Clinically Actionable Models. IEEE Access. 2025. doi: 10.1109/ACCESS.2025.3540261.
31. Razi M, Athappilly K. A comparative predictive analysis of Neural Networks (NNS), nonlinear regression and classification and regression tree (CART) models. Expert Systems with Applications. 2005 Jul;29(1):65–74. doi:10.1016/j.eswa.2005.01.006
32. Grinsztajn L, Oyallon E, Varoquaux, G. Why do tree-based models still outperform deep learningon typical tabular data? 2022. Preprint at https://doi.org/10.48550/arXiv.2207.08815
33. Gardner MW, Dorling SR. Statistical Surface Ozone Models: An improved methodology to account for non-linear behaviour. Atmospheric Environment. 2000 Jan;34(1):21–34. doi:10.1016/s1352-2310(99)00359-3
34. Song YY, Lu Y. Decision tree methods: applications for classification and prediction. Shanghai Arch Psychiatry. 2015 Apr 25;27(2):130-5. doi: 10.11919/j.issn.1002-0829.215044. PMID: 26120265; PMCID: PMC4466856.
35. Hastie TJ, Tibshirani RJ, Friedman JH. The Elements of Statistical Learning: Data Mining Inference and Prediction. Second Edition. Springer; 2009. ISBN 978-0-387-84857-0
36. Quinlan JR. Simplifying Decision Trees. International Journal of Man-Machine Studies. 198;27, 221-234.
37. Breiman L. Classification and regression trees. Routledge. 2017.
38. Hu L, Li L. Using tree-based machine learning for Health Studies: Literature Review and case series. International Journal of Environmental Research and Public Health. 2022 Dec 1;19(23):16080. doi:10.3390/ijerph192316080
39. Breiman L. Bagging predictors. Machine Learning. 1996;24(2):123–40. doi:10.1023/a:1018054314350
40. Probst, Philipp, and Boulesteix, Anne-Laure. To tune or not to tune the number of trees in random forest?. Journal of Machine Learning Research. 2017;18.
41. Dwork C. Differential Privacy: A survey of results. Lecture Notes in Computer Science. 2008;1–19. doi:10.1007/978-3-540-79228-4_1
42. Cramer R, Damgård IB. Secure multiparty computation. Cambridge University Press. 2015.
43. Shundong L, Daoshun W, Yiqi D, Ping L. Symmetric cryptographic solution to yao's millionaires' problem and an evaluation of secure multiparty computations. Information Sciences. 2008 Jan;178(1):244–55. doi:10.1016/j.ins.2007.07.015

44. Li L, Fan Y, Tse M, Lin K-Y. A review of applications in Federated Learning. Computers & Industrial Engineering. 2020 Nov;149:106854. doi:10.1016/j.cie.2020.106854
45. Rieke N, Hancox J, Li W, Milletarì F, Roth HR, Albarqouni S, et al. The future of digital health with federated learning. NPJ Digit Med. 2020 Sep 14;3:119. doi: 10.1038/s41746-020-00323-1. PMID: 33015372; PMCID: PMC7490367.
46. Menezes A. Another look at provable security. Advances in Cryptology – EUROCRYPT 2012. 2012;8–8. doi:10.1007/978-3-642-29011-4_2
47. Stern, J. Why provable security matters?. In: International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 449-461.
48. Meng X., Feigenbaum J. Privacy-preserving xgboost inference. 2020; arXiv preprint arXiv:2011.04789.
49. Mağara ŞS, Yıldırım F, Yaman B, Dilekoğlu FR, Tutaş E, Öztürk K, et al. Ml with HE: Privacy preserving machine learning inferences for genome studies, 21 October 2021. arXiv:2110.11446. https://doi.org/10.48550/arXiv.2110.11446
50. Magara, S.S.: Privacy-Preserving Xgboost Inference with Homomorphic Encryption. 2022. Master's thesis, Sabanci University.
51. Kiss Á, Naderpour M, Liu J, Asokan N, Schneider T. SoK: Modular and Efficient Private Decision Tree Evaluation. Proc Priv Enhancing Technol. 2019;2019(2):187–208.
52. Wang Q, et al. GTree: GPU-Friendly Privacy-preserving Decision Tree Training and Inference. arXiv preprint arXiv:2305.00645. 2023.
53. Truex S, Liu L, Gursoy ME, Yu L. Privacy-Preserving Inductive Learning with Decision Trees. 2020.
54. Chatel S, Pyrgelis A, Troncoso-Pastoriza JR, Hubaux JP. SoK: Privacy-Preserving Collaborative Tree-based Model Learning. Proc Priv Enhancing Technol. 2021;2021(1):222–243.
55. Yuan S, Li H, Qian X, Jiang W, Xu G. OnePath: Efficient and Privacy-Preserving Decision Tree Inference in the Cloud. OpenReview. 2024.
56. Frery J, Stoian A, Bredehoft R, Montero L, Kherfallah C, Chevallier-Mames B, Meyre A. Privacy-Preserving Tree-Based Inference with Fully Homomorphic Encryption. Cryptology ePrint Archive, Report 2023/258. 2023.
57. Gentry C. Fully homomorphic encryption using ideal lattices. Proceedings of the forty-first annual ACM symposium on Theory of computing. 2009 May 31. doi:10.1145/1536414.1536440
58. Zhou X, Tang X. Research and implementation of RSA algorithm for encryption and decryption. Proceedings of 2011 6th International Forum on Strategic Technology. 2011 Aug. doi:10.1109/ifost.2011.6021216
59. Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory, 1985, 31.4: 469-472.

60. Gathen J von zur. Cryptoschool. Berlin, Heidelberg: Springer Berlin Heidelberg; 2015.
61. Chase M, Chen H, Ding J, Goldwasser S, Gorbunov S, Hoffstein J, et al. Security of Homomorphic Encryption. [Internet]. [cited 2023 August 15]. Available from: https://www.microsoft.com/en-us/research/wp-content/uploads/2018/01/security_homomorphic_encryption_white_paper.pdf
62. Albrecht M, Chase M, Chen H, Ding J, Goldwasser S, Gorbunov S, et al. Homomorphic encryption standard. Protecting privacy through homomorphic encryption. 2021, 31-62.
63. Acar A, Aksu H, Uluagac AS, Conti M. A survey on homomorphic encryption schemes. ACM Computing Surveys. 2018 Jul 25;51(4):1–35. doi:10.1145/3214303
64. Ameur Y, Bouzefrane S, Audigier V. Application of Homomorphic Encryption in Machine Learning. Emerging Trends in Cybersecurity Applications. 2022 Jul 6;391–410.
65. Palisade Lattice Cryptography Library, Version 1.11.2. 2021. https://palisade-crypto.org/
66. Benesty J, Chen J, Huang Y, Cohen I. Pearson correlation coefficient. In: Noise reduction in speech processing. Springer; 2009. p. 37–40.
67. R Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. 2024.
68. Kuhn M. Building predictive models in R using the caret package. Journal of Statistical Software. 2008;28(5). doi:10.18637/jss.v028.i05
69. Wickham H, François R, Henry L, Müller K. dplyr: A grammar of data manipulation. 2022. https://dplyr.tidyverse.org. https://github.com/tidyverse/dplyr.
70. Liaw A, Wiener M. Classification and Regression by randomForest. R News 2(3). 2002;18-22.
71. Friedman JH, Popescu BE. Predictive learning via rule ensembles. The Annals of Applied Statistics. 2008 Sep;2(3):916–54. https://doi.org/10.1214/07-aoas148
72. Liu B, Hsu W, Ma Y. Integrating classification and association rule mining. Proceeding of the 1998 International Conference on Knowledge Discovery and Data Mining, ACM, 1998, pp. 80–86.
73. Deng H, Runger G, Tuv E, Bannister W. CBC: An associative classifier with a small number of rules. Decision Support Systems. 2014 Mar;59:163–70. https://doi.org/10.1016/j.dss.2013.11.004
74. Deng H. Interpreting tree ensembles with inTrees. International Journal of Data Science and Analytics. 2018 Jul 11;7(4):277–87. https://doi.org/10.1007/s41060-018-0144-8
75. Deng H, Runger G. Gene selection with guided regularized random forest. Pattern Recognition. 2013 Dec;46(12):3483–9. https://doi.org/10.1016/j.patcog.2013.05.018
76. Probst P, Boulesteix A-L. To tune or not to tune the number of trees in random forest?. Journal of Machine Learning Research. 2017;18.
77. Altmann A, Toloşi L, Sander O, Lengauer T. Permutation importance: a corrected feature importance measure. Bioinformatics. 2010;26(10):1340–7.

78. Müllner D. fastcluster: Fast Hierarchical, Agglomerative Clustering Routines forRandPython. Journal of Statistical Software [Internet]. 2013 [cited 2024 Mar 28];53(9). Available from: https://arxiv.org/pdf/1109.2378.pdf
79. Curtmola R, Garay J, Kamara S, Ostrovsky R. Searchable symmetric encryption. Proceedings of the 13th ACM conference on Computer and communications security - CCS '06. 2006.
80. Cheon JH, Kim A, Kim M, Song Y. Homomorphic Encryption for Arithmetic of Approximate Numbers. Advances in Cryptology – ASIACRYPT 2017. 2017;409–37.
81. Smith JS, Shaffrey CI, Virginie Lafage, Schwab FJ, Scheer JK, Protopsaltis TS, et al. Comparison of best versus worst clinical outcomes for adult spinal deformity surgery: a retrospective review of a prospectively collected, multicenter database with 2-year follow-up. Journal of neurosurgery. 2015 Sep 1;23(3):349–59.
82. Ferran Pellisé, Serra-Burriel M, Vila-Casademunt A, Gum JL, Obeid I, Smith JS, et al. Quality metrics in adult spinal deformity surgery over the last decade: a combined analysis of the largest prospective multicenter data sets. Journal of neurosurgery Spine. 2021 Oct 1;1–9.
83. Quarto E, Zanirato A, Pellegrini M, Vaggi S, Vitali F, Bourret S, et al. GAP score potential in predicting post-operative spinal mechanical complications: a systematic review of the literature. European spine journal. 2022 Sep 25;31(12):3286–95.
84. Dimitrova DS, Kaishev VK, Tan S. Computing the Kolmogorov-Smirnov Distribution When the Underlying CDF is Purely Discrete, Mixed, or Continuous. J Stat Soft. 2020;95(10):1-42.
85. Yagi M, Hosogane N, Fujita N, Okada E, Tsuji O, Nagoshi N, et al. Predictive model for major complications 2 years after corrective spine surgery for adult spinal deformity. Eur Spine J. 2019 Jan;28(1):180-7.
86. Scheer JK, Osorio JA, Smith JS, Schwab F, Lafage V, Hart RA, et al. Development of Validated Computer-based Preoperative Predictive Model for Proximal Junction Failure (PJF) or Clinically Significant PJK With 86% Accuracy Based on 510 ASD Patients With 2-year Follow-up. Spine. 2016 Nov 15;41(22):E1328-E1335.
87. Pellisé F, Serra-Burriel M, Smith JS, Haddad S, Kelly MP, Vila-Casademunt A, et al. Development and validation of risk stratification models for adult spinal deformity surgery. Journal of Neurosurgery: Spine. 2019 Oct;31(4):587-99.
88. Dinizo M, Dolgalev I, Passias PG, Errico TJ, Raman T. Complications After Adult Spinal Deformity Surgeries: All Are Not Created Equal. Int J Spine Surg. 2021 Feb;15(1):137-43.
89. Barone G, Giudici F, Martinelli N, Ravier D, Muzzi S, Minoia L, et al. Mechanical Complications in Adult Spine Deformity Surgery: Retrospective Evaluation of Incidence, Clinical Impact and Risk Factors in a Single-Center Large Series. JCM. 2021 Apr 21;10(9):1811.
90. Kim KH, Noh SH, Lee HS, Park GE, Ha Y, Park JY, et al. Predicting mechanical complications after adult spinal deformity surgery using a machine learning approach. 2022.
91. Noh SH, Ha Y, Obeid I, Park JY, Kuh SU, Chin DK, et al. Modified global alignment and proportion scoring with body mass index and bone mineral density (GAPB) for improving

predictions of mechanical complications after adult spinal deformity surgery. The Spine Journal. 2020 May;20(5):776-84.

92. Gupta MC, Yilgor C, Moon HJ, Lertudomphonwanit T, Alanay A, Lenke L, et al. Evaluation of global alignment and proportion score in an independent database. The Spine Journal. 2021 Sep;21(9):1549-58.
93. P. Vatcheva K, Lee M. Multicollinearity in Regression Analyses Conducted in Epidemiologic Studies. Epidemiology. 2016;06(02).
94. Kramer O, Kramer O. Dimensionality reduction with unsupervised nearest neighbors. Intelligent Systems Reference Library. Springer, Berlin, Heidelberg. 2013, 13-23.
95. Jolliffe IT, Cadima J. Principal component analysis: a review and recent developments. Phil Trans R Soc A. 2016 Apr 13;374(2065):20150202.
96. Quinlan J. Bagging, boosting, and C4.5. Proceedings of the thirteenth American association for artificial intelligence national conference on artificial intelligence, AAAI Press, Menlo Park. 1996:725-30.
97. Kawabata A, Yoshii T, Sakai K, Hirai T, Yuasa M, Inose H, et al. Identification of Predictive Factors for Mechanical Complications After Adult Spinal Deformity Surgery. Spine. 2020 Sep 1;45(17):1185-92.
98. Suen YL, Melville P, Mooney RJ. Combining bias and variance reduction techniques for regression trees. Gama J, Camacho R, Brazdil PB, Jorge AM, Torgo L (Eds.), Machine learning: ECML 2005. ECML 2005, Lecture notes in computer science, 3720, Springer, Berlin, Heidelberg. 2006.
99. Abbott D. Applied predictive analytics: principles and techniques for the professional data analyst (first ed.). Indiana Indianapolis (Ed.). John Wiley & Sons, Inc. 2014.
100. Silva JRC, Berriel RF, Badue C, Souza, AF, Oliveira-Santos T. CopycatCNN: stealing knowledge by persuading confession with random non-labeled data. CoRRabs/1806.05476. 2018.
101. Orekondy T, Schiele B, Fritz M. Knockoff Nets: Stealing functionality of black-box models. ArXiv. 2018.
102. Futoma J, Simons M, Panch T, Doshi-Velez F, Celi LA. The myth of generalisability in clinical research and machine learning in health care. The Lancet Digital Health. 2020 Sep;2(9):e489-e492.
103. Vokinger KN, Feuerriegel S, Kesselheim AS. Continual learning in medical devices: FDA's action plan and beyond. The Lancet Digital Health. 2021 Jun;3(6):e337-e338.
104. Strobl AN, Vickers AJ, Van Calster B, Steyerberg E, Leach RJ, Thompson IM, et al. Improving patient prostate cancer risk assessment: Moving from static, globally-applied to dynamic, practice-specific risk calculators. Journal of Biomedical Informatics. 2015 Aug;56:87-93.

APPENDIX

APPENDIX 1: Permutation Feature Importance Training Set Results

Table 30: Increase in error rates after permuting a cluster

Feature	F score		Accuracy		Sensitivity	
	RF	Ruleset	RF	Ruleset	RF	Ruleset
op_sacro_iliac_fixation	57.7%	129.5%	58.2%	112.6%	1814.9%	756.9%
post_GAP_score	36.9%	46.3%	29.9%	39.3%	1497.1%	0.0%
pre_BMI	25.8%	0.0%	17.9%	0.0%	1194.6%	0.0%
pre_Age	25.3%	0.0%	19.7%	0.0%	1009.9%	0.0%
pre_TL_L_Cobb	24.1%	42.9%	17.1%	34.4%	1080.3%	0.0%
pre_ODI_score	22.0%	0.0%	14.1%	0.0%	1080.3%	0.0%
pre_GAP_score	21.6%	43.7%	14.6%	36.3%	992.0%	265.4%
pre_T10_L2_Kyphosis	20.9%	0.0%	12.8%	0.0%	1063.6%	0.0%
follow_up_time	20.8%	53.1%	15.3%	44.0%	850.5%	331.1%
op_EBL	20.2%	0.0%	12.3%	0.0%	1021.5%	0.0%
post_RSA	20.1%	0.0%	12.2%	0.0%	1020.7%	0.0%
pre_SRS22_subtotal	20.1%	0.0%	12.7%	0.0%	978.0%	0.0%
op_Hospitalization_time	19.5%	44.4%	13.6%	38.4%	0.0%	252.3%
pre_T2_T5_Kyphosis	19.2%	54.5%	11.6%	44.9%	971.8%	343.7%
post_T2_T5_Kyphosis	18.8%	0.0%	12.0%	0.0%	890.9%	0.0%
op_delta_TK	18.6%	0.0%	11.6%	0.0%	899.4%	0.0%
op_biggest_Osteotomy_type	17.7%	56.8%	10.5%	46.8%	889.6%	356.7%
pre_Major_Cobb	17.5%	0.0%	10.8%	0.0%	846.8%	0.0%
op_Osteotomy_posterior_column_number	17.2%	0.0%	10.4%	0.0%	839.7%	0.0%
pre_Coronal_Balance	17.2%	0.0%	10.4%	0.0%	0.0%	0.0%
op_num_of_level_fused	17.0%	0.0%	10.0%	0.0%	850.8%	0.0%
pre_T5_T12_Kyphosis	16.9%	0.0%	10.0%	0.0%	0.0%	0.0%
pre_Diagnosis	16.9%	0.0%	10.0%	0.0%	0.0%	0.0%
pre_Leg_Length_Discrepancy	16.8%	0.0%	9.9%	0.0%	0.0%	0.0%
pre_Number_prior_Spine_Surgery	16.8%	0.0%	9.9%	0.0%	0.0%	0.0%
pre_Pre_ODI_Personal_Care	0.0%	44.5%	0.0%	37.2%	0.0%	269.1%

APPENDIX 2: RF PPML Inference Attacks with Random Parameters – Scenario A

Table 31: Parameters of the best-performing models with 30-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	46	495	13	20	16	F	balanced
2	54	425	14	4	20	F	balanced
3	36	700	13	14	17	AUC	balanced
4	30	469	16	16	18	AUC	balanced
5	69	450	16	4	14	AUC	balanced
6	45	354	15	14	19	AUC	balanced
7	28	995	23	2	5	F	none
8	62	445	21	18	20	AUC	balanced
9	25	906	36	18	13	F	none
10	81	855	29	3	14	AUC	none

Table 32: Parameters of the best-performing models with 40-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	64	200	13	9	8	F	Balanced
2	39	977	16	5	9	F	Balanced
3	16	474	12	4	5	AUC	Balanced
4	50	620	35	14	11	AUC	Balanced
5	82	849	12	6	2	F	Balanced
6	75	293	45	14	11	F	Balanced
7	58	836	20	5	2	F	Balanced
8	74	319	19	2	19	AUC	None
9	47	953	49	9	11	AUC	Balanced
10	28	995	23	2	5	F	None

Table 33: Parameters of the best-performing models with 50-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	69	450	16	4	14	AUC	balanced
2	74	319	19	2	19	AUC	none
3	46	495	13	20	16	F	balanced
4	45	354	15	14	19	AUC	balanced
5	16	474	12	4	5	AUC	balanced
6	28	995	23	2	5	F	none
7	36	700	13	14	17	AUC	balanced
8	30	469	16	16	18	AUC	balanced
9	62	445	21	18	20	AUC	balanced
10	54	425	14	4	20	F	balanced

APPENDIX 3: RF PPML Inference Attacks with Random Parameters – Scenario B

Table 34: Parameters of the best-performing models with 30-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	98	920	13	2	14	F	none
2	78	222	17	17	18	AUC	balanced
3	46	780	12	13	13	AUC	none
4	87	940	12	8	16	F	none
5	72	734	16	7	18	F	none
6	88	662	12	9	19	F	none
7	63	766	13	18	17	AUC	none
8	100	870	12	14	12	F	none
9	36	545	12	14	17	F	balanced
10	68	745	26	15	16	AUC	none

Table 35: Parameters of the best-performing models with 40-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	16	609	59	16	3	AUC	balanced
2	1	726	54	6	4	F	balanced
3	30	692	53	9	7	AUC	balanced
4	87	940	12	8	16	F	none
5	56	917	58	20	10	F	balanced
6	78	222	17	17	18	AUC	balanced
7	67	114	59	13	1	F	none
8	88	662	12	9	19	F	none
9	28	964	58	3	11	AUC	balanced
10	98	920	13	2	14	F	none

Table 36: Parameters of the best-performing models with 50-query-attack datasets

Rank	ID	Tree Count	Mrty	Maxnode	Nodesize	Metric	ClassWt
1	1	726	54	6	4	F	balanced
2	3	903	51	15	12	AUC	none
3	4	714	46	13	13	F	none
4	5	452	28	7	2	AUC	none
5	11	256	31	13	8	F	none
6	13	955	37	13	13	AUC	balanced
7	14	659	37	15	12	AUC	none
8	15	787	42	13	7	F	balanced
9	16	609	59	16	3	AUC	balanced
10	17	722	38	8	8	F	none

9 CURRICULUM VITAE