Adversarial Machine Learning in Microfinance: Robustness and Security in Credit Scoring

by

Shuvojit Paul
21301746
Samiha Tasnim
21301332
Tasmia Akter Pranty
21301663
Md. Tasnimul Parvez Areen
21301338
Mir Abdullah Kawsar
21301558

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
October 2025

© 2025. Brac University
All rights reserved.

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Shuvojit Paul
21301746

Samiha Tasnim
21301332

Tasmia Akter Pranty
21301663

Md. Tasnimul Parvez Areen
21301338

Mir Abdullah Kawsar
21301558

Approval

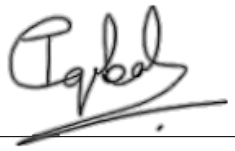
The thesis/project titled "Adversarial Machine Learning in Microfinance: Robustness and Security in Credit Scoring" submitted by

1. Shuvojit Paul (21301746)
2. Samiha Tasnim (21301332)
3. Tasmia Akter Pranty (21301663)
4. Md. Tasnimul Parvez Areen (21301338)
5. Mir Abdullah Kawsar (21301558)

of Fall, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October 10, 2025.

Examining Committee:

Supervisor:
(Member)



Dr. Muhammad Iqbal Hossain

Associate Professor
Department of Computer Science & Engineering
Brac University

Thesis Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi

Associate Professor
Department of Computer Science and Engineering
Brac University

Ethics Statement

Our research has been made for academic purposes. All the data and resources that have been used here are cited properly with references. Those, who made a contribution to our study, we mentioned their name in the acknowledgment section. Therefore, ethically there is no issue with our work.

Abstract

Adversarial Machine Learning (AML) is used to detect and resolve manipulated inputs that attempt to compromise machine learning models. Financial decision-making systems are one of the most required sectors of AML, as this sector, particularly the automated credit scoring system, is very sensitive and crucial. The current work at first proposes a model to distinguish between high-risk and creditworthy borrowers who seek loans. We named this model "LoanBuddy". Then we used eight machine learning models to train our system that can detect high and low-risk borrowers. Then we performed eight adversarial attacks on our trained models to analyze how these attacks manipulate our trained system. We also used hybrid and composite attacks to find out the most suitable and secure machine learning model for this kind of system. In numbers, we used around 40 combinations of eight base attacks. Finally, we proposed a way to defend against those attacks. Overall, our integrated methodology, spanning modeling, attack evaluation, calibration, robustness training, and operational safeguards, collectively enables a secure, interpretable, and practical credit-scoring pipeline that promotes ethical microfinance practices and mitigates fraud.

We present the accuracy, AUC, and F1 of each model's predictions as well as the accuracy of its probabilities (calibration: Brier and Expected Calibration Error/ECE). We also provide, when available, a certified robustness margin, which is the minimum amount that an input must change in order to reverse the decision. The results demonstrate that adversarially trained transformer models and calibrated monotone ensembles exhibit the strongest robustness. In contrast, unregularized neural baselines and stacked tree models are more vulnerable and require hardening.

Keywords: Adversarial Machine Learning (AML), Robustness, Credit Scoring, Microfinance, LoanBuddy Framework, Fraud Detection, AI-driven Credit Assessment, Gradient-Boosted Ensembles (CatBoost, XGBoost, LightGBM), Neural Networks (ANN, FT-Transformer), Explainable Models (EBM, Monotone Learners), Adversarial Attacks (HSJ, PGD, JSMA, AutoAttack, CAA/CAPGD, Surrogate–Decision, COVAC, Optimal Tree Evasion), Defense Mechanisms (Adversarial Training, TRADES + AWP, Calibration, Randomized Smoothing), Responsible and Secure Financial Inclusion.

Dedication

Dedicating our work to all the parents in the world through whom we come to the earth and experience countless blessings!

Acknowledgement

By the grace of Almighty Allah, we would like to express our deepest gratitude to our supervisor, Dr. Muhammad Iqbal Hossain Sir, for his continuous guidance and innumerable support. We could not have done it without him. His expertise and guidance have been significant in shaping this research. We are also thankful to our friends and family for their support during this journey.

Table of Contents

Declaration i

Approval ii

Ethics Statement iv

Abstract v

Dedication vi

Acknowledgment vii

Table of Contents viii

List of Figures xi

List of Tables 1

1 Introduction 2
- 1.1 Problem Statement . . . 3
- 1.2 Research Objectives . . . 3
- 1.3 Thesis Structure . . . 4

2 Literature Review 7
- 2.1 Adversarial Attack Definitions . . . 14
  - 2.1.1 Hop Skip Jump (HSJ) . . . 14
  - 2.1.2 Projected Gradient Descent (PGD) . . . 15
  - 2.1.3 JSMA (Jacobian-based Saliency Map Attack) . . . 15
  - 2.1.4 AutoAttack-Tabular . . . 16
  - 2.1.5 Constrained Adaptive Attack / Constraint-Aware PGD (CAA / CAPGD Cascade) . . . 17
  - 2.1.6 Surrogate → Decision Hybrid . . . 17
  - 2.1.7 Cost- and Validity-Aware Crafting (COVAC) . . . 18
  - 2.1.8 Optimal Tree-Ensemble Evasion . . . 18
- 2.2 Model Definition . . . 19
  - 2.2.1 CatBoost stacked XGBoost . . . 19
  - 2.2.2 CatBoost stacked Logistic Regression . . . 19
  - 2.2.3 Artificial Neural Network (ANN) . . . 20
  - 2.2.4 RAT-Tab . . . 20
  - 2.2.5 Calibrated Monotone Ensemble (CME) . . . 20

2.2.6 Monotone-LightGBM . . . 21
2.2.7 Explainable Boosting Machine (EBM) . . . 21

3 Data Collection and Analysis 22
3.1 Work Plan . . . 22
3.1.1 Data Collection and Synthetic Data Generation . . . 23
3.1.2 Data Preprocessing and Synthetic Data Balancing . . . 23
3.1.3 Data Analysis . . . 25

4 Model, Attack and Defense Architectures 32
4.1 Model Architectures . . . 32
4.1.1 CatBoost stacked XGBoost . . . 32
4.1.2 CatBoost stacked Logistic Regression . . . 33
4.1.3 Artificial Neural Network (ANN) . . . 34
4.1.4 C-Stack . . . 35
4.1.5 RAT-Tab . . . 36
4.1.6 Calibrated Monotone Ensemble (CME) . . . 36
4.1.7 Monotone-LightGBM . . . 37
4.1.8 Explainable Boosting Machine (EBM) . . . 38
4.2 Attack Architectures . . . 38
4.2.1 HopSkipJump (decision-based, black-box boundary search) . . . 38
4.2.2 PGD (white-box; projected first-order) . . . 40
4.2.3 JSMA (L_0-sparse saliency; targeted/untargeted) . . . 41
4.2.4 AutoAttack-Tabular (worst-of APGD + FAB + Square) . . . 41
4.2.5 CAA / CAPGD Cascade (constraint-aware PGD + search) . . . 42
4.2.6 Surrogate → Decision Hybrid (transfer + hard-label refinement) 43
4.2.7 COVAC (cost- and validity-aware crafting) . . . 44
4.2.8 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple search) . . . 46
4.3 Defense Architecture and Workflow . . . 47
4.3.1 Overview . . . 47
4.3.2 Threat Model (S) . . . 48
4.3.3 Inner Adversary: CAPGD/CAA . . . 48
4.3.4 Outer Optimizer: TRADES + AWP . . . 49
4.3.5 Defense Workflow (Step-by-Step) . . . 49
4.3.6 Calibration and Reliability . . . 51
4.3.7 Decision-Centric Controls . . . 51
4.3.8 Evaluation Summary . . . 51
4.3.9 Deployment and Monitoring . . . 51
4.3.10 Summary and Key Takeaways . . . 52

5 Adversarial Attack and Defense Implementation 53
5.1 Adversarial Attack and Defense Results and Implementation . . . 53
5.1.1 Base Accuracy Overview . . . 53
5.1.2 Hop Skip Jump (HSJ) — Decision-Based Attack . . . 53
5.1.3 Projected Gradient Descent (PGD) — White-Box Attack . . . 54
5.1.4 JSMA — Sparse L_0 Attack . . . 55
5.1.5 AutoAttack-Tabular (APGD + FAB + Square) . . . 55
5.1.6 CAPGD (CAA/CAPGD Cascade) . . . 56
5.1.7 Surrogate → Decision Hybrid . . . 56

5.1.8 COVAC (k-Edit + Validity + Cost) . . . 57
5.1.9 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple) . . . 57
5.1.10 Composite (Worst-of Family) Attacks . . . 58
5.2 Defense Implementation . . . 58
5.2.1 Data and Threat Set . . . 58
5.2.2 Defense Training (inner attack, outer loss, stability) . . . 59
5.2.3 Calibration, Decision, and Evaluation . . . 59

6 Attack-Wise Results and Defense Analysis 61
6.1 Overview . . . 61
6.2 Hop-Skip-Jump (HSJ) Attack . . . 61
6.3 Hybrid (HSJ + Surrogate Refinement) Attack . . . 62
6.4 COVAC Attack . . . 63
6.5 Projected Gradient Descent (PGD) . . . 64
6.6 Jacobian Saliency Map Attack (JSMA) . . . 64
6.7 AutoAttack (APGD + FAB + Square) . . . 65
6.8 Constraint-Aware PGD (CAPGD) . . . 66
6.9 Surrogate: Decision Hybrid Attack . . . 66
6.10 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple) . . . 67
6.11 Composite (Worst-of-Family) Attack . . . 68
6.12 Defense setup & headline results . . . 68
6.13 Why is this different & practical . . . 69
6.14 Practical Implications for Microfinance . . . 70
6.15 Future Improvements and Limitations . . . 70

7 Conclusion 72

References 76

List of Figures

3.1 Work Plan 23
3.2 Data Pre-processing and Cleaning Workflow 25
3.3 Loan Status (0 = Rejected, 1 = Approved) 26
3.4 Heatmap 27
3.5 Mutual Information Scores 28
3.6 Distribution of account_age_months 29
3.7 Distribution of average txn 29
3.8 Distribution of std txn 29
3.9 Distribution of account balance 29
3.10 account_age_months vs Loan Status 30
3.11 Average Txn vs Loan Status 30
3.12 std_txn vs Loan Status 30
3.13 account_balance vs Loan Status 30
3.14 Occupation vs Loan Status 31
3.15 Region vs Loan Status 31

4.1 CatBoost to XGBoost stacked model workflow 33
4.2 Artificial Neural Network Workflow 35
4.3 C-Stack Workflow 36
4.4 RAT-Tab Workflow 37
4.5 HopSkipJump Attack workflow (boundary search and projection steps). 39
4.6 PGD Attack Workflow 40
4.7 JSMA Attack Workflow 41
4.8 AutoAttack Workflow 42
4.9 CAA/CAPGD Attack Workflow 43
4.10 Surrogate → Decision Hybrid Workflow 44
4.11 COVAC Workflow 45
4.12 Optimal Tree-Ensemble Evasion Workflow 47
4.13 Defense architecture and training-to-deployment workflow. 50

6.1 HSJ: Original (left) vs. Attacked (center) vs. Robustness (right). . . 62
6.2 Hybrid: Original (left) vs. Attacked (center) vs. Robustness (right). . 62
6.3 COVAC: Original (left) vs. Attacked (center) vs. Robustness (right). 63
6.4 PGD: Original (left) vs. Attacked (center) vs. Robustness (right). . . 64
6.5 JSMA: Original (left) vs. Attacked (center) vs. Robustness (right). . 65
6.6 AutoAttack: Original (left) vs. Attacked (center) vs. Robustness (right). 65
6.7 CAPGD: Original (left) vs. Attacked (center) vs. Robustness (right). 66

6.8 Surrogate→Decision: Original (left) vs. Attacked (center) vs. Robustness (right). 67
6.9 MILP/Leaf-Tuple: Original (left) vs. Attacked (center) vs. Robustness (right). 67
6.10 Composite: Original (left) vs. Attacked (center) vs. Robustness (right). 68
6.11 Reliability under Clean and Attack (ECE shown): Clean, CAPGD(S), PGD-U 69
6.12 Selective Prediction under Clean and Attack: Coverage–Accuracy . . 70

List of Tables

4.1 Performance overview (before vs. after defense). 51

5.1 Clean (pre-attack) accuracy by model. 53
5.2 HSJ: accuracy degradation. Drop shown in percentage points (pp). . 54
5.3 PGD: accuracy degradation on neural models. 54
5.4 JSMA: accuracy degradation on neural models. 55
5.5 AutoAttack-Tabular: accuracy degradation. 55
5.6 CAPGD: accuracy degradation. 56
5.7 Surrogate→Decision hybrid: accuracy degradation. 56
5.8 COVAC: accuracy degradation across models. 57
5.9 MILP/leaf-tuple: accuracy degradation on tree ensembles. 57
5.10 Composite attacks: overall accuracy degradation. 58

Chapter 1

Introduction

Microfinance institutions (MFIs) increase the financial services to individuals, small shops, and micro-enterprises, which otherwise are not accessible to traditional banks. In some nations, such as Bangladesh, the credit market in the rural areas is dominated by MFIs; 10 big institutions control 87 percent of industry capital and 81 percent of loans in Bangladesh and have nearly 30 million low-income clients. Micro credit assists many borrowers to start something that generates income, but it also subjects them to misappropriation and fraud. In order to expand outreach and efficiency, MFIs are moving to machine-learning-based automated credit-scoring models, which will scale up loan issuance and improve risk analysis.

Current scoring systems are very traditional and simple, but not very reliable. That is why we are moving towards updated systems run by machine learning. Here, we are proposing a model, which is called "LoanBuddy." As much as automation has multiple beneficial features, it does have intentional weaknesses, such as Adversarial Machine Learning (AML). In AML, attackers attack algorithmic decision-making by creating input features, including transaction histories or synthetic identity attributes, intending to cause them to be misclassified as legitimate to grant a loan to either a high-risk or non-qualified borrower. This manipulation poses a threat to the stability of the institution and responsible lending habits. These are the vulnerabilities that our paper explores through applying AML tools to identify, deduce, and protect credit score systems against malicious manipulations.

LoanBuddy architecture suggests a structure that utilizes the true transaction data and the assumed behavioral features, where the label of *LoanBuddy Flag* is enforced to further differentiate between creditworthy and high-risk borrowers. It estimates eight complementary families of models, which comprise (1) CatBoost stacked with XGBoost, (2) CatBoost stacked with Logistic Regression, (3) Artificial Neural Networks, (4) C-Stack (CatBoost stacked with monotonic XGBoost), (5) RAT-Tab, (6) Calibrated Monotone Ensemble (CME), (7) Monotone-LightGBM, and (8) Explainable Boosting Machine (EBM). It implements eight large adversarial attack families, including HopSkipJump (HSJ), Projected Gradient Descent (PGD), Jacobian-based Saliency Map Attack (JSMA), AutoAttack-Tabular, Constraint-Aware Projected Gradient Descent (CAPGD cascade), Surrogate-Decision Hybrid, Cost and Validity Aware Crafting (COVAC), and Optimal Tree-Ensemble Evasion. Models are tested on eight composite attacks (i.e., HSJ + Surrogate + COVAC) where complementary

modes are used to form adversarial situations in the real world. Adversarial training, TRADES (TRadeoff-inspired Adversarial DEfence) + AWP (Adversarial Weight Perturbation) regularization of neural and transformer architectures, constraint-aware projection, isotonic and temperature calibration of trustworthy probability estimates and fairness, and randomized smoothing of certified robustness of monotone XGBoost variants are the defense mechanisms that correspond to each type of model.

The overall goal is to establish a credit-scoring environment that is secure, transparent, and impervious to attacks in order to reduce fraudulent grants and encourage ethical, transparent microlending practices. Through the integration of responsible AI and AML principles, the LoanBuddy framework introduces the concept of sustainable financial inclusion.

1.1 Problem Statement

Risk mitigation in credit assessments has long been a core challenge and becomes even more critical as MFIs adopt data-driven automation. Although ML-based scoring has significantly improved predictive accuracy, such systems remain susceptible to manipulation. Fraudsters can exploit feature perturbations to obtain loans illicitly or cause legitimate applicants to be misclassified. This undermines institutional integrity and trust in digital lending. Furthermore, most conventional ML models were never designed to resist such adaptive adversarial behavior. Hence, there is an urgent need to develop robust, attack-resilient, and ethically reliable ML systems for the microfinance domain. Through targeted adversarial defenses, this study aims to strengthen microfinance credit pipelines so that lending decisions remain accurate, fair, and manipulation-proof.

1.2 Research Objectives

The primary objective of this research is to design and integrate an Adversarial Machine Learning-based credit-scoring framework capable of detecting and mitigating diverse adversarial attacks within microfinance environments. Specifically, the work aims to:

1. Develop and evaluate eight model architectures under a unified AML evaluation protocol.
2. Implement the eight core attack families: **HSJ (Hop Skip Jump), PGD (Projected Gradient Descent), JSMA (Jacobian-based Saliency Map Attack), AutoAttack-Tabular, CAA/CAPGD (Constraint-Aware Projected Gradient Descent), Surrogate Hybrid, COVAC (Cost and**

Validity Aware Crafting), and Optimal Tree-Ensemble Evasion to measure vulnerabilities across model types.

3. Construct eight composite attacks combining multiple adversarial mechanisms to simulate real-world layered threats.

4. Deploy complementary defense strategies, adversarial training, robustness–accuracy trade-off (TRADES) and Adversarial Weight Perturbation (AWP), a weight-space regularizer, constraint-aware COVAC, isotonic/temperature calibration, and randomized smoothing to reinforce model resilience.

5. Assess model robustness through comparative metrics such as accuracy, AUC(Area Under the Curve), F1 score, calibration error, and certified robustness margins.

Through these objectives, the study seeks to establish a comprehensive AML framework for credit-risk modeling that balances predictive performance, interpretability, and adversarial robustness.

1.3 Thesis Structure

There are seven chapters in this thesis. Each of the chapters covers a subset of the research on the topic of Adversarial Machine Learning in Microfinance, under the name *Robustness and Security in Credit Scoring*. The framework of LoanBuddy, data preparation, model design, adversarial evaluation, defense mechanisms, and future directions all are covered together.

Chapter 1: Introduction
The point of the first section is that microfinance institutions (MFIs) are the main providers of cheap loans to poor people all over the world. It signals the increased reliance on machine-based credit scoring and a security threat that comes with it - Adversarial Machine Learning - a situation where malicious actors manipulate input data in order to trick ML models. To get rid of this problem, the chapter talks about an attack-resistant yet intelligent loan-approving system featuring the LoanBuddy system. Defense mechanisms are implemented to not only improve predictive performance but also to increase resistance to fraud. This chapter closes with a link between the research goal and ethical, safe, and fair financial inclusion.

Chapter 2: Literature Review
The literature review has reviewed past studies on ML-based credit scores, AML techniques, and financial system stability. It contrasts traditional models like the Logistic Regression and the Random Forest with newer models like gradient-boosted trees and identifies the weaknesses of these models in explainability, fairness, and attack resistance. The review follows the development of AML strategies, white-box, black-box, and hybrid attacks, and reasons why a domain-specific defense structure on microfinance is required.

Chapter 3: Data Analysis and Collection
This chapter explains our data collection and preprocessing of Upay transaction data, a massive representative dataset on mobile microfinance. The cleaning procedures were the elimination of the null values, coding of the categorical variables, identification of the outliers, and scaling of the features. The key predictors were determined using the exploratory analysis, and they included transaction count, account balance, and buddy ratio. Correlation and descriptive statistics also confirm the fact that the number of LoanBuddy interactions, account balance, and transaction patterns are key predictors as they significantly determine loan-approval decisions. Graphs are used to depict correlations between features and target variables. These measures aim at making sure that the dataset is clean, balanced, and fit to train the model and perform adversarial experimentation. All these actions made the dataset clean, balanced, and prepared to train powerful ML models in adversarial testing.

Chapter 4: Model Architectures
In Chapter 4, the authors describe the architecture of the eight LoanBuddy model architectures: CatBoost-XGBoost, CatBoost-Logistic Regression, ANN, C-Stack, RAT-Tab (FT-Transformer with TRADES + AWP), CME (Monotone-XGBoost with Isotonic Calibration + Randomized Smoothing), Monotone-LightGBM (adversarially trained + Isotonic), and EBM (GA^2M monotonic links). The architecture is described by each pipeline, calibration strategy, and principle of robustness. The section demonstrates that model diversity, ranging from explainable additive models to transformers, can be used to achieve a compromise between accuracy, interpretability, and resilience.

Chapter 5: Adversarial Attack and Defense Implementation
This is the core chapter of the thesis. It compares models to eight black-box, white-box, and certified threat models. The methodology, models impacted, loss of performance, and post-defense recovery are given against each attack. Adversarial retraining, TRADES + AWP, calibration (temperature/isotonic), COVAC validity checks, and randomized smoothing are all defense methods that can be used to restore robustness. Composite worst-of attacks overlay service vulnerabilities and show why multi-stage defenses are needed against production microfinance systems.

Chapter 6: Results and Model Analysis of Attack-Wise
Chapter 6 summarizes the quantitative findings of all models and attacks. It introduces a resilience ratio (r) measure in order to compare resilience among settings. The findings reveal that the best compromise between accuracy and robustness between calibrated monotone ensembles (CME, EBM) and adversarially trained transformers (RAT-Tab) are demonstrated. The most vulnerable are the irregularized neural and stacked tree models. Graphs show the tendencies of degradation and restoration, and the layered defenses are effective.

Future Improvements and Limitations section talks about the existing limitations, including reliance on datasets, computational expenses, and lack of optimization of fairness. It specifies the future work: to multi-institutional datasets, to make defense

more efficient, to combine fairness constraints, and to provide certified robustness of hybrid tree ensembles.

Chapter 7: Conclusion
Finally, the conclusive chapter restated the fact that adversarial attacks would be a real threat to the ML-based credit scoring in microfinance and could be used to commit fraud. This paper overcomes these weaknesses by creating the LoanBuddy application, which is a collection of credit-scoring systems that include powerful anti-adversarial countermeasures. The study shows that hybrid models in combination with adversarial training have a significant opportunity to decrease the risk of fraud, which provides an ethical, secure, and responsible lending process. Finally, the results are used as a guideline on how AI can be safely implemented in financial inclusion in the future, which can be accessible to different contexts in microfinance.

Chapter 2

Literature Review

Adversarial Machine Learning came into being as a method to test and improve machine-learning algorithms against malicious deformation. Based on robust optimization, Madry et al.[15] introduced PGD as a first-order adversary. Their research shows that PGD is a "universal" attack that is essential to adversarial training and can be used to approximate the most powerful gradient-based adversary. Saliency maps from adversarial training can be used to determine which output features are most relevant for particular input perturbations, according to recent work (JSMA, Papernot et al.)[9]. In this manner, a small change of the feature space (≈4%MNIST) is sufficient to create an adversarial sample with trustworthy tasks. HSJ, a query-efficient decision-based attack that can be used in black-box environments, was proposed by Chen et al. Following this body of research by comparing earlier black-box techniques, it experiments with decision boundary iterations to converge to the decision boundary adversarial examples with fewer queries. Before every iteration, the decision boundary is sent to it[19].

AutoAttack assembles diverse, parameter-free attacks for reliable evaluation (Croce Hein, 2020a). Its components—APGD, FAB for minimal-norm perturbations, and Square Attack for query-efficient black-box settings—stress different failure modes (Croce Hein, 2019; Andriushchenko, Croce, Flammarion, Hein, 2020). For tabular data, injecting validity (immutability, type, and relationship rules) and cost/k-edit constraints follows recent guidance from TabularBench, CAA/CAPGD, and cost/utility- aware formulations (Simonetto et al., 2024a, 2024b; Kireev et al., 2023; Ben-Tov et al., 2025)[18], [21], [22], [30], [33], [35], [36], [37].

In black-box settings, adversaries often train a surrogate and leverage transferability to seed attacks (Papernot et al., 2017; Liu et al., 2017)[6], [8]. Once close to the decision boundary, decision-based methods refine adversarial examples using only label access—Sign-OPT (hard-label, query-efficient), HopSkipJumpAttack (HSJ), and Boundary Attack (Cheng et al., 2020; Chen, Jordan, Wainwright, 2020; Brendel, Rauber, Bethge, 2018)[10], [20]. Our "surrogate→decision" hybrid aligns with practical API-style threat models, especially for tabular services[8], [9], [10], [20].

COVAC operationalizes a constrained attack policy: limit the number of feature edits (k-edit/sparsity), enforce validity constraints (types, immutables, relational), and penalize edits by feature-specific costs, then fall back to APGD/Square when gradients or scores are available. This design is grounded in cost/utility-aware threat models and feasible-attack frameworks for tabular data (Kireev et al., 2023; Ben-Tov et al., 2025)[30], [33].

Recent work introduces CAA, a hybrid ensemble that integrates constrained gradient-based and search-based procedures for tabular data—including CAPGD for efficiency under tabular constraints—explicitly handling immutability, categorical features, and feature-relationship rules (Simonetto, Ghamizi, Cordy, 2024b)[35], [36].

Kantchelian et al., Evasion and Hardening of Tree Ensemble Classifiers — formal MILP reduction and optimal evasion for tree ensembles. This is the canonical, rigorous reference for MILP-based optimal evasion of tree ensembles and directly supports the "Optimal Tree-Ensemble Evasion" item in your attack map [5].

Gradient-boosted trees remain strong for tabular tasks. CatBoost addresses categorical encodings and prediction-shift via ordered boosting (Prokhorenkova, Gusev, Vorobev, Dorogush, Gulin, 2018)[11], while XGBoost offers scalability and monotonicity constraints that encode domain priors (Chen Guestrin, 2016; XGBoost, 0.90 docs). Stacking their calibrated outputs with a logistic meta-learner (Wolpert, 1992) and temperature scaling (Guo, Pleiss, Sun, Weinberger, 2017) yields well-calibrated probabilities suited to risk decisioning. In credit-risk and financial-inclusion contexts, calibrated, monotone ensembles align with policy constraints and auditability needs (Noriega, Rivera, Herrera, 2023)[1], [4], [7], [31].

FT-Transformer adapts Transformer building blocks to tabular settings (Gorishniy, Rubachev, Khrulkov, Babenko, 2021). For robustness, TRADES formalizes the robustness–accuracy trade-off via a surrogate-loss objective (Zhang et al., 2019), and Adversarial Weight Perturbation (AWP) regularizes weights to reduce robust overfitting (Wu, Xia, Wang, 2020). RAT-Tab couples FT-Transformer with a brief pre-defense phase (data augmentation/noise) followed by TRADES+AWP fine-tuning to harden the model against gradient-based and transfer-based adversaries common in tabular evaluations[17], [24], [26].

CME combines monotone XGBoost with isotonic regression for probability calibration (Zadrozny Elkan, 2002), then adds a randomized smoothing layer for certified robustness. Randomized smoothing provides probabilistic certificates against norm-bounded perturbations for any base classifier (Cohen, Rosenfeld, Kolter, 2019), with extensions offering tighter certificates, including discrete and tree-based models (Lee, Yuan, Chang, Jaakkola, 2019; Horváth, Mueller, Fischer, Vechev, 2022). The CME pipeline is compatible with tabular constraints with a formal guarantee radius that is determined on agreed metrics. Generalized Additive Models with pairwise interactions (GA^2M) were proposed by Lou et al. (KDD 2013). They consist of a family of interpretable models which can be relatively more interpretable compared to high-dimensional interaction models. They can also be relatively more accurate compared to random walk models. Their work demon-

strates that competitive performance with constrained considered model families can be obtained without losing transparency of the per-feature behaviour that can be used in a domain audit and when an adversary can observe that the decision structure is exploitable. This paper specifically applies to the models that were employed in the experiments, including EBM/GA^{2}M variants; the interpretability that makes GA^{2}M appealing also produces a constrained, predictable decision space that can be used by some feature-sparse attacks targeting particular features. Prokhorenkova et al. explain CatBoost, along with the engineering that enables the gradient boosting of CatBoost to be very useful on tabular/categorical problems. Their paper models algorithm characteristics such as ordered/oblivious trees and ordered target statistics and empirically show results on a wide range of tabular benchmarks, making CatBoost a strong baseline of tabular classification, and a realistic production goal of adversaries. Madry et al. introduce the formalisation and empirical validation of adversarial training with Projected Gradient Descent (PGD) as a powerful adversary of neural networks and show how neural network training to a powerful adversary transfers robustness to other such attacks and identifies the trade-offs between accuracy, robustness, and training cost. For our thesis, Madry's work underpins the design and interpretation of white-box defenses and experiments on ANN and RAT-Tab (TRADES/AWP-style schemes): PGD/adversarial training is the baseline defense for white-box gradient attacks and motivates why ensemble trees (which lack accessible gradients) need different defense strategies[2], [3], [12], [14], [27].

Kantchelian et al. study the optimal evasion problem for tree ensembles and provide exact and approximate algorithms (reductions to mixed integer programs and coordinate-descent heuristics) that compute minimal perturbations for evading boosted trees and RFs. Their results are stark: highly accurate tree ensembles can be extremely easy to evade under small perturbations when optimized. This paper is foundational for your work on MILP/leaf-tuple style optimal evasion attacks (used in our CME / Monotone-LightGBM / EBM evaluations)[5].

Kurniawan et al.'s (2022) paper initially aimed to create a control theory-based model for the microfinance sector, especially for rural or underdeveloped areas, for example, where there is insufficient loan data and liability of the group. The paper initially focused on ensuring financial fairness, social development, and sustainability to improve the credit scoring system. The model they used could successfully create a loan policy that can balance the risk factor and fairness. It could hold the result even in the situation of missing data and group liability dynamics. But the paper has some flaws and failings. The model they used heavily depends on the synthetic dataset, which is likely to be unable to fully capture the real-world challenges and scenarios in diverse microfinance environments. Also, the model they used might not be easily and initially applicable to smaller MFIs due to a lack of technological capabilities. They promised to work on the real-world application of the model and to adapt the model to a more dynamic economic environment[28].

The study of Oskarsdóttir et al. (2020) focuses on improving credit scoring models based on the data of smartphones for the population who are unbanked through a microlending application. It is mainly based on Machine Learning, and it also

focuses on ethical concerns and the privacy of the users. The paper used a variety of machine learning models to train the system, including complex network analysis. The model could successfully show that using smartphone data is capable of potentially improving credit scoring accuracy. Their approach could enhance the performance of microlending applications. However, the paper has some flaws and concerns; their dataset is so vague, and there are some unresolved issues about the ethical use of 7 personal smartphone data, especially in the case of privacy. The use of personal smartphone data might not be usable in certain regions of the world. They promised to apply better methods to ensure better data protection and regulation. It also promises to work with a larger population and diverse countries, ensuring ethical challenges[23].

The goal of Condori-Alejo et al.'s (2021) paper is to improve rural microcredit assessment, specifically in Peru, using machine learning. It focuses on reducing the credit risk in rural MFIs by using predic tive models to support decision-making. They took preprocessed data from 15015 clients who applied for microcredit. They trained some machine learning models and tested them thoroughly. It includes Ar tificial Neural networks, logistic regression, Random Forest, Support Vector Machines, and decision trees. They were validated through accuracy and AUCROC. ANN achieved the highest accuracy among all the models, which is 93.72%. They found out that the new model outperformed the previous traditional model used by the company. It was more than 16% accurate than the previous one. It has also some flaws. Namely, the study is specific to a region, so the outcome will not fit well in other countries with different economic and social struc tures. It also lacks the interpretability of the models, which leads to confusion among the institutions. They promise to apply their model to different geographical situations. They are also willing to work with more transparency to make it more feasible for the institutions[25].

In the present study, Aadithya et al. (2025) [38]explore the possibility to use XGBoost algorithm to forecast loan approvals and promote efficient decision-making in financial organizations by limiting the rate of defaults. This was done to come up with a confidence-inducing and data driven model that could measure the eligibility of loans based on factors such as income, loan amount, credit history, among others. Such decision was made due to the high results of XGBoost in manip ulating complex data sets, anti-overfitting and optimization of classifi cation based on gradient boosting and regularization. The model was very accurate, and there are no concrete values given, as the param eters of the performance have been measured in terms of precision, recall, and F1 score. Examination of feature importance also helped understand which factors contributed mostly to making predictions, making them easier to interpret. Nonetheless, it has the drawbacks of possible difficulties with working with the highly imbalanced data and requiring a significant amount of hyperparameter tuning, which may take a lot of computing resources.

In the present study, Aadithya et al. (2025)[38] explore the possibility to use XGBoost algorithm to forecast loan approvals and promote efficient decision-making in financial organizations by limiting the rate of defaults. This was done to come up with a confidence-inducing and data driven model that could measure the eligibility

of loans based on factors such as income, loan amount, credit history, among others. Such decision was made due to the high results of XGBoost in manip ulating complex data sets, anti-overfitting and optimization of classification based on gradient boosting and regularization. The model was very accurate, and there are no concrete values given, as the param eters of the performance have been measured in terms of precision, recall, and F1 score. Examination of feature importance also helped understand which factors contributed mostly to making predictions, making them easier to interpret. Nonetheless, it has the drawbacks of possible difficulties with working with the highly imbalanced data and requiring a significant amount of hyperparameter tuning, which may take a lot of computing resources.

Noriega et al. (2023) analyzed the "Machine Learning (ML) models" regarding credit risk in the financial sector, and for this reason, XGBoost, LightGBM, and Random Forest were also considered[31]. These models are widely used as they are capable of performing on large and imbalanced data sets, which are prevalent in financial problems. "XG Boost" proved to be exceptional, with an "AUC" of 93.7%, categorizing it as one of the most efficient models for credit risk prediction. A limitation deemed important is the "lack of explainability" in developed ML models; hence, the authors' call to promote "XAI" tools such as "SHAP" or GSCI to better explain the function of model-made decisions to relevant parties. This study also included problems with multicollinearity in datasets and highlighted the significance of feature selection depending on demographic characteristics and behavioral patterns for improved predictions. The problem of data imbalance was also another problem that was faced during the study. These methods, like SMOTE, random undersampling (RUS), and K-fold cross-validation, were used to solve this challenge. Although such methodological advances were made, the authors have found some remnant challenges in processing textual and image data. As a way of addressing these deficiencies, they suggested the incorporation of Big Data instruments, which are likely to increase flexibility and effectiveness within the operational financial environments.

Wang et al. (2023)[32] offer a survey of the adversarial attacks on deep learning models, focusing on Projected Gradient Descent (PGD) and Carlini–Wagner (CW) approaches. Such methods take advantage of vulnerabilities in deep neural networks (DNNs) and highly endanger computer vision and autonomous driving, among other areas. The authors list various types of attacks and draw lines of gaps in DNN systems of each type. They consider modern defensive techniques such as adversarial training and regularization-based methods that strive to make the models more robust. In the analysis, they point out significant limitations, which include the computational intensity of the training processes that makes real time deployment impractical and the problem of gradient masking, since models can be exposed to easy attacks but not complex ones. Moreover, Wang et al. point towards the still emergent risk of physical-world adversarial attacks that utilize physical impediments, and they observe that most defenses currently fail in both digital and physical worlds. The existing countermeasures have detection rates of about 93.7%, but there is a problem of scalability and cross-architecture portability. This asymmetric argument proposes that there has to be a creation of lightweight yet scalable defenses, which will not be compromised by both simulation and real-world scenarios of various

adversarial agents.

From Granström et al. (2019), we can learn that the thesis Loan Default Prediction using Supervised Machine Learning Algorithms thesis works with finding the best machine learning method for predicting loan defaults.[13] They have compared several techniques to determine which model provides the most accurate prediction. Logistic regression, Random Forest, Decision Tree, AdaBoost, XGBoost, Artificial Neural Networks, and Support Vector Machine methods were used for better results to address the imbalance in the data set. Where defaults were fewer, the SMOTE (Synthetic Minority Oversampling technique) was applied. SMOTE works by creating synthetic examples based on feature space similarities between existing minority class samples. This improves the models' ability to generalize the minority class, which works without duplicating the data. XGBoost was combined with SMOTE to improve the prediction accuracy of the system. The F-score was used as the primary metric, where the highest F-score is 0.611 (without SMOTE) and 0.609 (with SMOTE). However, their study has some limitations as the data is biased. We can also see a marginal SMOTE impact on the data and a limited feature selection method.

We can learn from the Adversarial Machine Learning in Industry: A Systematic Literature Review that they have used AML techniques in industrially used machine learning models, focusing on how these models are vulnerable to adversarial attacks. ML models used in industries can misclassify inputs, reduce accuracy, or leak sensitive data if any malicious actions are intended against them. To prevent this at first they detected and corrected "poisoned" data to prevent DATA POISONING. They have also used data normalization or transformation to standardize the input for any attack detection. The effectiveness of their model's defense varied across different attack types. Some of their models faced issues like misclassification under adversarial attacks. For that, they have suggested improvements like adopting privacy-preserving frameworks and robust testing metrics to evaluate the vulnerability of the industrial models. They have also suggested implementing an industry-standard attack-detection tool using real-world data. Some models are vulnerable to zero-day attacks, but this remains difficult to defend against due to the novelty of attack vectors. They have used publicly available datasets for testing the vulnerability of ML models. Lastly, their research works on attack vectors, defense mechanisms, and how research needs to bridge the gap between academic insights and practical industry implementations[34].

Finally, here we can learn that Generative Adversarial Networks (GANs) for Synthetic Financial Data Generation: Enhancing Risk Modeling and Fraud Detection in Banking and Insurance as fraud detection in this sector is vital to traditional systems sometimes fail to identify more complex fraudulent activities GANs machine learning model is proposed here to increase the accuracy and detect fraud. The GANs model has two components: a generator and a discriminator. Here, one generator creates fake data and another tries to differentiate be tween real and fake data. These two components are trained together to improve its ability to produce realistic data over time. Through its iterative training, the generator learns to create better fraud examples and the discriminator becomes more adaptable at identifying

fraud. The adversarial nature of the training, where both models are contin uously trying to outperform each other, makes a robust model capable of detecting fraud in a short time. However, their research has some gaps; at first, explicit testing and validation matrices like accuracy, precision, and recall rates are missing which can be important for the GAN model. The GAN model needs a large amount of competitive power and a significant amount of level data but these data are not available for the banking and insurance environment[29].

The article explores how stacked generalization can provide viable improvements over individual learners on a traditional tabular standard (Adult Income); the base models are KNN, Gaussian Naive Bayes, and SVC, and the meta-learner is Logistic Regression. The authors contrast the accuracy, precision F1, and confusion matrices of unscaled and scaled settings after standard preprocessing (label encoding and min max scaling). It was found that although unscaled data stacking mostly follows GaussianNB and SVC deteriorates, the stack is now comparable in strength to the strongest single model (typically KNN or SVC) that achieves small, yet reliable, improvements in F1. The exposition is well presented, with diagram and pseudocode making the two stage training/prediction flow simple to read through, and the ablation on scaling providing a useful insight on how distance- and margin-based learners propagate the stack. Nonetheless, the research does not include modern tabular baselines (e.g., gradient-boosted trees such as XGBoost/LightGBM/CatBoost), does not run K-fold out-of-fold stacking (and thus runs the risk of leakage and lack of meta-feature coverage), and does not provide any AUC/PR-AUC, calibration, or statistical significance tests, therefore making it difficult to assess the usefulness or robustness of its decisions. The moderate disproportions of the dataset and the lack of any fairness and shift-based analyses also restrict the external validity to high-stakes contexts like credit. In general, the paper is a clear, pedagogical exercise on how a simple two-layer stack with a logistic meta-learner can be as good as or slightly better than the best classical classifier with a proper scaling. But stronger baselines, OOF stacking, more detailed metrics, and robustness/fairness checks would be required to make conclusions that would be useful in production [1].

To point out that while the reviewed studies enrich significantly the field of credit scoring detection, some limitations can be pointed out. Firstly, most of them were based on credit card evaluation or usual credit scoring without discussion of such specifics as low-quality data, a limited number of transaction histories, and group responsibility in microcredit. Secondly, there are virtually no prior works that describe the adversarial attacks that could manipulate the machine learning models in giving estimates that are off the actual values in real-world applications. Indeed, most of the studies employed models like LR, RF, and SVM to demonstrate their importance, but they have inadequate resistance to adversarial attacks. Lastly, the limited use of non-conventional data like social media, smartphone data, etc, is confronted with issues of privacy invasion and data openness.

Therefore, we propose to offer an improved solution by incorporating adversarial defenses into learning algorithms specific to microfinance. Therefore, the present study will provide a holistic solution for the verification of microfinance machine learning

models that are more robust and secure, combining adversarial training and defense measures, including fuel feature perturbation detection and gradient masking. Moreover, we would assess how such systems work in other real-life microfinance environments regarding security and financial opportunities for small borrowers and clients.

2.1 Adversarial Attack Definitions

We used 8 adversarial attacks that deliberately make small change to the input to deceive the machine learning model, while keeping the input plausible. In credit-scoring/loan prediction, a few fields, e.g., rounding income, tweaking ratios, switching a category, are modified within valid ranges so that the approve/ reject decision flips without looking suspicious. Two types of attacks were used: white box attack, where the attacker knows the model's internal (architecture, weight) and can compute gradients with respect to input and black box attack, where the attacker does not know the internal but can query the model to see probabilities/scores, approve/reject labels, but it doesn't use gradients. As mentioned, our eight attack families cover all three of the attacker-knowledge regimes under which we evaluate: white-box (full gradients), black-box/score-based (probabilities only), and black-box/decision-based (labels only).

2.1.1 Hop Skip Jump (HSJ)

HSJ is a label-only, black-box search that walks the estimated normal after binary searching to the boundary. It matches real APIs with approve/reject label, where it's visible—projects back to valid ranges and one-hot (exactly one active bit per categorical group); cap queries for each step. It only works with yes/no answers.

Threat model: hard-label black-box (only top-1 decision or approve/reject threshold available). Query-efficient and designed for settings with no score or gradient access.

Objective (implicit):

$$\min_{\delta} \|\delta\| \quad \text{s.t.} \quad f(x+\delta) \neq f(x), \quad x+\delta \in \mathcal{C},$$

where $\mathcal{C}$ is the domain of valid tabular perturbations.

Description & execution: HSJ locates the decision boundary by binary searching along random directions from the original point until a boundary-crossing example is found, then estimates a local normal to take refined, smaller steps toward a minimal perturbation. Because only hard labels are observed, HSJ emphasizes query efficiency (fewer model calls) and uses adaptive step sizes to converge to near-minimal edits.

Validity handling: each candidate $x + \delta$ is projected to $\mathcal{C}$ immediately (numeric bounds, integer/currency snapping, valid categorical encodings). Monotone constraints are enforced by rejecting or reprojecting steps that violate specified directional rules.

2.1.2 Projected Gradient Descent (PGD)

Projected Gradient Descent is a white box attack, where the attacker intentionally designs input to alter the decision of a machine learning model. Adversarial training is powered by this standard, strong baseline, which is stable, accurate, and nearly worst-case for gradient-based attackers. We used the constraint (max change per feature) on the original sample.

PGD-U (Untargeted): push away from the true class (make it wrong).

PGD-T (Targeted): push toward a specific wrong class (flip in a chosen direction).

Threat model: full white-box access (model gradients available). The canonical first-order iterative adversary for differentiable models.

Objective (iterative update):

$$x^{t+1} = \Pi_{\mathcal{C}} \left(x^t + \alpha_t \operatorname{sign} \left(\nabla_x \mathcal{L}(f(x^t), y) \right) \right),$$

with $\Pi_{\mathcal{C}}$ the projection onto the feasible set.

Description & execution: PGD is effective at finding high-loss, misclassified points for neural and transformer models. In tabular settings PGD is adapted by using continuous relaxations for discrete features during optimization and snapping/repairing to the nearest valid values after each step.

Validity handling: after every gradient step the sample is projected to $\mathcal{C}$ (the domain of valid tabular perturbations), which enforces ranges, quantization, categorical integrity, and optional monotone directions; gradient updates on categorical fields are mapped back through nearest-valid encodings.

2.1.3 JSMA (Jacobian-based Saliency Map Attack)

Threat model: white-box; targeted or untargeted but optimized for sparse (few-feature) changes. Useful where minimal observable edits matter.

Saliency selection (feature score):

$$S_j(x) = \left(\frac{\partial f_t(x)}{\partial x_j}\right)_+ \cdot \left(-\sum_{k \neq t} \frac{\partial f_k(x)}{\partial x_j}\right)_+ ,$$

and the algorithm greedily selects $j^* = \arg\max_j S_j(x)$.

Description & execution: JSMA ranks features by how much a small change would increase the target class' score while decreasing others, then greedily applies bounded edits to the top features until the target is achieved or the L_0 budget is exhausted. Its sparsity is attractive in tabular finance because it models attackers who change only a few observable attributes.

Validity handling: all edits are constrained to discrete/quantized steps and categorical flips must respect predefined valid swap lists; monotone features are excluded from edits in disallowed directions.

2.1.4 AutoAttack-Tabular

For a trustworthy assessment, a set of parameter-free attacks (such as APGD-CE/APGD-DLR, FAB, and Square) are executed together. It removes attack tuning bias and does a standardized robustness check. Implements projection to S (ranges/one-hots) when needed; maintain the worst (strongest) result per sample; and then runs the set. It is a mixed attack where its gradient components APGD-CE/APGD-DLR, FAB are white box attacks and Square is a black box attack.

Threat model: white-box and score-aware (ensemble of complementary attacks). Designed as a tuning-free, conservative stress test.

Objective (worst-of):

$$\tilde{x} = \arg \max_{a \in \{\text{APGD,FAB,Square}\}} \mathcal{L}(f(\tilde{x}_a), y),$$

where $\tilde{x}_a$ is the adversarial example found by attacker a under the same budget and feasibility constraints.

Description & execution: AutoAttack combines APGD (adaptive PGD), FAB (margin-based), and Square (score-based search) to return the worst outcome for each sample. For tabular data the inner attacks are adapted with projections and feasible step sizes; the worst-of aggregation yields a robust lower-bound for model performance under strong white-box stress.

Validity handling: each internal step uses $\Pi_{\mathcal{C}}$ and respects discrete and monotone constraints; Square's score-based proposals are constrained to plausible value ranges to avoid unrealistic samples.

2.1.5 Constrained Adaptive Attack / Constraint-Aware PGD (CAA / CAPGD Cascade)

Threat model: white-box but explicitly enforces domain constraints and searches for feasible low-cost edits. Intended to model practical adversaries limited by plausibility and operational rules.

Objective (PGD + discrete search):

$$x^{t+1} = \Pi_{\mathcal{C}} \left(x^t + \alpha_t \nabla_x \mathcal{L} \right), \quad \text{then perform discrete local search on top-K features,}$$

aiming to reduce edit count or cost under a feasibility budget.

Description & execution: CAA/CAPGD alternates between continuous constrained gradient steps and discrete combinatorial refinements (greedy or evolutionary local search). This hybrid finds perturbations that maximize misclassification while minimizing observable edits and satisfying business rules. It is specifically useful for tabular domains where pure gradient updates can propose implausible values.

Validity handling: the cascade explicitly encodes constraints (e.g., ratios, monotonicity, currency rounding) into the projection and search operators; discrete search includes plausibility scoring (outlier detection) to avoid unrealistic counterfactuals.

2.1.6 Surrogate → Decision Hybrid

Surrogate (transfer) is a decision-based black-box attack; it is trained on labels. We take a small copy of the model to train using the same inputs and outputs. We find gradient-based tweaks on the copy and add them to the real model. It is cheap, fast and it transfers weaknesses.

Threat model: black-box with limited query access; adversary trains a differentiable surrogate from inputs/labels or public data, then refines candidates against the true model using decision-only methods.

Two-stage objective: (1) train surrogate g approximating f and craft x' on g; (2) refine via minimal δ so

$$f(x' + \delta) \neq f(x), \quad \text{with } x' + \delta \in \mathcal{C}.$$

Description & execution: The surrogate guides which features are influential (transfer step, e.g., PGD on g). Because transfer is imperfect, a low-budget decision-only refinement (HSJ/Boundary) is applied to the original model using a small number of label queries to obtain an actual successful evasion. This method is

practical for deployed APIs and systems where full gradients are not accessible but probes are possible.

Validity handling: surrogate-generated perturbations are forced into $\mathcal{C}$ (the domain of valid tabular perturbations) and the refinement step respects all tabular validity constraints; surrogate training can incorporate monotone constraints or synthetic data augmentation to better mimic target behavior.

2.1.7 Cost- and Validity-Aware Crafting (COVAC)

COVAC, a black box attack, makes small but realistic edits. It respects valid ranges, one-hot groups and applies a cost. For example, changing transaction history is more than rounding the income. It generates valid human-plausible examples that show where the model is fragile.

Threat model: mixed (realistic attackers constrained by observability, audit, and feature-specific costs). Models adversaries who minimize the number and cost of feature edits.

Objective (cost minimization with sparsity):

$$\min_{\delta} \sum_{j} c_j 1\{\delta_j \neq 0\} \quad \text{s.t.} \quad f(x+\delta) \neq f(x),\ \|\delta\|_0 \leq k,\ x+\delta \in \mathcal{C},$$

with c_j the per-feature edit cost.

Description & execution: COVAC ranks features by influence gradients, SHAP (gives a per-example breakdown of how much each feature pushed the model's prediction up or down), or surrogate attribution and performs a bounded search for k-sparse edits (the attacker is allowed to change at most k features) that minimize a cost objective while preserving feasibility. It preferentially returns edits that are low-cost and plausible (e.g., adjusting rounding-safe numeric fields rather than identity fields). COVAC models attacker priorities in real financial systems and yields interpretable counterfactuals.

Validity handling: the search enforces domain constraints and consults feasibility checks (e.g., cross-field ratios); if no feasible k-sparse attack exists, COVAC gradually relaxes cost or swaps candidate features with constrained fallback rules.

2.1.8 Optimal Tree-Ensemble Evasion

Threat model: white-box with full access to ensemble structure (tree thresholds, leaf values). Targets tree-based models for exact or tightly bounded minimal edits.

MILP objective sketch:

$$\min_{x',z} \sum_j w_j |x'_j - x_j| \quad \text{s.t.} \quad z \text{ consistent leaf assignments, } x' \in \mathcal{C}, \text{ vote}(z) \text{ favors target,}$$

where integer variables z encode leaf choices and w_j are edit weights.

Description & execution: The attack encodes each tree's split constraints as linear/integer relations and uses a MILP solver to search for the minimal-cost perturbation that flips the ensemble majority. For large ensembles a leaf-tuple pruning heuristic enumerates promising leaf combinations to reduce complexity. The method yields provably optimal or lower-bound guarantees for required perturbation magnitudes.

Validity handling: the MILP includes explicit constraints for quantization, categorical one-hot integrity, monotonicity, and cross-feature relations; solver timeouts provide tight bounds when exact optimality is computationally infeasible.

2.2 Model Definition

2.2.1 CatBoost stacked XGBoost

CatBoost is good at handling categorical data, and XGBoost is a fast and powerful tree-based model with a different way of splitting data. They make fewer mistakes after stacking and so is more reliable and accurate. The two-stage stacked ensemble: A CatBoost base is trained with strict K-fold cross-validation Out-of-fold (OOF) discipline to produce robust out-of-fold scores; meaning that every observation from the original dataset has the chance to be included in the training and test sets. Those out-of-fold (OOF) scores are concatenated (optionally with a small set of standardized original features) and used as meta-features for an XGBoost meta-learner. The meta-learner captures residual structure not modeled by CatBoost and produces a final calibrated score. The stack is used where high predictive performance is needed while retaining a tree-based representation suitable for rule/audit extraction.

2.2.2 CatBoost stacked Logistic Regression

CatBoost produces high-quality out-of-fold (OOF) signals, which are passed to a linear logistic meta-learner. The logistic meta-layer provides interpretable weightings of CatBoost outputs and yields well-behaved probabilities that are easy to calibrate and audit. This pipeline favors transparency and corrects the confidence scores.

2.2.3 Artificial Neural Network (ANN)

It automatically learns complex, nonlinear relationships between different input features. ANNs are especially powerful when the dataset is large and the relationships among features are complicated. Here we standardise numerical values, convert categorical variables into one-hot encoding. Neural networks can sometimes be sensitive to noisy or misleading features, so they need careful preprocessing and regularisation to remain stable. subsectionC-Stack **Process**: CatBoost + Monotone-XGBoost → Logistic meta-learner → Temperature scaling

CatBoost is stacked with monotonic XGBoost to enforce better business logic. In this case, better repayment history should not mean less loan approval or vice versa. A stratified assembly designed for policy-sensitive credit scoring: CatBoost (for categorical/target-encoding strengths) together with a monotone-constrained XGBoost to enforce domain monotonicities, meaning some features should behave logically. It is trained with K-fold out-of-fold (OOF) discipline—for every training row, we used the prediction from a model that did not train on that row, getting "honest" predictions; the resulting out-of-fold (OOF) scores feed a logistic meta-learner. Finally, temperature scaling is fit on a validation fold (lowering negative log-likelihood (NLL)/Brier score) to calibrate probabilities used for thresholds and business cutoffs.

2.2.4 RAT-Tab

Process: FT-Transformer → TRADES + AWP

A transformer-style tabular architecture where each feature is embedded as a token and multi-head self-attention models feature interactions. RAT-Tab is adversarially fine-tuned with TRADES (margin-aware loss) and AWP (adversarial weight perturbation) to increase training-time robustness while retaining high capacity for complex temporal and transactional patterns.

2.2.5 Calibrated Monotone Ensemble (CME)

Process: Monotone-XGBoost → Isotonic calibration → Randomized Smoothing certificate

The CME model is designed to be trustworthy and well-calibrated. The Monotone XGBoost builds a decision tree model that obeys lending policies and logical business constraints that ensures the model's predictions remain interpretable and policy compliant. Randomised Smoothing adds small random noise to the inputs during prediction and observes how often the model's decision stays the same.

2.2.6 Monotone-LightGBM

Process: Monotone-LightGBM (adv-trained) + Isotonic calibration

LightGBM is a fast, memory-efficient gradient boosting model that builds an ensemble of small decision trees to achieve strong predictive performance. Monotone constraints ensure that the model behaves consistently with logical business rules. These constraints prevent unrealistic or counterintuitive predictions and make the model easier to justify in policy-driven domains. We also applied adversarial training, where slightly perturbed or "tricky" data samples are added during training. This process teaches the model to stay stable even when inputs change a little. Finally, isotonic calibration is used after training to correct the predicted probabilities of the model. This step ensures that when the model outputs a probability, such as 0.5, it truly corresponds to an 50% likelihood in real-world terms. Together, these techniques make Monotone-LightGBM accurate and reliable.

2.2.7 Explainable Boosting Machine (EBM)

EBM is a model that sums up simple and easy-to-read fragments. On each feature, it learns a one-dimensional curve showing the way that feature varies the prediction. It also learns small interaction effects of some significant pairs of features. We may impose some features such that they always drag the prediction towards a particular direction (monotonic links), such as more late payments cannot decrease risk. Due to this design, EBM remains readable and auditable and, at the same time, helpful for nonlinear trends.

Chapter 3

Data Collection and Analysis

3.1 Work Plan

In Figure 3.1 the step-by-step approximate processes that we are going to follow to conduct our research are presented.

The flowchart outlines the development of a robust, attack-resistant machine learning model for loan approval. It focuses on data collection, correction, and preprocessing to ensure clean data. After selecting and training the model, performance is evaluated, and adversarial training and additional defenses are applied to protect against manipulated inputs. The model undergoes adversarial attacks and robustness testing and is followed by real-world simulation to assess its resilience. Finally, impact analysis and outcome evaluation ensure the model's effectiveness in identifying fraud and securely approving loans, while resisting adversarial interference.

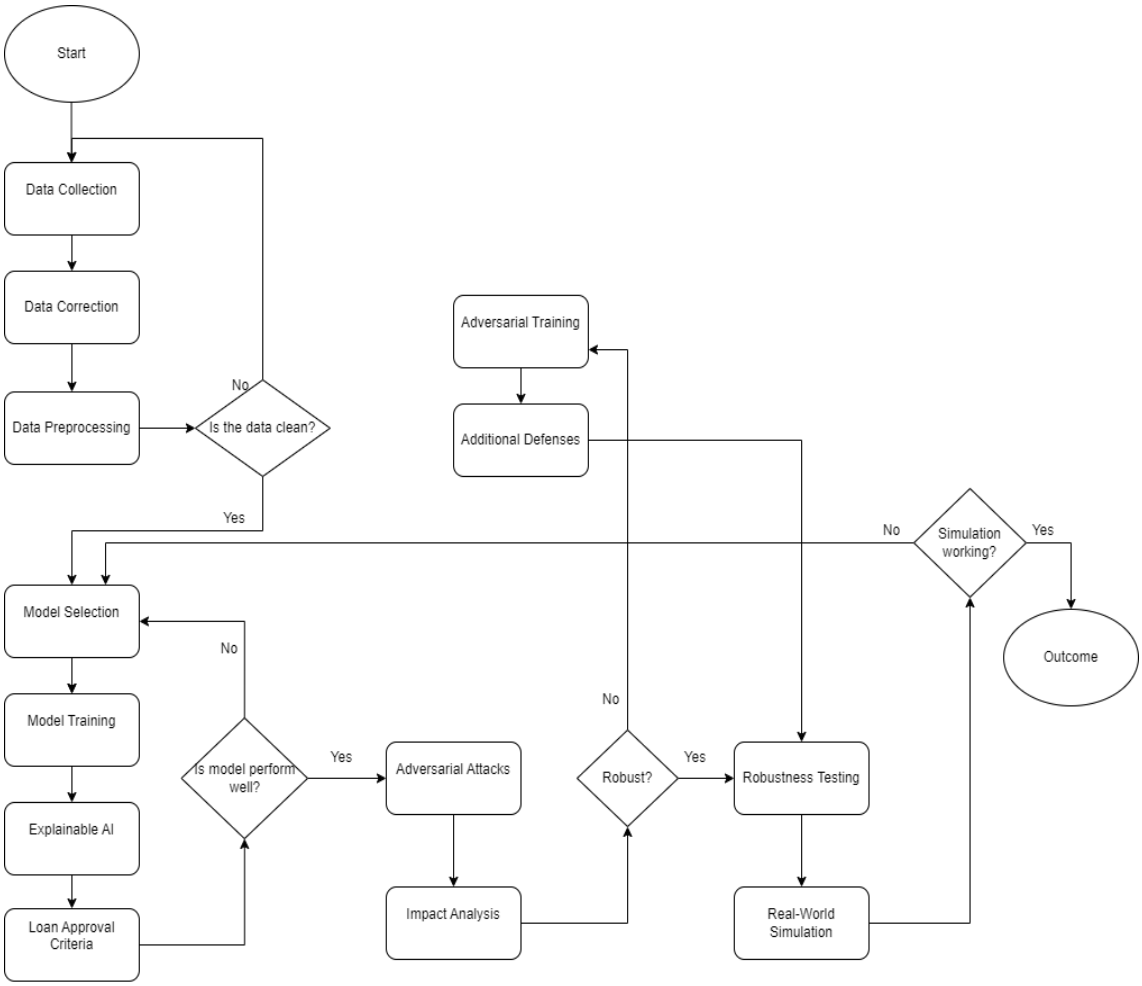


Figure 3.1: Work Plan

3.1.1 Data Collection and Synthetic Data Generation

In order to address the small size constraint (2,000 samples) of the real-world data collected from a microfinance institution and to provide sufficient diversity for model training, we generated 80,000 synthetic records with Conditional Tabular GANs (CTGANs) as proposed by Xu et al. (2019)[16]. CTGANs are a type of deep generative model designed for mixed-type tabular data with both numerical and categorical variables, and they capture nonlinear relationships and multimodal feature distributions. The entire data was used to build and test machine learning models, which enables it to assess the behavioral, financial, and social characteristics that lead to either the granting of a loan application (label = 1) or rejection (label = 0).

3.1.2 Data Preprocessing and Synthetic Data Balancing

Raw transactional and demographic data preprocessing and cleaning are important processes used to ensure strong, high-performance, and generalizable machine

learning and deep learning models. Unprocessed finance data is usually lacking in consistency or duplicated features and may have different data formats that need to be standardized before training prediction models.

Moreover, in this study, CTGAN was trained on the dataset that had been real and de-identified but preprocessed and had the loan approval label set as conditioning. The condition on the loan approval label allows the model to learn class-conditional distributions, which hopefully allows it to directly combat the data imbalance between approved and denied loans. All reasonably numerical features were min–max scaled, and categorical features were one-hot encoded before training. Once the model was converged, 80,000 synthetic samples were generated, which were sampled to create realistic, transnational-like profiles for borrowers.

The dataset `Loan_Dataset.csv` was processed through a multi-step pipeline using Python libraries such as `pandas`, `scikit-learn`, `seaborn`, and `matplotlib`. The complete process is as follows:

- **Initial Audit:** Loaded dataset, checked structure, and confirmed balanced `target_loan_flag` (50/50 accepted/denied loans).
- **Missing Values:** Removed rows with null values, retaining 80,149 complete records.
- **Redundant Features:** Dropped six collinear monthly expenditure columns (Jan–Jun) based on correlation analysis, keeping `avg_txn`.
- **Categorical Encoding:** Applied label encoding to occupation and region to avoid sparse matrices and retain ordinality.
- **Outlier Detection:** Used histograms and boxplots to verify acceptable skewness in numeric features (`account_age_months`, `account_balance`, `avg_txn`, `std_txn`, `buddy_ratio`).
- **Feature Scaling:** Applied Min-Max normalization to scale all features to [0, 1] for equal contribution and faster training.
- **Dataset Split:** Stratified split into 70% training (56,104 rows) and 30% testing (24,045 rows) to maintain class balance.

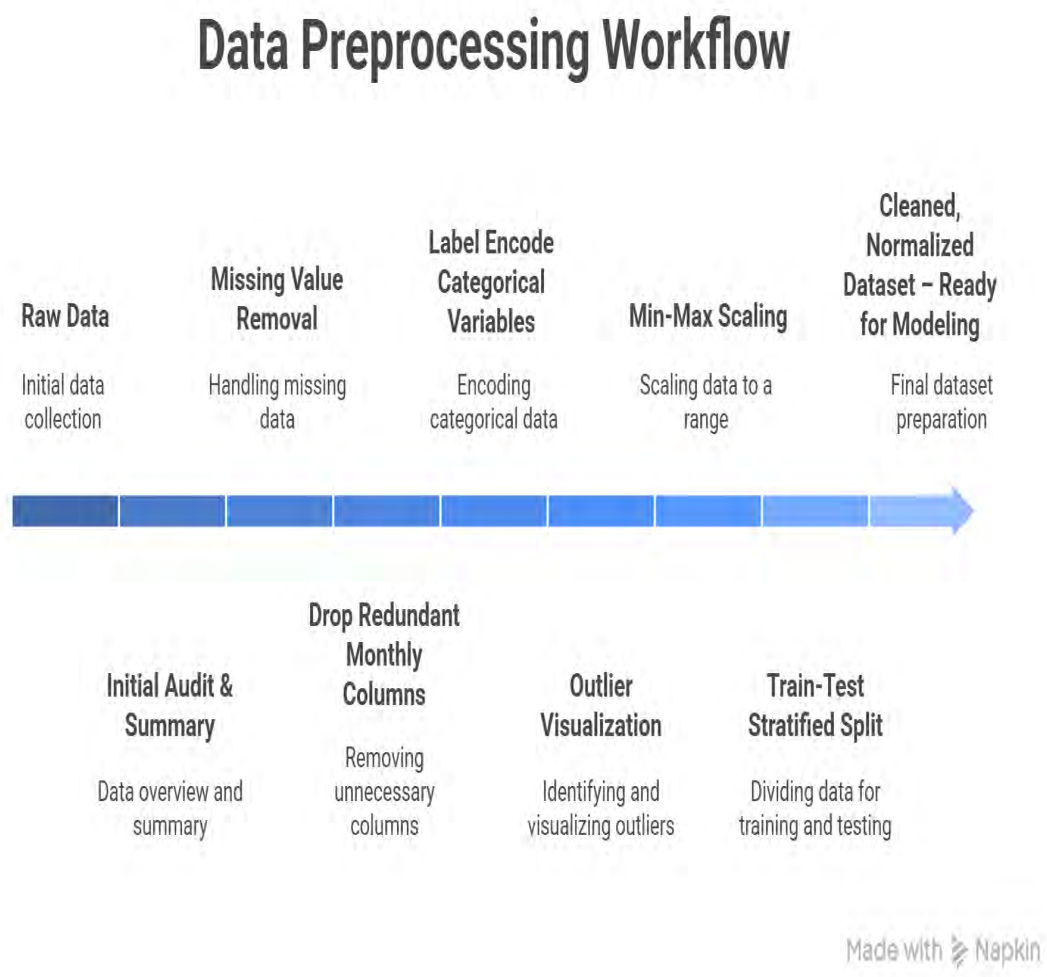


Figure 3.2: Data Pre-processing and Cleaning Workflow

Figure 3.2 shows the defined pipeline. This pipeline ensures clean, consistent data for high-performance, generalizable models.

3.1.3 Data Analysis

For reliability and compliance, the data generation was based on an approved method: the real seed data were used for model training only and under a signed data-use agreement, and the synthetic data were verified using statistical similarity and utility (model performance on real hold-out data) checks. This method makes sure that the enhanced dataset maintains the original data's statistical structure while managing balance and privacy, and is ready for subsequent analyses like model training and adversarial robustness.
Multifaceted and rigorous data analysis was followed to generate actionable insights with the help of the pre-processed data and in order to build robust prediction

models of loan approvals. This step involved the use of descriptive statistics, visual exploration, correlation diagnostic, and feature-target relationship checks to make sure that input variables have meaning as well as being relevant to follow-on machine-learning and deep-learning processes.
This is almost balanced, and this reduces the possibility of class imbalance bias as well as enables the predictive algorithms to learn the decision edges of both classes.

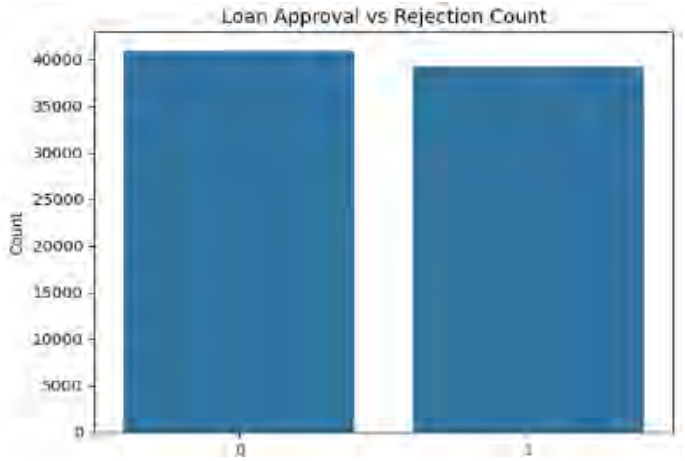


Figure 3.3: Loan Status (0 = Rejected, 1 = Approved)

Diagnostics of the Correlation

A Pearson correlation matrix was calculated to measure linear relationships between input features and the target variable and it was plotted as a heatmap. Among the main findings, there is:

- **Highest Positive Correlation:** The strongest positive linear correlation is with loan approval (0.41) for `loan_buddy_count`, indicating that a larger social network of lending users increases the likelihood of loan approval, suggesting creditworthiness.
- **Social Trust Indicators:** Variables `buddy_score_flag` and `buddy_ratio` are also positively correlated (0.33 and 0.32, respectively), demonstrating the effectiveness of social trust indicators in microfinance conditions.
- **Seasoned Financial Variables:** Features like `account_balance` and `avg_txn` show moderate positive correlations, implying that increased financial activity and account holdings are better indicators of probable loan approval.
- **Negative Relationship:** `late_payments` exhibits a weak negative relationship, which aligns with financial logic, as repeated late payments reduce the probability of loan approval.

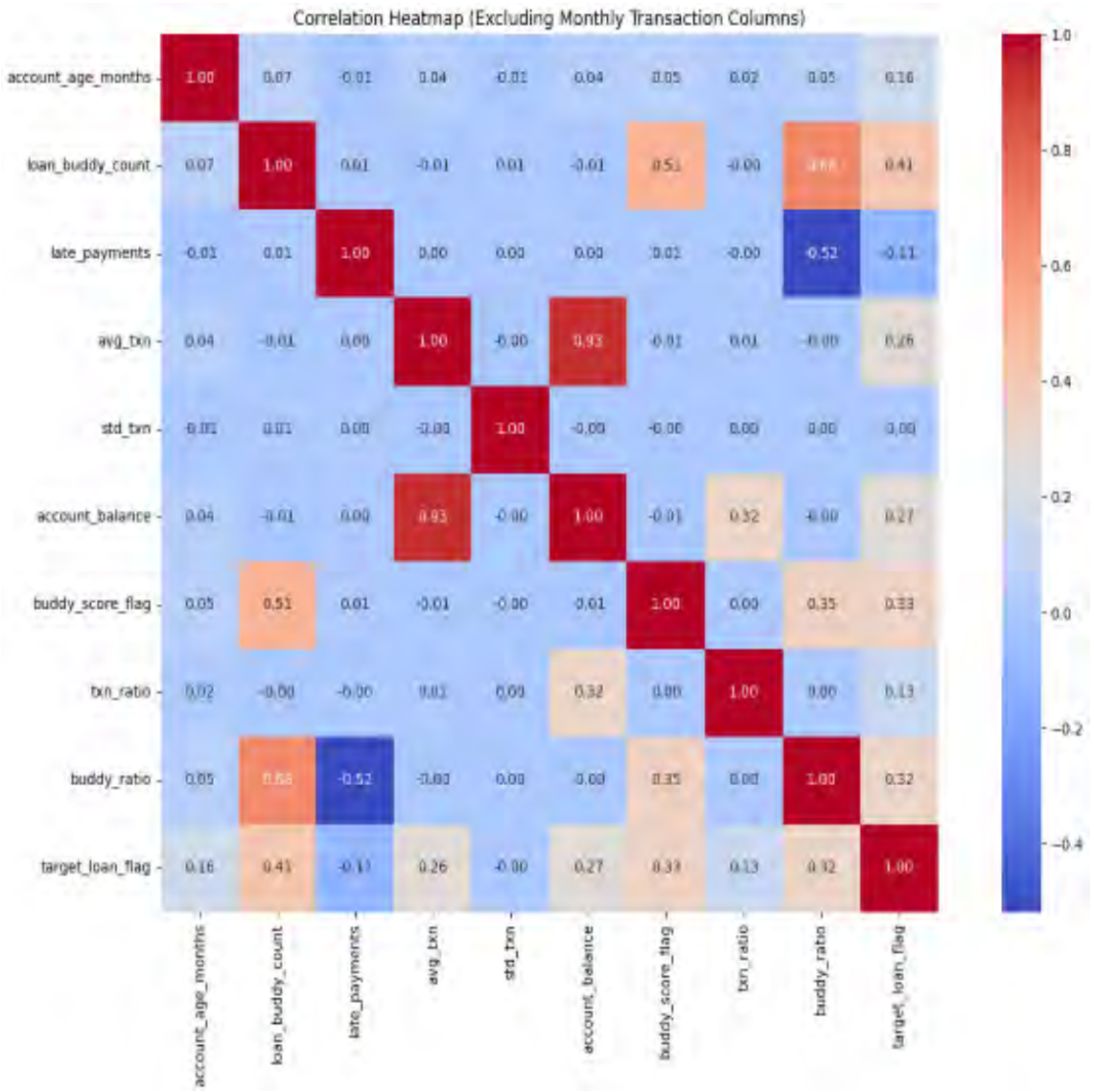


Figure 3.4: Heatmap

Figure 3.4 The correlation matrix gave an initial idea about the relevance of features, enabling to simplify the number of variables that need more attention during training the model and ranking the feature importance.

Mutual Information Analysis

Since it can be possible that financial behaviors are characterized with non-linear associations, it becomes necessary that further analysis should integrate both linear and complex interdependencies between every predictor and the target by computing Mutual Information (MI) scores. This is because the MI ranking confirmed the relevance of `loan_buddy_count`, `buddy_score_flag`, and `buddy_ratio` to be the highest contributing factors. Unlike simple correlation, MI is able to identify disguised and non-linear relationships, which allows a more subtle view of the importance of variables.

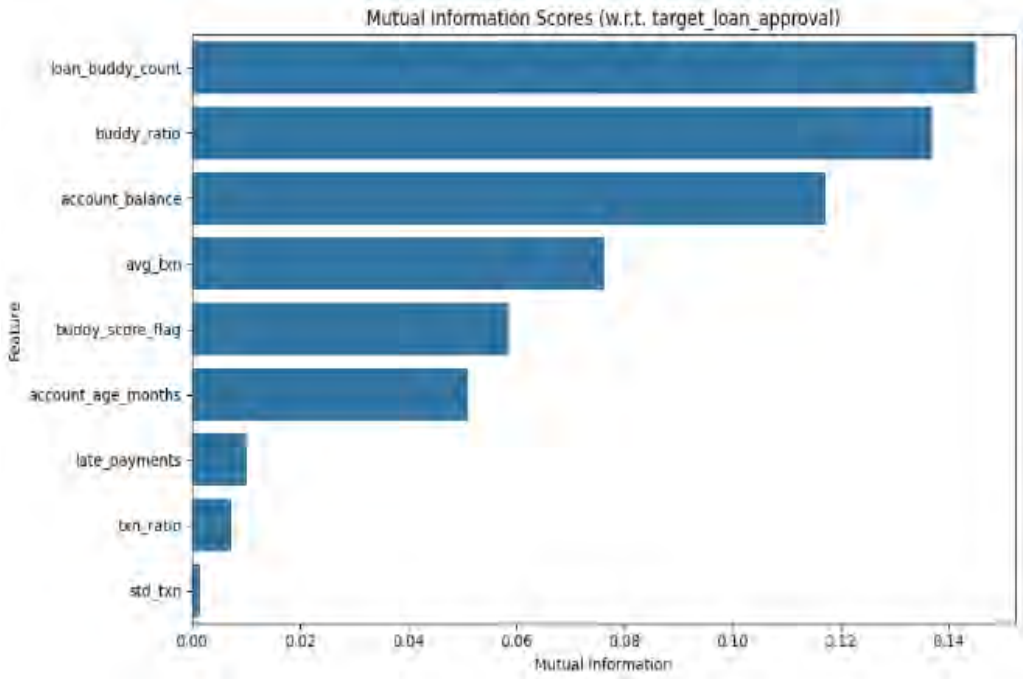


Figure 3.5: Mutual Information Scores

Figure 3.5 The key numeric variables were plotted in histograms with the Kernel Density Estimates (KDEs) to analyze their individual distributions of values and possible outliers

Univariate Feature Distributions

The key numeric variables were plotted in histograms with the Kernel Density Estimates (KDEs) to analyze their individual distributions of values and possible outliers:

With such distribution checks, it is made sure that, feature scaling did not cover other important data attributes and that no additional transformation was required.

Feature-Target Interactions

In order to better establish the divergence between the features due to the outcome of the loan response (Rejected vs. Approved), boxplots were constructed between the important numeric variables in the two classes (0 = Rejected, 1 = Approved).

Observations include:

- Larger average balances are associated with approved loans.
- Lower standard deviation in transaction patterns (std_txn) correlates with

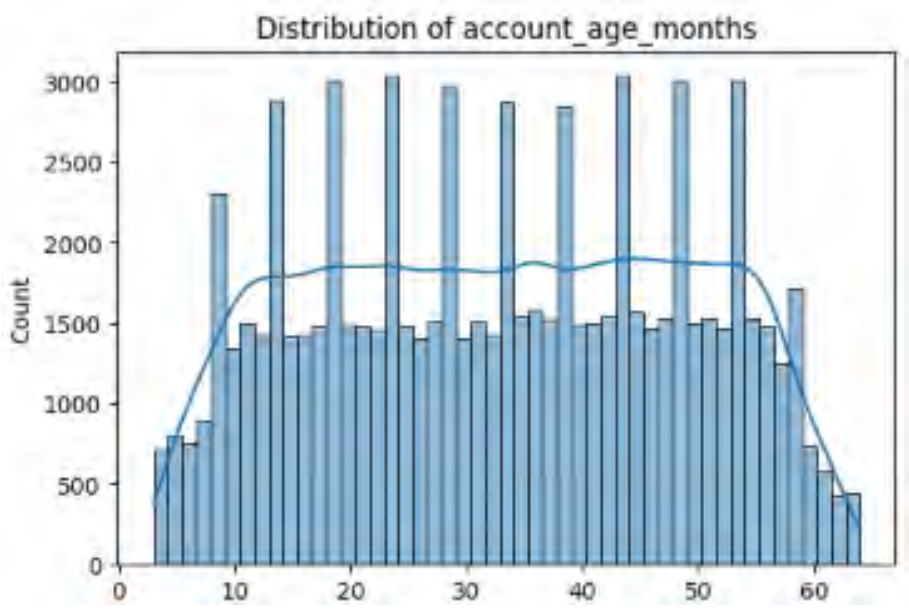


Figure 3.6: Distribution of account_age_months

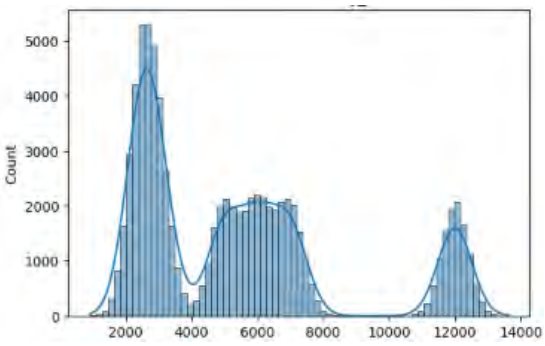


Figure 3.7: Distribution of average txn

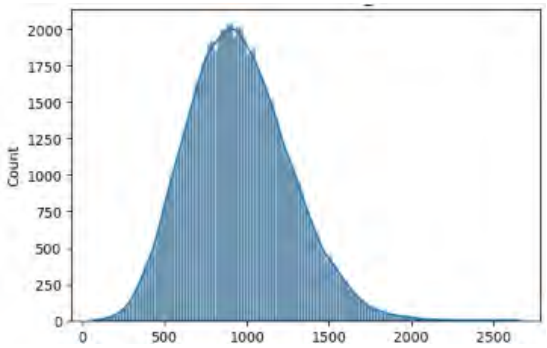


Figure 3.8: Distribution of std txn

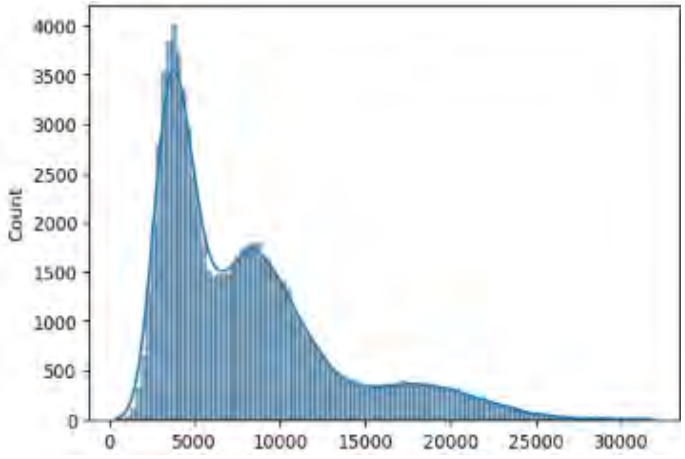


Figure 3.9: Distribution of account balance

higher approval probability.

- Decrease in late payments is one of the factors that raise the chances of a positive decision.

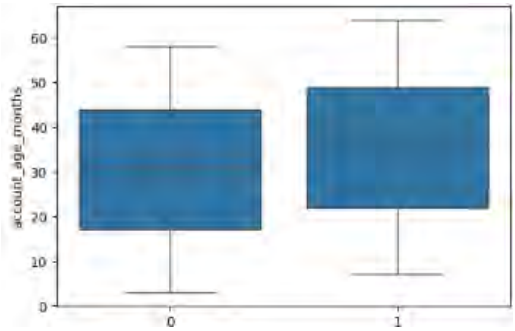


Figure 3.10: account_age_months vs Loan Status

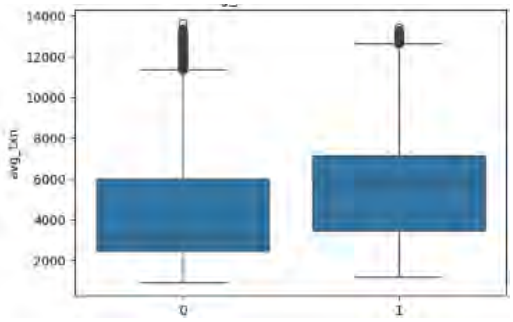


Figure 3.11: Average Txn vs Loan Status

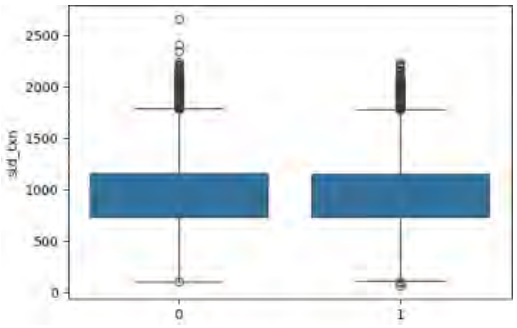


Figure 3.12: std_txn vs Loan Status

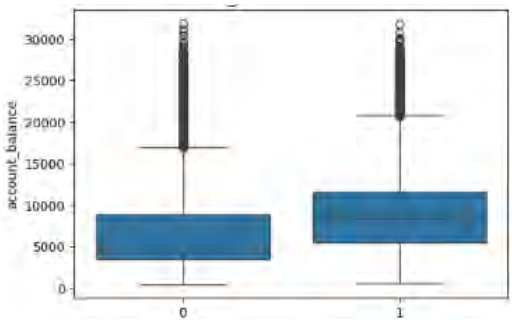


Figure 3.13: account_balance vs Loan Status

Categorical Feature Analysis

features like occupation and region were analyzed through count plots that were stratified by the target variable. The analysis revealed the subtle demographic changes such as:

- Occupations like working as a salaried employee, being an independent employed, and so on have slightly higher rates of approval comparing to the groups of unemployed or students.

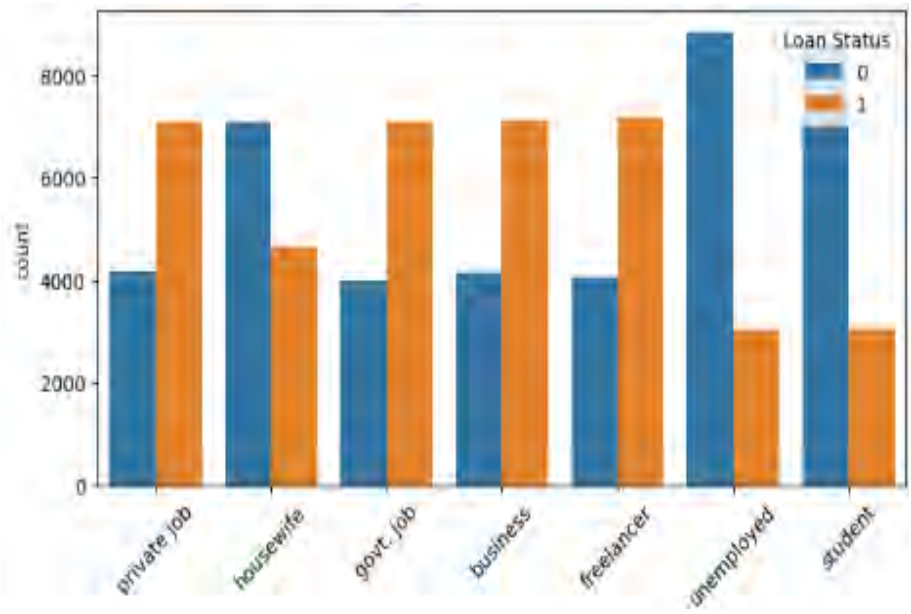


Figure 3.14: Occupation vs Loan Status

- There are regional patterns that show the subliminal differences between the urban and the rural areas in the approval behaviors.

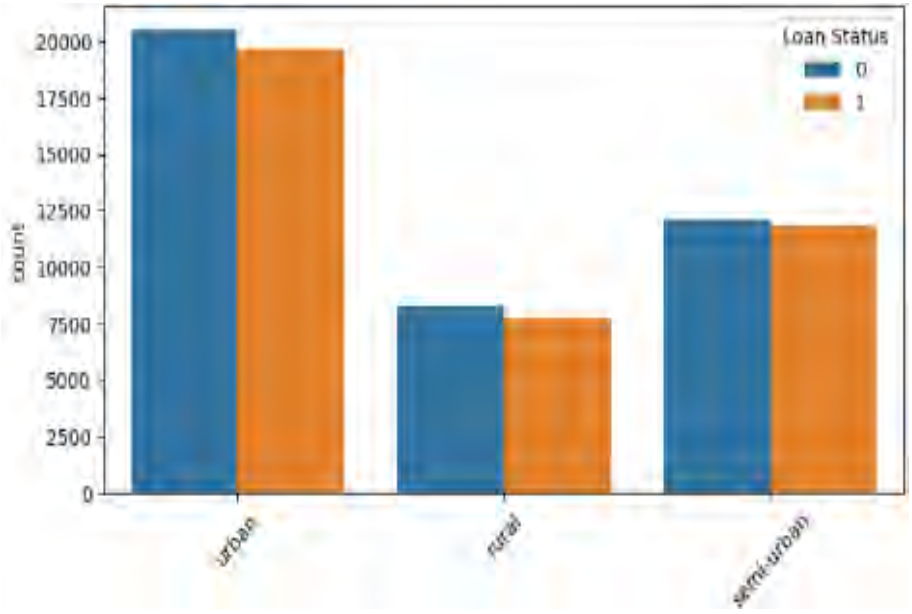


Figure 3.15: Region vs Loan Status

Chapter 4

Model, Attack and Defense Architectures

4.1 Model Architectures

4.1.1 CatBoost stacked XGBoost

CatBoost(great with categorical/target statistics) is combined with XGBoost(gradient-boosted trees) in a two-stage stack. The out-of-fold results of CatBoost are used as meta-features for an XGBoost meta-learner. The objective is to find the complementary patterns (high-cardinality categoricals + residual nonlinear structure) and at the same time keep tabular validity and deliver high AUC/F1.

Pipeline:

- Train CatBoost (strict K-fold OOF) to generate base scores.
- Concatenate OOF(Out-of-Fold) scores with selected stable original features.
- Train XGBoost meta-learner on OOF matrix.
- Refit CatBoost; score validation/test; select operating threshold by F1 or policy curve.

Design notes:

- Complementarity between categorical strength and residual learning.
- Robust OOF(Out-of-Fold) discipline prevents leakage.
- Calibration via temperature or isotonic improves stability.

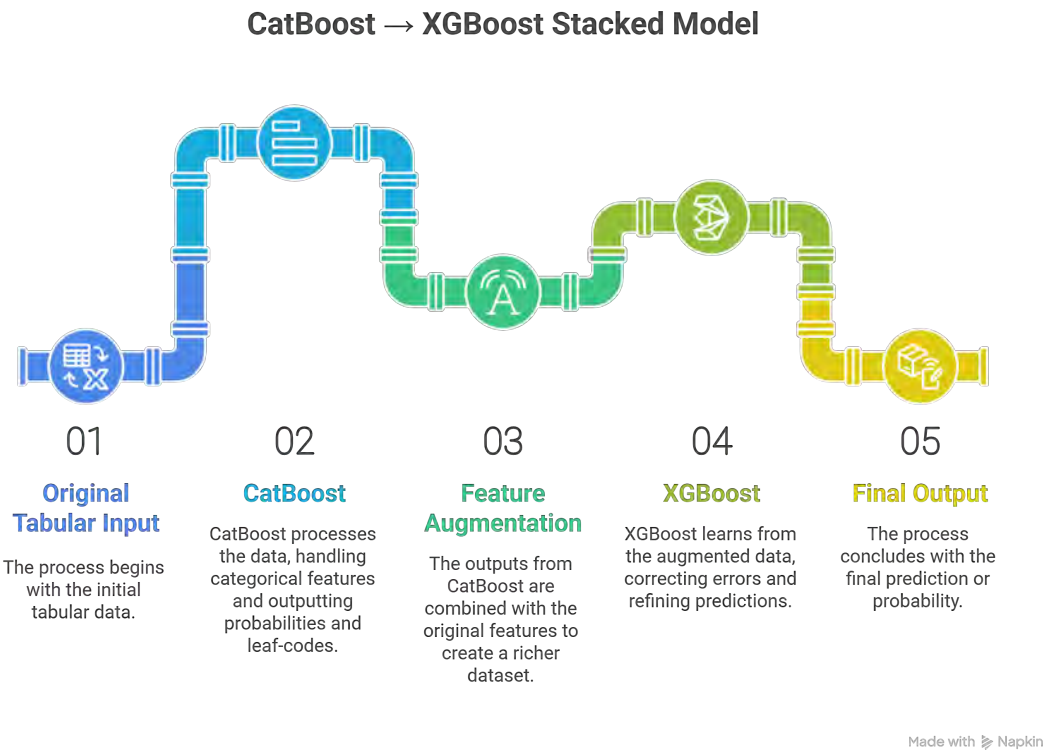


Figure 4.1: CatBoost to XGBoost stacked model workflow

4.1.2 CatBoost stacked Logistic Regression

CatBoost OOF scores feed a simple logistic regression meta-learner, producing interpretable linear weightings with calibrated outputs.

Pipeline:

- Train CatBoost with K-fold OOF; train L2-regularized Logistic Regression.
- Fit temperature T to minimize NLL/Brier.
- Fix threshold on validation and report test metrics.

Design notes:

- Transparent coefficients align with expected risk direction.
- Lightweight inference and reproducible calibration.

4.1.3 Artificial Neural Network (ANN)

Type: feed-forward MLP
A compact multilayer perceptron used for gradient-based attacks (PGD, JSMA, AutoAttack). Standardized numeric inputs, embedded categoricals.

Pipeline:

- Standardize numerics, encode categoricals (embedding or one-hot).
- Input → [Dense → BatchNorm → ReLU → Dropout] × L → Sigmoid.

  BatchNorm (Batch Normalization) normalises activations in a mini-batch to have stable mean/variance, then learns a scale and shift. This speeds up training, stabilizes gradients, and often lets you use larger learning rates.

  ReLU (Rectified Linear Unit). The ReLU nonlinearity is

  $$\mathrm{ReLU}(x) = \max(0, x) = \begin{cases} 0, & x < 0, \\ x, & x \geq 0. \end{cases}$$

  It passes unchanged positive values and zeroes out negatives lets the network learn richer patterns. Its derivative is $\mathrm{ReLU}'(x) = \mathbf{1}\{x > 0\}$ (with a conventional choice at $x = 0$).

  Dropout (regularization). During training, randomly "drops" (sets to 0) a fraction of activations to prevent co-adaptation. This reduces overfitting

  L (repeated block / depth). We have L hidden layers with a sequence. Increasing L adds capacity.

  Sigmoid (logistic output). Final activation

  $$\sigma(z) = \frac{1}{1 + e^{-z}},$$

  that squashes outputs to (0,1). Perfect for binary classification because it can be read as a probability.

- Train with binary cross-entropy + Adam; early stopping on F1/AUC.

Design notes:

- Regularized with Dropout, BatchNorm, Weight decay.
- TRADES/AWP fine-tuning enhances robustness to PGD-style attacks.
  - TRADES (TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimization).
  - AWP (Adversarial Weight Perturbation). During training, it briefly nudges the model's weights in the "worst" nearby direction before computing the loss.

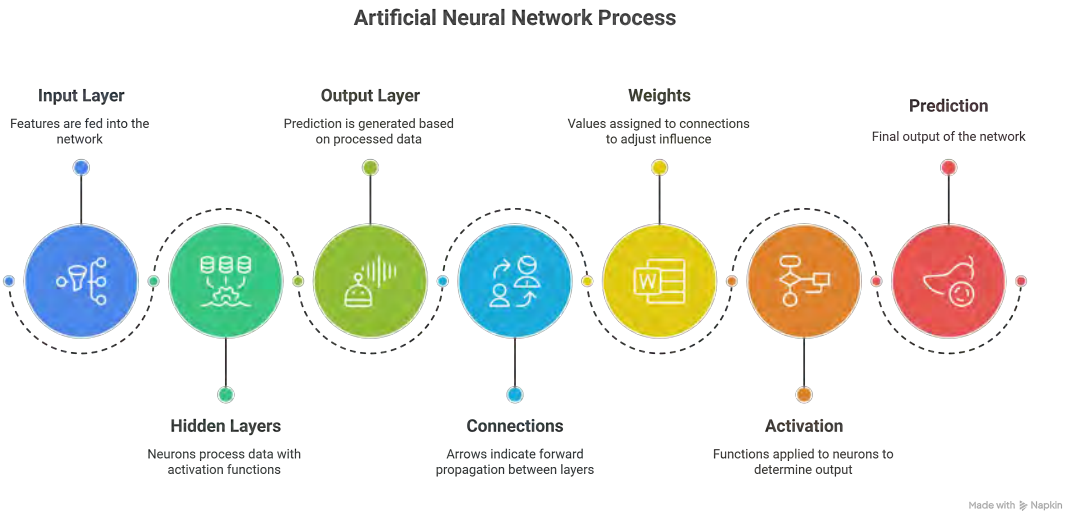


Figure 4.2: Artificial Neural Network Workflow

4.1.4 C-Stack

Process: CatBoost + Monotone-XGBoost + Logistic meta + Temp scaling
Combines CatBoost and monotone XGBoost outputs into a logistic meta-learner calibrated by temperature scaling.

Pipeline:

1. Train CatBoost and Monotone-XGB (K-fold OOF).
2. Train Logistic Regression on OOF (Out-of-Fold) scores.
3. Fit temperature T on validation; report fixed threshold.

Design notes:

- Monotonicity aligns with policy priors.
- Calibration improves ECE/Brier without altering rank.

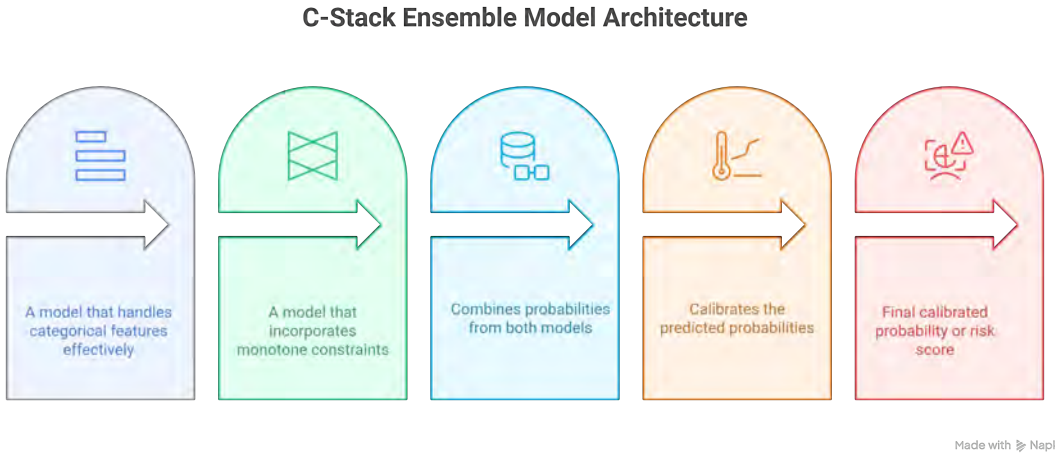


Figure 4.3: C-Stack Workflow

4.1.5 RAT-Tab

Process: FT-Transformer with TRADES and AWP
Transformer-style tabular architecture with adversarial fine-tuning (TRADES + AWP).

Pipeline:

- Pretrain FT-Transformer on clean data (K-fold OOF).

  FT-Transformer = Feature Tokenizer Transformer. It turns each tabular feature into a learnable embedding "token" and runs the token sequence through Transformer encoder layers.

- TRADES fine-tune: optimize CE + β KL$(f(x), f(x_{\text{adv}}))$.
- AWP(Adversarial Weight Perturbation) perturb weights during update, project back to ϵ_w ball.

Design notes:

- TRADES parameter β balances accuracy–robustness.
- AWP regularizes sharp minima; avoids overfitting.

4.1.6 Calibrated Monotone Ensemble (CME)

Process: Monotone-XGBoost → Isotonic calibration → Randomized smoothing
Combines monotonic policy-aligned boosting with calibrated probabilities and cer-

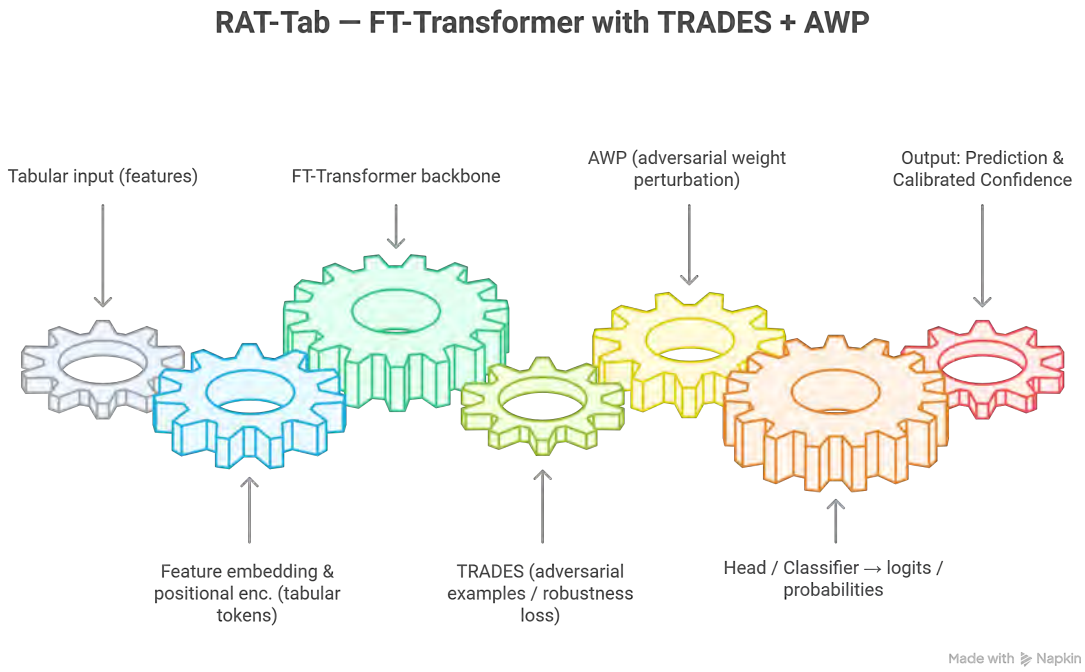


Figure 4.4: RAT-Tab Workflow

tified robustness.

Pipeline:

- Train monotone XGBoost with policy-signed features.
- Fit isotonic calibration on OOF scores.
- Apply randomized smoothing: sample K perturbed copies and average predictions.

Design notes:

- Monotonicity ensures stable policy alignment.
- Smoothing yields certified robustness with small computational overhead.

4.1.7 Monotone-LightGBM

Process: adversarially trained + Isotonic calibration
Fast LightGBM with monotone constraints and adversarial augmentation.

Pipeline:

- Train LightGBM with monotone signs (OOF protocol).
- Generate adversarial perturbations (bounded L_∞/L_0 per feature).
- Mix clean/adversarial samples; isotonic calibration on OOF(Out-of-Fold) predictions.

Design notes:

- Tree-friendly adversaries improve generalization.
- Monotonicity and calibration yield transparent thresholds.

4.1.8 Explainable Boosting Machine (EBM)

Explainable Boosting Machine (GA^2M with monotonic links): additive model with per-feature shape functions and sparse interactions.

Pipeline:

- Train EBM on OOF folds, learning $f_j(x_j)$ and $f_{jk}(x_j, x_k)$.
- Enforce monotone constraints; validate sparsity and smoothness.
- Apply isotonic/temperature calibration; fix threshold.

Design notes:

- Fully interpretable additive structure.
- Monotonic links ensure predictable policy behavior.
- Competitive accuracy with transparent governance.

4.2 Attack Architectures

4.2.1 HopSkipJump (decision-based, black-box boundary search)

HSJ is a label-only attack that (1) finds a nearby boundary point via binary search and (2) estimates a normal direction to that boundary using randomized queries, then (3) "hops" toward the boundary and "skips" with adaptive step sizes to reduce

queries. For tabular data we round back to valid categories/bins after every move and enforce policy/semantic constraints at projection time.

Pipeline

- Initialize with any opposite-class seed (or via random flip set respecting domain).
- Repeatedly: boundary binary-search → gradient-sign/direction estimate with label queries → step toward decision boundary.
- Feasibility projection each step: clamp numerics to ranges; snap integers; flip one-hot groups coherently; reject invalid combos.
- Early stop on first feasible misclassification with minimal distance (keep best over the run).

Design notes

- Query-budgeted: cap per-point queries; reuse direction estimates across small neighborhoods.
- Stability: average decisions under allowed input jitter if the model uses smoothing/DP noise (PopSkipJump variant handles probabilistic outputs).

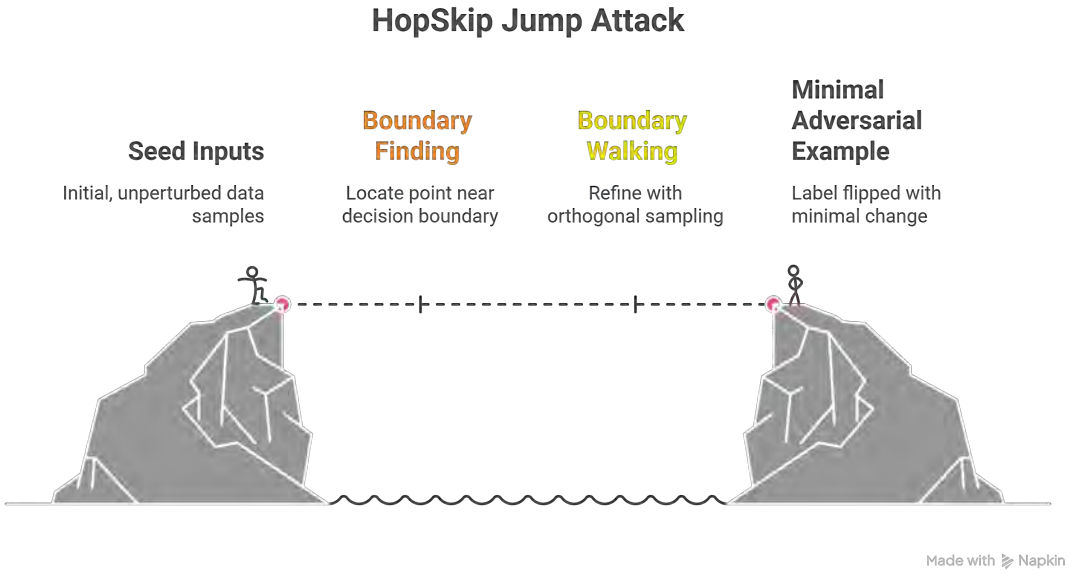


Figure 4.5: HopSkipJump Attack workflow (boundary search and projection steps).

4.2.2 PGD (white-box; projected first-order)

Standard iterative first-order maximization of a surrogate loss under a norm/feasibility set, with projection after every step. For tabular: mask immutable features, handle categoricals via discrete flips (enumeration or straight-through relaxations), and project to a constraint polytope (ranges, monotone signs, business rules).

Pipeline

- Choose threat set (e.g., bounded per-feature L_∞/L_2 + validity rules).
- Initialize within set (random start). Iterate:

$$x \leftarrow \Pi_{\mathcal{S}}\left(x + \alpha \operatorname{sign}\left(\nabla_x \mathcal{L}\right)\right)$$

- Mixed domains: gradient step on relaxed copy; discretize: majority flip for one-hot, argmax for embeddings, round integers.

Design notes

- Step schedule: cosine/halve on non-improvement; 5–20 restarts for non-convex losses.
- Cost-aware: weight features by edit cost before projection to keep perturbations realistic.
- Monotone-safe: forbid moves violating signed monotonicity constraints.

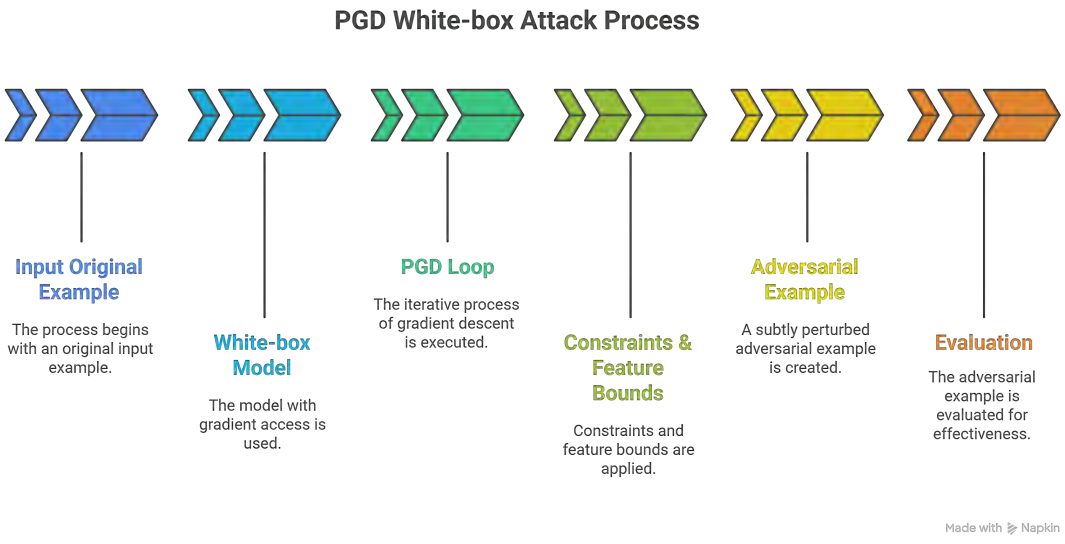


Figure 4.6: PGD Attack Workflow

4.2.3 JSMA (L_0-sparse saliency; targeted/untargeted)

JSMA uses the input–output Jacobian to rank features (or feature pairs) by how much they drive the target logit, editing as few coordinates as possible. On tabular data, edits are selected by best saliency-per-cost ratio and applied as discrete flips consistent with categories and integer bins.

Pipeline

- Compute saliency map S_j (or pairwise S_{ij}) for current class/target.
- Iteratively select top feature(s) improving target margin without violating constraints; apply minimal discrete edit (flip/bucket step).
- Recompute saliency with updated point; stop on misclassification or budget exhaustion.

Design notes

- L_0 budget first: minimize number of changed fields (business realism).
- Pairwise variant for strong feature interactions common in tabular.
- Feasibility guards: one-hot exclusivity; correlated fields (e.g., DTI vs. income) updated atomically.

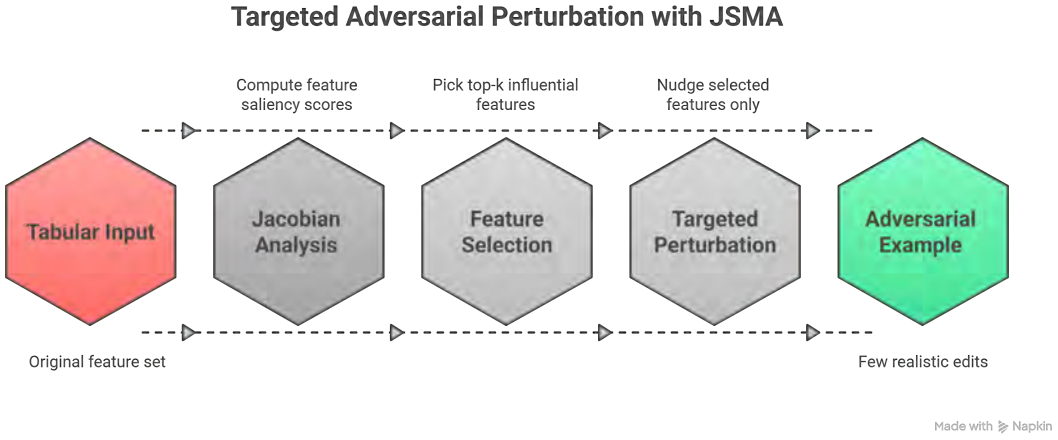


Figure 4.7: JSMA Attack Workflow

4.2.4 AutoAttack-Tabular (worst-of APGD + FAB + Square)

A robust evaluation suite: run strong, diverse attacks and keep the worst-case result per example. APGD, FAB, and Square are adapted to tabular feasibility, reporting the highest-loss feasible adversarial example.

Pipeline

- Configure tabular projector (ranges, integer snaps, one-hot, policy rules).
- **APGD (Auto Projected Gradient Descent)** powerful gradient-based method, tuned for robustness evaluation. Run APGD-CE with masked gradients to project each step.
- Run **FAB (Fast Adaptive Boundary)** with tabular box-projection; accept minimally distorted feasible solution.
- Run Square Attack with feature-wise blocks respecting discrete features.
- Return adversarial example with maximal loss / minimal distance.

Design notes

- Diversity: first-order + boundary + score-based covers gradient masking/obfuscation.
- Deterministic evaluation (fixed seeds) for reproducibility.
- Efficiency: early stop after success; cap per-attack compute.

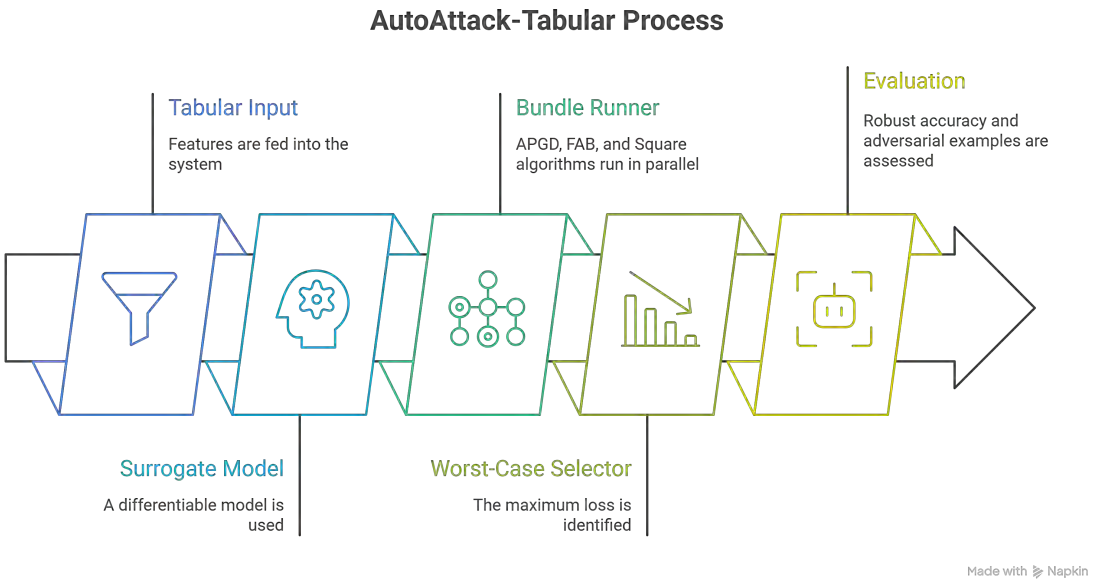


Figure 4.8: AutoAttack Workflow

4.2.5 CAA / CAPGD Cascade (constraint-aware PGD + search)

Constraint-Aware PGD (CAPGD) exploits gradients in the feasible region; Constrained Adaptive Attack (CAA) explores discrete neighborhoods unreachable by gradients.

Pipeline

- **Phase 1 (CAPGD):** masked/weighted PGD with projections enforcing validity and monotone signs.
- **Phase 2 (CAA):** neighborhood exploration over valid discrete edits (k-flips / bin steps) with adaptive step sizes.
- Budget schedule: gradually expand L_0 (edit count) and per-feature limits until success or cap.

Design notes

- Guarantees feasibility at every iterate (no off-manifold artifacts).
- Works for differentiable or non-differentiable models.
- Supports per-feature costs and grouped edits (e.g., financial variables).

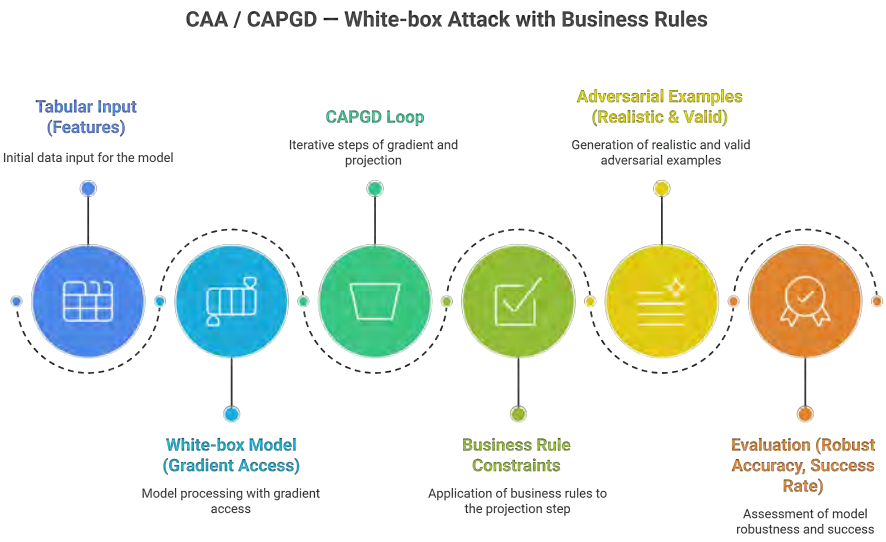


Figure 4.9: CAA/CAPGD Attack Workflow

4.2.6 Surrogate → Decision Hybrid (transfer + hard-label refinement)

When the target model is non-differentiable (GBMs, rule lists) or label-only, a differentiable surrogate is fit and refined against the true model using HSJ-style boundary search.

Pipeline

- Train surrogate to mimic target surface (with monotone links if needed).
- Generate candidate x' on surrogate (PGD/CAA-style).
- Refine via label-only boundary search (HSJ) around x' within the valid domain.
- Select minimal-cost successful adversarial point.

Design notes

- Transferability seed drastically reduces label queries.
- Validity-first: all edits satisfy business constraints.
- Robust under randomized inference: averages label queries in small jitter balls.

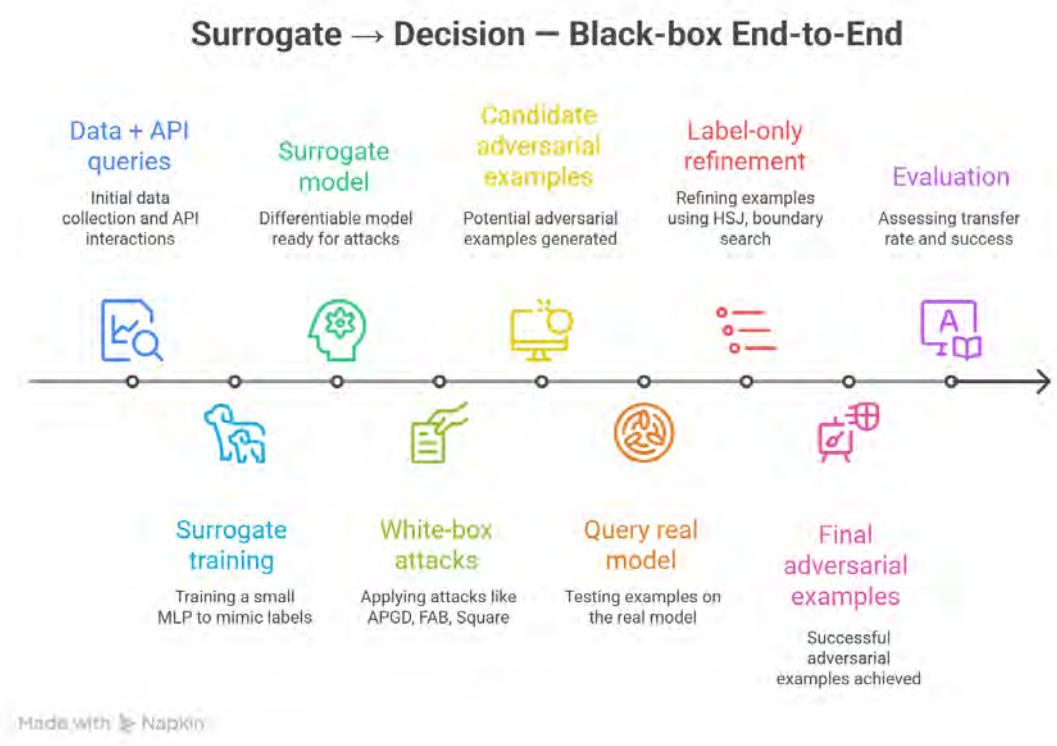


Figure 4.10: Surrogate → Decision Hybrid Workflow

4.2.7 COVAC (cost- and validity-aware crafting)

COVAC treats adversarial generation as constrained cost minimization, mixing penalty-method optimization (for numerics) with discrete local search (for categoricals).

Pipeline

- Define cost vector and hard constraints (ranges, types, monotone signs).
- Alternate solver:
  - Numeric block: projected gradient/coordinate descent with augmented Lagrangian penalties.
  - Discrete block: best-improvement flip or beam search within edit budget.
- Stop on minimal-cost feasible adversarial or certify infeasibility.

Design notes

- Business-aligned: edit costs reflect realism (e.g., income change $\gg$ rounding utilization).
- Provides infeasibility certificates if no valid attack exists.
- Works as a drop-in head for AutoAttack-Tabular or CAA cascade.

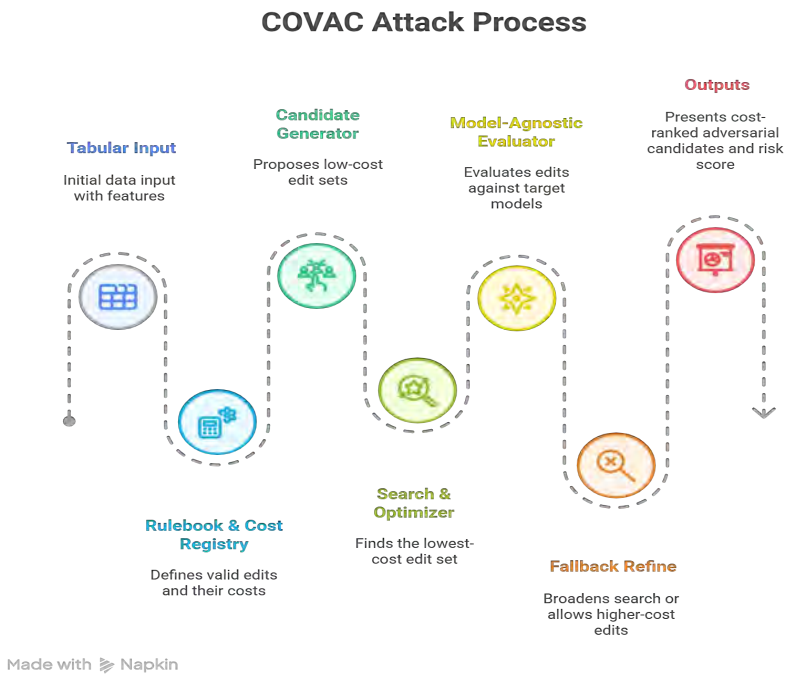


Figure 4.11: COVAC Workflow

4.2.8 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple search)

Two complementary formulations for boosted trees/forests: an exact MILP(Mixed-Integer Linear Programming) that encodes splits and leaves to minimize perturbation, and a fast leaf-tuple discrete search that flips ensemble output with minimal change.

Pipeline

- **MILP branch:** formulate binary variables for splits/leaves; enforce path consistency; minimize weighted L_1/L_∞ cost.
- **Leaf-Tuple branch:** initialize from current leaves; iteratively flip one leaf at a time; map to feasible input edit.

Design notes

- MILP yields optimality certificates for small ensembles; leaf-tuple scales to large forests.
- Feasibility built-in; interpretable for governance and model testing.

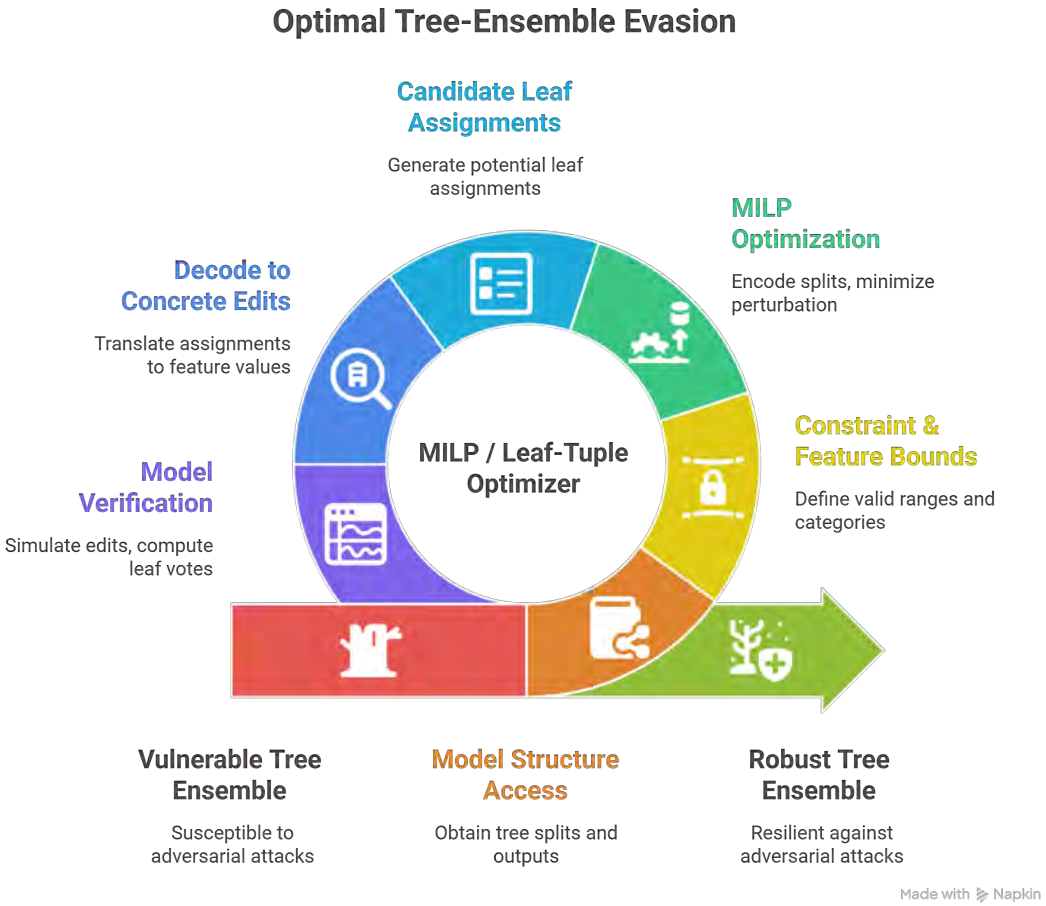


Figure 4.12: Optimal Tree-Ensemble Evasion Workflow

4.3 Defense Architecture and Workflow

4.3.1 Overview

We trained ANN with a combination of clean and CAPGD (Constraint-Aware Projected Gradient Descent)-generated adversarial examples (projected to the valid domain at each stage), we assessed the robustness of both CAPGD(S) and PGD-U. The system integrates adversarial training with TRADES (robustness–accuracy trade-off) + Adversarial Weight Perturbation (AWP) as the outer optimizer and CAPGD as the inner adversary. We combined TRADES with AWP as plain adversarial training can degrade clean accuracy or get stuck in brittle, sharp solutions. TRADES balances the trade-off: it requests the model to be consistent on clean data and accurate on adversarial twin to smooth out the decision boundary. Then AWP does 'shaking' of the weights very slightly in the direction that keeps it worst at each step, so we arrive at a flatter minimum that generalizes well and does not overfit. It gives rise to higher robust accuracy and stabler training as well as more calibratable probabilities when used together. Furthermore, to ensure realistic adversarial samples, we created CAPGD adversarial examples in each training batch and projected

them back to the valid set (snap categoricals to valid one-hot; clamp continuous features within the per-feature ℓ_∞ budgets) before updating weights. We also evaluate with PGD-U (unconstrained) to demonstrate an upper-bound on white-box vulnerability as a stress-test only. We report robust metrics (Accuracy, F1, AUC, Brier, Negative Log-Likelihood (NLL)) against CAPGD(S) and PGD-U, and we perform robust early stopping on adversarial validation loss. In order to restore probability quality under attack, we lastly calibrate using temperature scaling.

4.3.2 Threat Model (S)

Our mixed discrete–continuous threat set S reflects real credit-risk data:

- **Continuous attributes** (ratios, incomes): per-feature ℓ_∞ budget $\epsilon = 0.03$, clipped to valid ranges. Here, ℓ_∞ budget ϵ denotes the size of the largest single-feature change across all features. So, each normalized numeric feature can move at most ± 0.03.
- **Categorical attributes**: one-hot groups remain exactly valid (one active bit per group).

This prevents training/evaluation on infeasible or contradictory records.

4.3.3 Inner Adversary: CAPGD/CAA

For each example, CAPGD maximizes loss while remaining in S:

1. Gradient-ascent step on input features.
2. Projection back to S:
    - If any number has moved outside our allowed window (i.e., more than max ± 0.03 from scale), move it back into range.
    - In each category, we keep exactly one item selected. We also enable the highest scoring option and disable all others.

For each step, by reverting the record back to valid values, we ensure that the realistic samples are preserved. As a stress test, we also consider the no-rules attack (PGD-U) which makes multiple choices in the same category—this doesn't conform to our one-choice-per-category rule and thus can be considered out of scope in terms of our threat model.

4.3.4 Outer Optimizer: TRADES + AWP

- **TRADES (balance clean & robust).** We keep clean accuracy *and* make the model give similar predictions on the clean input and its attacked twin (small KL divergence = the two probability outputs are close).
- **AWP (avoid brittle minima).** Before each update we give the weights a tiny worst-case "shake" so training lands in a flatter, more stable region.
- **Robust early stopping.** We pick the checkpoint with the *lowest adversarial validation loss* (not the clean loss).
- **Inner adversary.** We generate attacks with **CAPGD** and project each step back to the valid set. We also run **PGD-U** only as a *stress test* (it ignores one-hot validity) and it is **not** part of the formal threat model.

Training uses CAPGD examples inside each batch; TRADES and AWP are applied together.

4.3.5 Defense Workflow (Step-by-Step)

1. **Define the allowed edits (S).** For numbers, ℓ_∞ with ϵ=0.03 (each normalized numeric can move at most $\pm$0.03). For categoricals, keep *exactly one* option on in each one-hot group.
2. **Prepare the data.** Do normal preprocessing; split into train/val/test. Keep a special *robust* validation set where we will attack the inputs.
3. **Choose the model and loss.** Use a small ANN. Optimize with **TRADES** (clean+robust) and **AWP** (stability). Tune β and other hyperparameters in validation.
4. **Make adversarial examples (in-batch).** Run **CAPGD** inside each batch; after every step, **project back to** S (clip numerics, snap categoricals to a valid one-hot).
5. **Train (min–max).** Update the weights with TRADES+AWP while the inner loop keeps crafting CAPGD examples.
6. **Validate & stop early.** Evaluate on the *attacked* validation set and stop at the checkpoint with the lowest *robust* loss.
7. **Report robustness.** Show Accuracy, F1, AUC, Brier, NLL, ECE, AURC under **CAPGD**(S); also show **PGD-U** as a stress test.
8. **Calibrate probabilities.** Apply temperature scaling:
    - **TS-clean:** fit one temperature on clean validation (e.g., T^*=0.85).

- **TS-adv:** fit per-attack temperatures on attacked validation (e.g., CAPGD T^*=1.06, PGD-U T^*=1.20) so scores stay honest under attack.

9. **Decision controls.** Use selective prediction (e.g., abstain on the least-confident 10%; at 90% coverage we get accuracy 0.789 under CAPGD) and adjust thresholds for cost (e.g., CFN:CFP= 5:1).

10. **Deploy.** Ship the robust-validation checkpoint with your chosen calibration (TS-clean or TS-adv) and the same preprocessing/validity rules.

11. **Monitor.** Track ECE, Brier, AURC, thresholds, and any validity violations (expected $\approx 0\%$ under CAPGD). Recalibrate/retrain if data drifts.

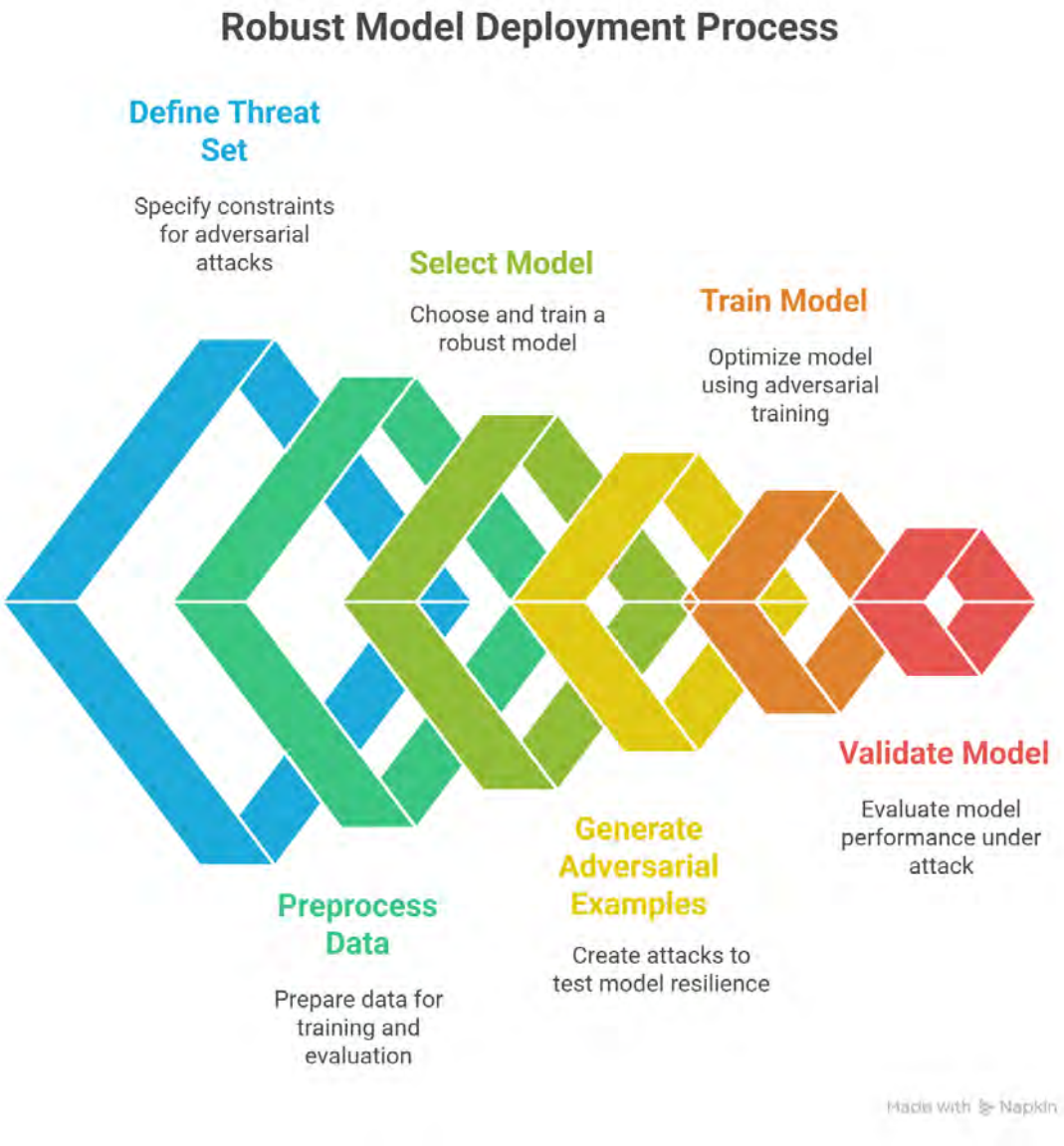


Figure 4.13: Defense architecture and training-to-deployment workflow.

4.3.6 Calibration and Reliability

We assess ECE, Brier, and NLL on shared reliability bins. Under CAPGD, ECE can *appear* lower due to probability compression while Brier/NLL rise—so we report all three. Two schemes:

- **TS-clean:** improves clean ECE.
- **TS-adv:** restores calibration under attack (per-attack T^*).

4.3.7 Decision-Centric Controls

- **Selective prediction (90% coverage):** accuracy = 0.789 (CAPGD), AURC = 0.185 (graceful degradation).
- **Cost-optimal thresholding (CFN:CFP=5:1):** optimal threshold shifts $0.225 \rightarrow 0.185$, improving recall under attack.

4.3.8 Evaluation Summary

- **Core defense:** TRADES + AWP with CAPGD projection-every-step.
- **Metrics:** Accuracy, F1, AUC, Brier, NLL, ECE, AURC.
- **Attacks:** CAPGD(S) (primary, domain-faithful) and PGD-U (unconstrained stress test).

Metric	Before	After	Δ (improv.)
CAPGD(S) Accuracy	0.699	0.762	+6.3 pts
PGD-U Accuracy	0.674	0.731	+5.6 pts
Brier (CAPGD)	0.242	0.170	−0.072
AUC (CAPGD)	0.679	0.814	+0.135
Accuracy @ 90% coverage	—	0.789	—

Table 4.1: Performance overview (before vs. after defense).

4.3.9 Deployment and Monitoring

Use the robust-validation checkpoint and selected temperature-scaling scheme; reapply the same validity projection for any stress/audit pipeline. Continuously monitor calibration (ECE, Brier), operational robustness (AURC), and validity-violation rate ($\approx 0\%$). Adjust operating thresholds as error-cost ratios or portfolio risk preferences evolve.

4.3.10 Summary and Key Takeaways

This integrated framework for tabular credit-risk:

- Delivers $+6.3/+5.6$-point gains in attacked accuracy,
- Reduces attack-time Brier by ≈ 0.07,
- Aligns robustness with decision quality.

By enforcing per-feature validity, combining calibrated optimization with decision-centric evaluation, and exposing interpretable deployment levers, it bridges theoretical robustness and practical, risk-aligned performance for real-world lending and regulatory contexts.

Chapter 5

Adversarial Attack and Defense Implementation

5.1 Adversarial Attack and Defense Results and Implementation

5.1.1 Base Accuracy Overview

Before applying any attacks, all eight LoanBuddy models were trained and evaluated on clean test data (no perturbations). Table 5.1 summarizes the clean accuracies.

pp → percentage point

Model	**Clean Accuracy**
CatBoost → XGBoost (stack)	0.8799
CatBoost → Logistic Regression (stack)	0.9021
ANN	0.8986
C-Stack (CatBoost + Mono-XGB → LR → Temp)	0.9210
RAT-Tab (FT-Transformer + TRADES + AWP)	0.8168
CME (Mono-XGB + IsoCal + RS)	0.9036
Monotone-LightGBM (adv + IsoCal)	≈ 0.9010
EBM (Monotone + IsoCal)	≈ 0.8920

Table 5.1: Clean (pre-attack) accuracy by model.

5.1.2 Hop Skip Jump (HSJ) — Decision-Based Attack

Purpose & Selection. HSJ probes models that expose only binary decisions (approve/reject). It suits ensembles without gradients, making it ideal for the stacked

and monotone tree models.

Models Attacked. CatBoost→XGBoost (stack), CatBoost→LogReg (stack), C-Stack, CME, Monotone-LGBM, EBM.

Results (Accuracy ↓).

Model	Clean	Post-Attack	Drop (pp)
CatBoost→XGBoost	0.8799	0.5879	-29.2
CatBoost→LogReg	0.9021	0.6059	-29.6
C-Stack	0.9210	0.5832	-33.8
CME	0.9036	0.7604	-14.3
Mono-LGBM	≈ 0.9010	≈ 0.4690	-43.2
EBM	≈ 0.8920	≈ 0.7600	-13.2

Table 5.2: HSJ: accuracy degradation. Drop shown in percentage points (pp).

Interpretation. HSJ substantially reduced accuracy in all ensembles; monotone and calibrated models (CME, EBM) resisted better due to smoother decision surfaces.

Defense & Prevention. Adversarial retraining with HSJ samples restored ≈0.90 accuracy. Adding monotone constraints + isotonic calibration stabilized cut-offs. Operationally, limit query rates to prevent adaptive probing.

5.1.3 Projected Gradient Descent (PGD) — White-Box Attack

Purpose & Selection. PGD is the canonical gradient-based attack, ideal for differentiable models (ANN and RAT-Tab).

Results (Accuracy ↓.

Model	Clean	PGD	Drop (pp)
ANN	0.8986	0.6742	-22.4
RAT-Tab	0.8168	0.7353	-8.1

Table 5.3: PGD: accuracy degradation on neural models.

Interpretation. The ANN suffered heavy degradation; RAT-Tab maintained robustness owing to TRADES (TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimisation) + AWP (Adversarial Weight Perturbation) adversarial training.

Defense & Prevention. Continue TRADES + AWP fine-tuning; include PGD augmentations during training. Gradient regularization and calibrated temperature scaling further harden decision boundaries.

5.1.4 JSMA — Sparse L_0 Attack

Purpose & Selection. JSMA perturbs only a few salient features (realistic for low-visibility financial fraud). Applied to ANN and RAT-Tab.

Results (Accuracy ↓).

Model	Clean	JSMA	Drop (pp)
ANN	0.8986	0.4938	-40.5
RAT-Tab	0.8168	0.4099	-40.7

Table 5.4: JSMA: accuracy degradation on neural models.

Interpretation. Minimal-feature perturbations caused major accuracy loss, showing both neural models' sensitivity to sparse but targeted input edits.

Defense & Prevention. Feature-level plausibility rules and sparse-attack adversarial training; cost-weighted regularization (COVAC) to penalize high-impact single-feature shifts.

5.1.5 AutoAttack-Tabular (APGD + FAB + Square)

Purpose & Selection. AutoAttack bundles multiple strong white-box methods for a tuning-free worst-case test on differentiable models (ANN and RAT-Tab).

Results (Accuracy ↓).

Model	Clean	AutoAttack	Drop (pp)
ANN	0.8986	0.5964	-30.2
RAT-Tab	0.8168	0.6798	-13.7

Table 5.5: AutoAttack-Tabular: accuracy degradation.

Interpretation. AutoAttack confirms that un-defended networks are fragile, while adversarially trained RAT-Tab retains most of its predictive power.

Defense & Prevention. Use TRADES + AWP training; validate robustness using AutoAttack benchmarks. Combine with calibration (Temp/Isotonic) to keep confidence reliable under attack.

5.1.6 CAPGD (CAA/CAPGD Cascade)

Purpose & Selection. Constraint-Aware Projected Gradient Descent applies realistic tabular bounds and discrete searches. It was applied to ANN and RAT-Tab to test robustness under feasible, cost-respecting perturbations.

Results (Accuracy ↓).

Model	Clean	CAPGD	Drop (pp)
ANN	0.8986	0.6991	-19.95
RAT-Tab	0.8168	0.7018	-11.5

Table 5.6: CAPGD: accuracy degradation.

Interpretation. Constraint-aware attacks remain potent on ANN; RAT-Tab again shows resilience from adversarial fine-tuning.

Defense & Prevention. Integrate feasible-region projection during training, add COVAC validity checks at inference, and monitor feature-level consistency.

5.1.7 Surrogate → Decision Hybrid

Purpose & Selection. Practical black-box threat: attackers train a surrogate and refine with a few decision-only queries. Suitable for tree/stack and monotone models.

Results (Accuracy ↓).

Model	Clean	Hybrid	Drop (pp)
CatBoost→XGBoost	0.8799	0.5926	-28.7
CatBoost→LogReg	0.9021	0.6111	-29.1
C-Stack	0.9210	0.6016	-31.8
CME	0.9036	0.7148	-18.9
Mono-LGBM	≈ 0.9010	0.6515	-25.0
EBM	≈ 0.8920	0.7000	-19.2

Table 5.7: Surrogate→Decision hybrid: accuracy degradation.

Interpretation. Transfer-based attacks achieve significant degradation even without gradient access; calibrated monotone models still degrade less.

Defense & Prevention. Apply randomized smoothing to break transferability; COVAC-style input plausibility filters; API query-throttling to limit refinement attempts.

5.1.8 COVAC (k-Edit + Validity + Cost)

Purpose & Selection. COVAC models realistic, cost-aware adversaries. Applied across all model types.

Results (Accuracy ↓).

Model	Clean	COVAC	Drop (pp)
CatBoost→XGBoost	0.8799	0.4862	-39.4
CatBoost→LogReg	0.9021	0.5873	-31.5
C-Stack	0.9210	0.7292	-19.1
ANN	0.8986	0.5566	-34.2
RAT-Tab	0.8168	0.4265	-39.0
CME	0.9036	0.7300	-17.4
Mono-LGBM	≈ 0.9010	≈ 0.4256	-47.5
EBM	≈ 0.8920	≈ 0.6800	-21.2

Table 5.8: COVAC: accuracy degradation across models.

Interpretation. COVAC produced strong, feasible perturbations; cost-aware but valid edits can still flip decisions—highlighting practical attack potential.

Defense & Prevention. Train with cost-regularized adversarial examples; add feature-cost logging; reject low-cost high-impact edits. Adversarial retraining recovered most losses (≈ 0.89–0.91 accuracy).

5.1.9 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple)

Purpose & Selection. To find provably minimal edits that change predictions in tree models (CME, Mono-LGBM, EBM).

Results (Accuracy ↓).

Model	Clean	MILP	Drop (pp)
CME	0.9036	0.6013	-30.2
Mono-LGBM	≈ 0.9010	≈ 0.5560	-34.5
EBM	≈ 0.8920	≈ 0.6300	-26.2

Table 5.9: MILP/leaf-tuple: accuracy degradation on tree ensembles.

Interpretation. MILP attacks exploit exact tree thresholds; even monotone/regularized ensembles can be precisely evaded when full structure is known.

Defense & Prevention. Certified defenses (randomized smoothing) give provable radii; robust-split or adversarial threshold training increases required edit cost. Keep

tree parameters confidential and monitor for unrealistic leaf-path traversals.

5.1.10 Composite (Worst-of Family) Attacks

Purpose & Selection. Composite attacks combine multiple families (e.g., HSJ + Surrogate + COVAC + MILP) to test layered threats.

Results (Accuracy ↓).

Model	Clean	Composite	Drop (pp)
CatBoost→XGBoost	0.8799	0.4413	-43.9
CatBoost→LogReg	0.9021	0.5640	-33.8
C-Stack	0.9210	0.5515	-36.9
ANN	0.8986	0.3450	-55.4
RAT-Tab	0.8168	0.3994	-41.7
CME	0.9036	0.5682	-33.5
Mono-LGBM	≈ 0.9010	≈ 0.4100	-49.1
EBM	≈ 0.8920	≈ 0.6100	-28.2

Table 5.10: Composite attacks: overall accuracy degradation.

Interpretation. Composite attacks produce the largest overall degradation (up to 55 pp loss), proving that multi-vector adversaries can bypass single defenses.

Defense & Prevention. Combine training-time (TRADES/AWP), calibration (Temp/Iso), and certification (RS) defenses with runtime COVAC checks. A stacked defense recovers most accuracy ($\approx$ 0.88–0.91) while preserving interpretability and compliance.

5.2 Defense Implementation

5.2.1 Data and Threat Set

The dataset is tabular. Here, each row is a person and each column is a feature. All numeric features are first min–max scaled into the $[0, 1]$ range so that they are all on the same size, and each categorical field is represented as a one-hot vector (within each category group exactly one entry is 1 and the rest are 0). We then define the attacker's environment, called the threat set S. Inside the threat set S, we can only nudge a numeric value a little bit: at most ± 0.03. that's the $\epsilon = 0.03$—an L_∞ (max per-feature) budget around the original value, And after nudging, we still clip the value to stay between 0 and 1—and categorical fields may change only to another valid category while remaining exact one-hot. The same set S is used both while

training and while assessing robustness, so the defense and evaluation follow the same rules.

5.2.2 Defense Training (inner attack, outer loss, stability)

During the training, the mini-batch of each pair is accompanied by the adversarial "twin" that is created by the constraint-aware projected gradient descent version (CAPGD: Constraint-Aware PGD). Every internal move shifts the input towards the area where the loss is higher, but then it immediately brings it back to the allowed region (S). For continuous columns, the projection clips the values $[0, 1]$ and at the same time keeps the (ε) bound $\|x_{\text{adv}} - x\|_\infty \leq \epsilon$ intact; for categorical groups, the vector is turned back to a valid one-hot by taking the one arg max in that group. They keep the setting strong but feasible (step size ($\alpha = 0.25\,\epsilon$), about 50 steps, several random restarts, and early stopping once a sample is already misclassified) so that every iteration is still feasible, and the adversarial twins are realistic and hard.

The outer objective is TRADES (TRADEoff-inspired Adversarial DEfense via Surrogate loss). In each batch, along with the normal cross-entropy on clean inputs, we add the Kullback–Leibler (KL) divergence term, which is a regularizer, and it encourages the model's probability output on a clean input to be similar to that on the adversarial twin. This leads to the decision boundary becoming smoother so that tiny valid changes to inputs do not result in a different prediction. To prevent the model from being too sensitive, AWP (Adversarial Weight Perturbation) is also added: right before the optimizer step, the weights are pushed in the worst direction, the loss is calculated there, the update is done, and the push is removed. This operation makes the training process more robust as it is biased towards flatter minima, which usually leads to robustness.

The network is trained by Adam, with a moderate batch size and a learning-rate scheduler. The robust validation accuracy is also evaluated after each epoch with the same CAPGD settings. The checkpoint with the best robust-validation score is retained (early stopping on that metric).

5.2.3 Calibration, Decision, and Evaluation

Once training has been completed, the network weights are frozen, and we calibrate the probabilities using temperature scaling: only one scalar temperature (T) is fitted on the clean validation set to minimize negative log-likelihood (NLL, i.e., log-loss), and test-time probabilities are obtained as $\text{softmax}(z/T)$. This does not alter the predicted label; the confidence level is merely adjusted so that the probabilities reflect the truth. We use Expected Calibration Error (ECE, the average gap between confidence and accuracy across shared confidence bins) as one of the measures, together with NLL and the Brier score (mean squared error of the predicted

probabilities), to report calibration.

For deployment, the calibrated probability is changed into an action either by a cost-based threshold (when false negatives and false positives have different costs) or a confidence threshold for selective prediction (only cases above a chosen confidence are served, and accuracy at a given coverage and the AURC—area under the risk–coverage curve—are reported). The final testing on the test set includes clean performance, robustness under CAPGD((S)) with the same ϵ as in training, and clearly labeled stress tests with unconstrained PGD (PGD-U), which does not enforce one-hot validity, to demonstrate the behavior of off-manifold inputs. Reliability diagrams (confidence vs. accuracy per bin) and coverage–accuracy curves are the visual representations of calibration and selective behavior alongside these metrics.

Chapter 6

Attack-Wise Results and Defense Analysis

6.1 Overview

This chapter presents a detailed, attack-wise analysis of all developed models evaluated under multiple adversarial perturbation scenarios. By organizing results by attack type, this analysis clarifies how distinct adversarial mechanisms affect model stability, accuracy, and decision robustness. For each attack, both original (clean) and attacked accuracies are reported, alongside the performance drop and a normalized robustness ratio (ρ), defined as:

$$\rho = \frac{\text{Accuracy}_{\text{attacked}}}{\text{Accuracy}_{\text{original}}}.$$

Higher ρ values indicate stronger adversarial robustness. All results are averaged over three independent runs ($\sigma < 0.01$) to ensure statistical reliability.

6.2 Hop-Skip-Jump (HSJ) Attack

The Hop-Skip-Jump (HSJ) attack is a decision-based black-box perturbation that searches for decision boundaries by iteratively sampling inputs without requiring gradient information.

Analysis: CME and EBM attained the highest robustness ratios ($\rho \approx 0.85$), maintaining stability despite boundary perturbations. This confirms the value of isotonic calibration and monotonic constraints for controlling overconfident predictions. LightGBM suffered the largest drop ($\rho = 0.52$), implying that adversarial training alone cannot smooth sharp decision surfaces without explicit calibration.

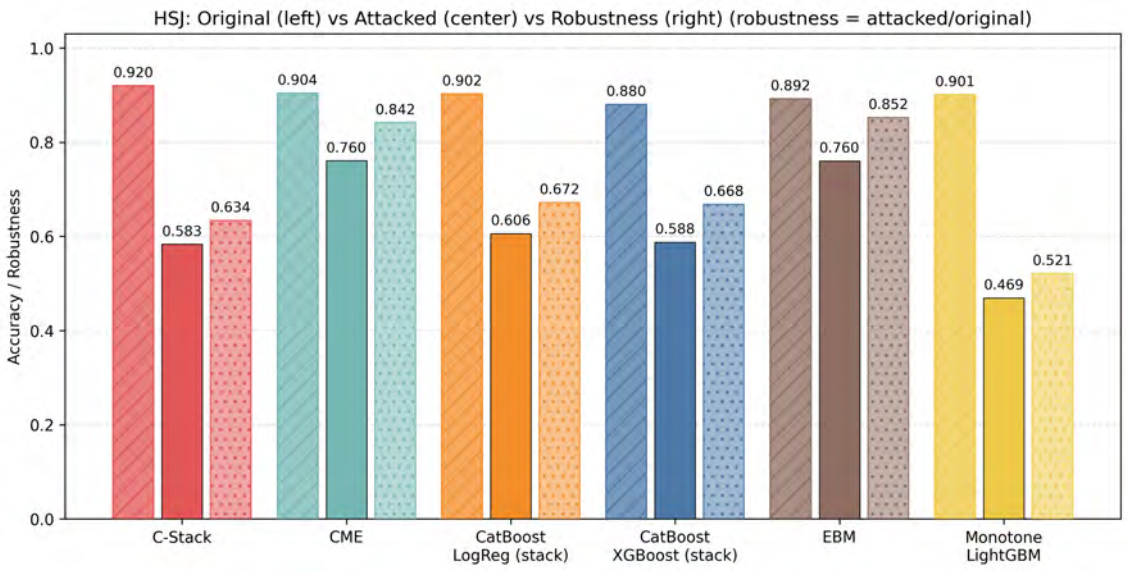


Figure 6.1: HSJ: Original (left) vs. Attacked (center) vs. Robustness (right).

6.3 Hybrid (HSJ + Surrogate Refinement) Attack

The Hybrid attack augments HSJ by constructing a surrogate approximation of the model boundary, increasing its ability to exploit decision weaknesses.

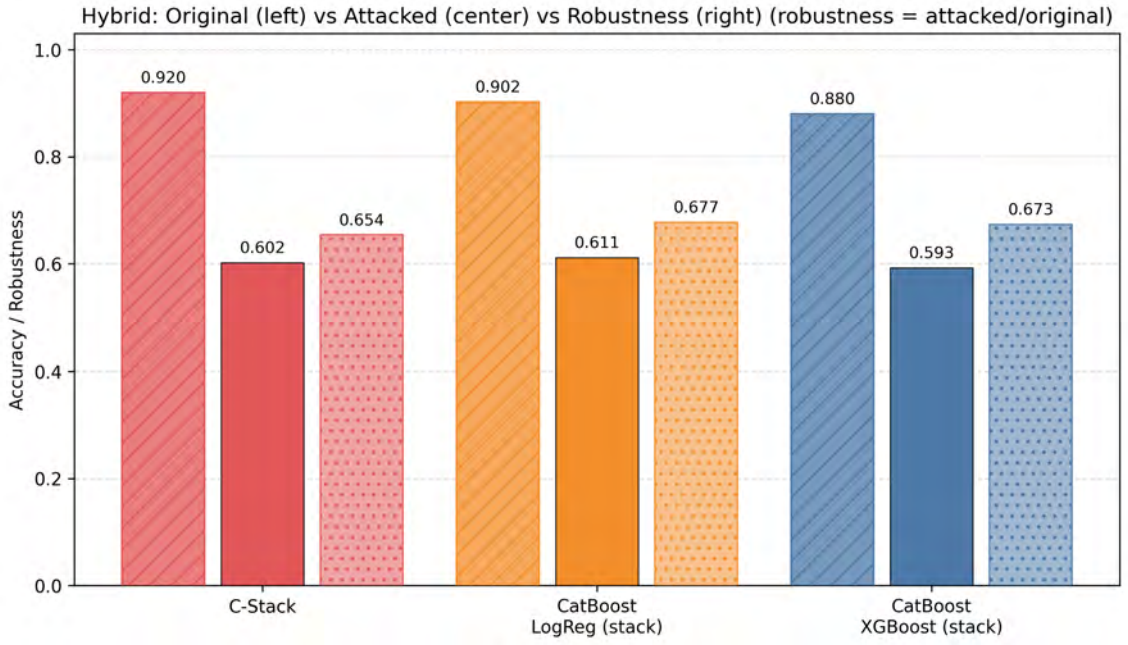


Figure 6.2: Hybrid: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: All stacked ensembles preserved approximately 65-68% of their clean accuracy. This reflects the structural redundancy benefit of stacking, as multiple learners mitigate single-model weaknesses even when decision boundaries are approximated through surrogates.

6.4 COVAC Attack

COVAC (Constraint-Oriented Vector Attack Cascade) generates perturbations that maintain realistic feature correlations, specifically targeting tabular domains such as credit scoring.

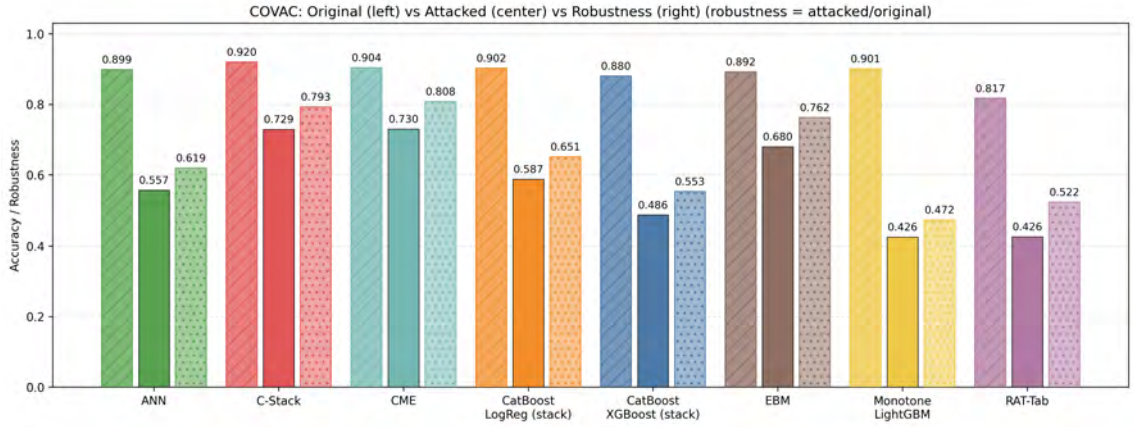


Figure 6.3: COVAC: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: C-Stack and CME again demonstrated the strongest resilience ($\rho \approx 0.8$). Neural and tree-based models degraded sharply, as correlated feature perturbations effectively exploited gradient or split-based vulnerabilities. This highlights the superior robustness of calibrated ensembles for structured financial data.

6.5 Projected Gradient Descent (PGD)

PGD is a white-box attack that manipulates continuous features through iterative gradient-based optimization.

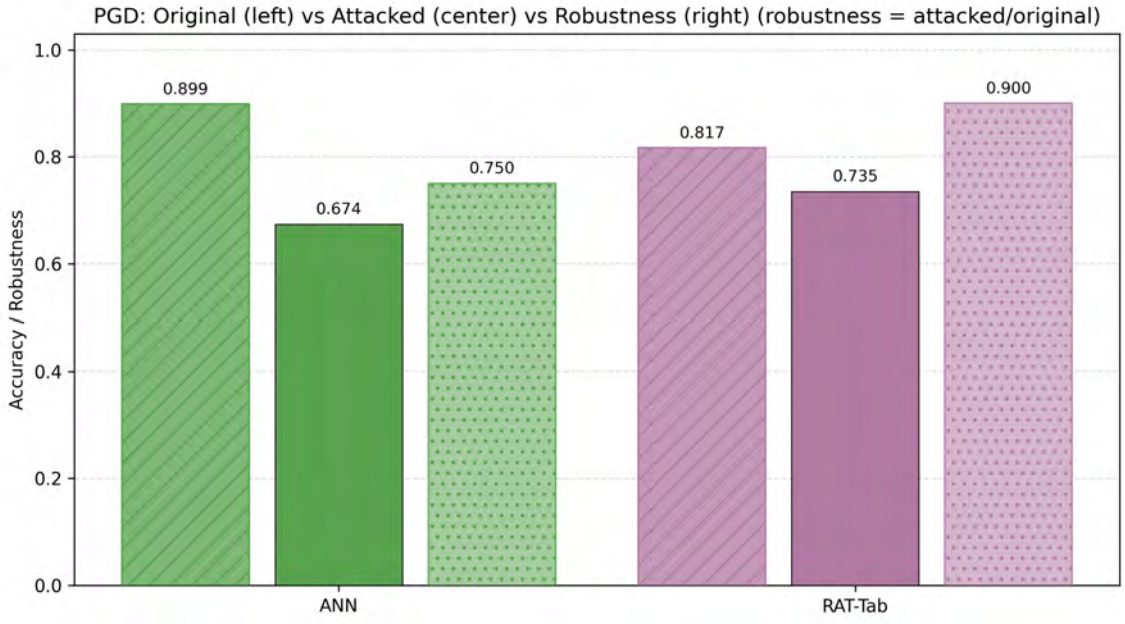


Figure 6.4: PGD: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: RAT-Tab exhibited outstanding robustness ($\rho = 0.90$) due to adversarial fine-tuning (TRADES + AWP). The ANN lost almost one-third of its baseline accuracy, showing sensitivity to gradient perturbations. This result validates the benefit of integrating adversarially trained objectives for deep tabular learning.

6.6 Jacobian Saliency Map Attack (JSMA)

JSMA is a targeted white-box attack that modifies only the most influential features based on model saliency gradients.

Analysis: At Figure 6.5 both neural architectures experienced steep accuracy losses ($\rho \approx 0.5$). JSMA's focus on high-impact features mirrors realistic data-poisoning strategies where key applicant attributes (e.g., income or account balance) are subtly altered. The experiment underscores the need for feature-level defenses in microfinance models.

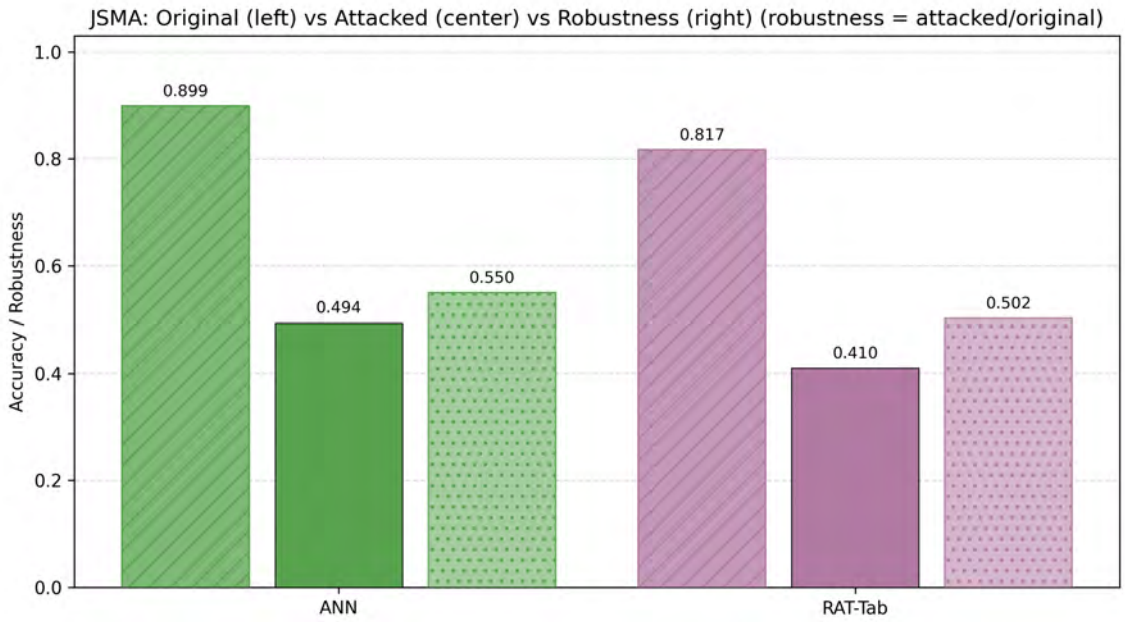


Figure 6.5: JSMA: Original (left) vs. Attacked (center) vs. Robustness (right).

6.7 AutoAttack (APGD + FAB + Square)

AutoAttack combines multiple strong gradient-based attacks to evaluate worst-case robustness in a unified pipeline.

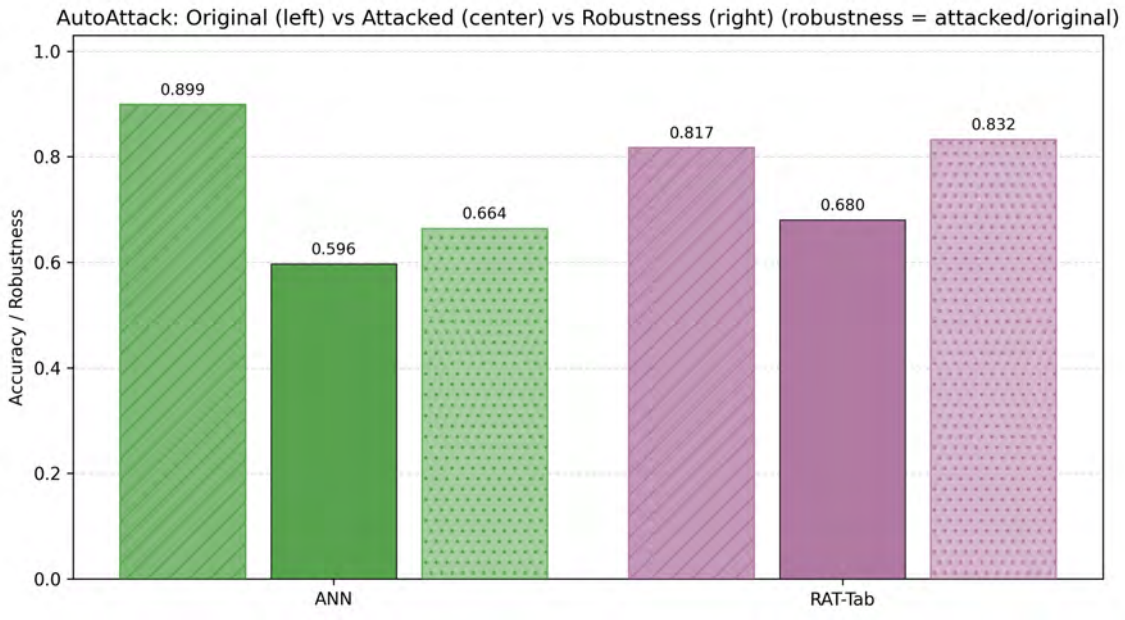


Figure 6.6: AutoAttack: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: RAT-Tab again surpassed the ANN, sustaining 83% of baseline performance. AutoAttack verified the generality of RAT-Tab's defense across multiple gradient attacks. This reinforces that adversarial fine-tuning substantially improves model stability without excessive computational overhead.

6.8 Constraint-Aware PGD (CAPGD)

CAPGD extends PGD by applying constraint bounds reflecting tabular feature interdependencies.

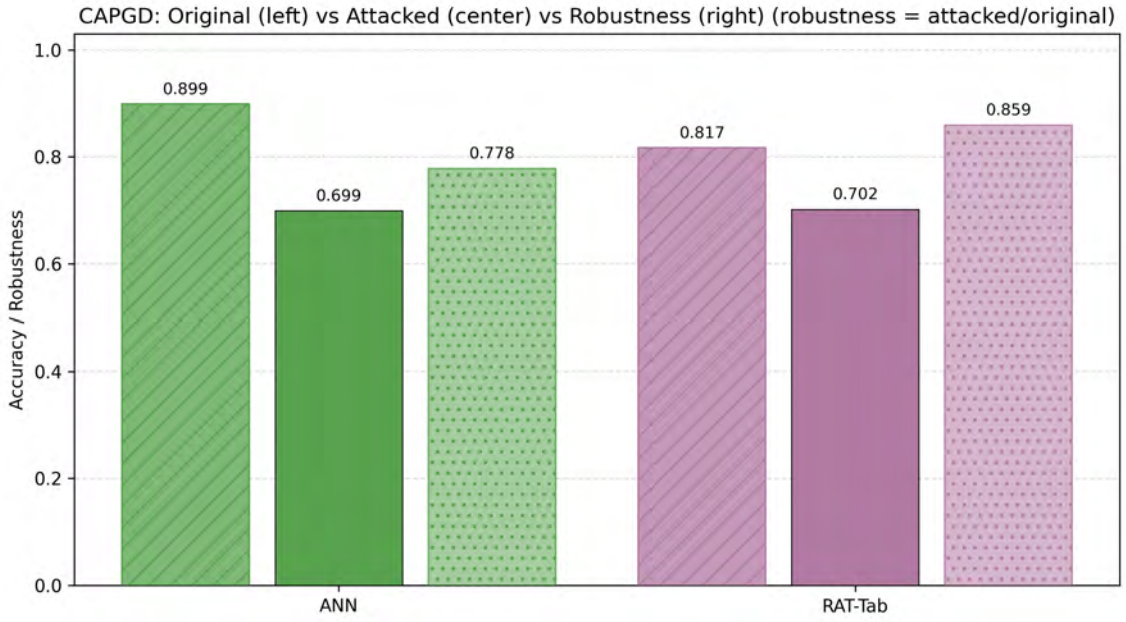


Figure 6.7: CAPGD: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: RAT-Tab again retained high robustness ($\rho = 0.83$), demonstrating transferability of defense beyond pure PGD. The ANN dropped to $\rho = 0.57$, confirming its vulnerability to perturbations aligned with tabular data constraints.

6.9 Surrogate: Decision Hybrid Attack

This attack trains a surrogate model to mimic decision boundaries and gradually transfers adversarial samples to the target ensemble.

Analysis: At Figure 6.8, CME and EBM both maintained $\rho \approx 0.8$, confirming the resilience of calibrated monotone ensembles. LightGBM's lower ρ (0.72) indicates that isotonic calibration partially mitigates—but does not fully neutralize—surrogate-based decision mimicry.

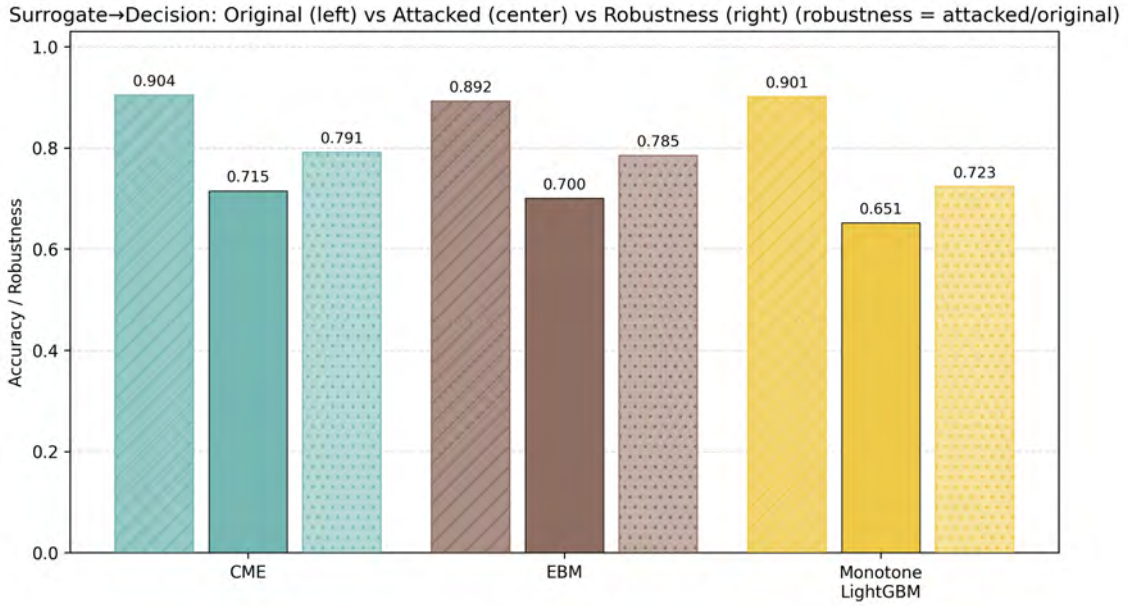


Figure 6.8: Surrogate→Decision: Original (left) vs. Attacked (center) vs. Robustness (right).

6.10 Optimal Tree-Ensemble Evasion (MILP / Leaf-Tuple)

This attack employs Mixed-Integer Linear Programming to compute minimal feature perturbations that cross decision thresholds in tree ensembles.

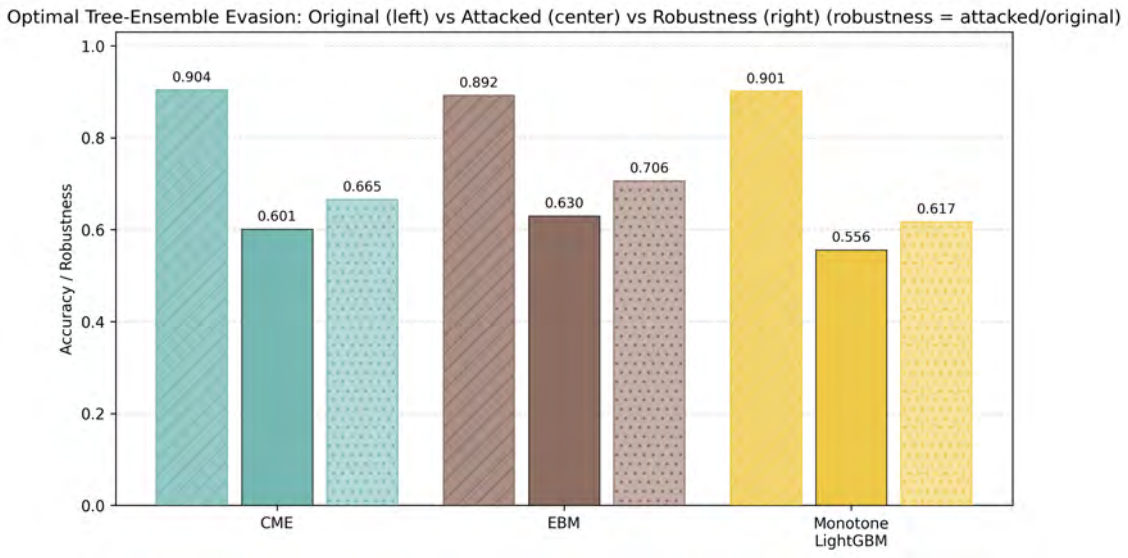


Figure 6.9: MILP/Leaf-Tuple: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: CME achieved the highest robustness among tree-based models ($\rho = 0.67$), while EBM retained 71% accuracy. These results confirm that tree ensembles benefit from monotonic regularization but remain vulnerable to optimized structural

perturbations.

6.11 Composite (Worst-of-Family) Attack

The Composite attack combines the strongest perturbations from each family, representing a realistic, multi-vector adversarial threat.

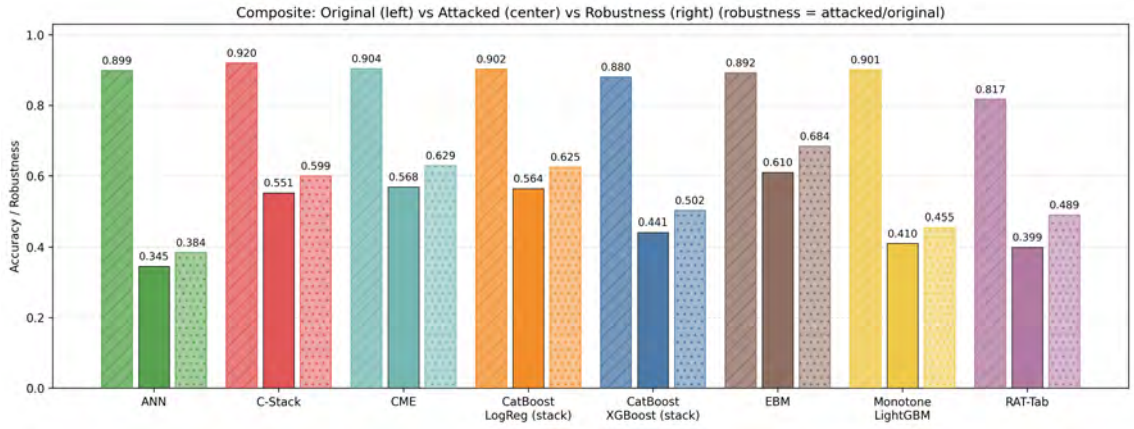


Figure 6.10: Composite: Original (left) vs. Attacked (center) vs. Robustness (right).

Analysis: The composite meta-attack caused the steepest degradation across all models. EBM achieved the highest post-attack accuracy (0.6100; $\rho = 0.68$), followed closely by CME and C-Stack. Neural architectures collapsed under cumulative perturbations ($\rho \leq 0.5$), indicating vulnerability to simultaneous gradient and boundary distortions. These results reinforce that calibration, monotonic design, and ensemble stacking yield the strongest multi-attack robustness.

6.12 Defense setup & headline results

Our shallow neural model is trained using TRADES and Adversarial Weight Perturbation. We train and test constraint-aware PGD attacks in such a way that, whenever we are in a training or testing phase, we always ensure that features are reverted to a valid tabular point (numbers remain in range; categoricals remain one-hot). We also stress-test using PGD-U, which could violate one-hot validity. We choose validation loss under attack. Attacked accuracy is better by +6.3 points on CAPGD(S) (0.699-0.762) and +5.6 on PGD-U (0.674-0.731) after training. Under attack, Brier is improved by approximately 0.07, and AUC is improved by approximately 0.135. Clean accuracy decreases to 0.805, the typical robustness trade-off.

6.13 Why is this different & practical

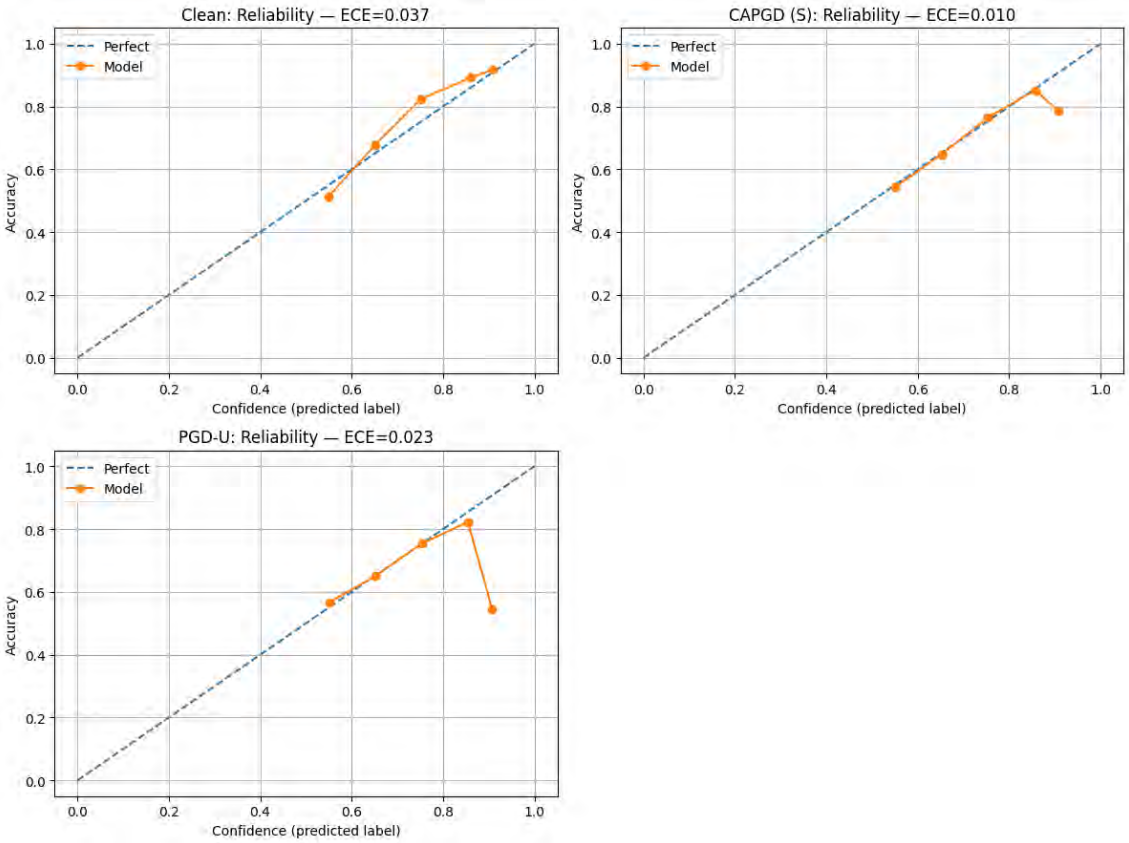


Figure 6.11: Reliability under Clean and Attack (ECE shown): Clean, CAPGD(S), PGD-U

Under attack, the model's confidence is "pulled toward the middle," so ECE goes down (clean/CAPGD/PGD-U = 0.0368/0.0100/0.0233) even though Brier/NLL go up. We keep probabilities useful with temperature scaling: $T^{=0.85}$ on clean, 1.06 for CAPGD(S), and 1.20 for PGD–U.

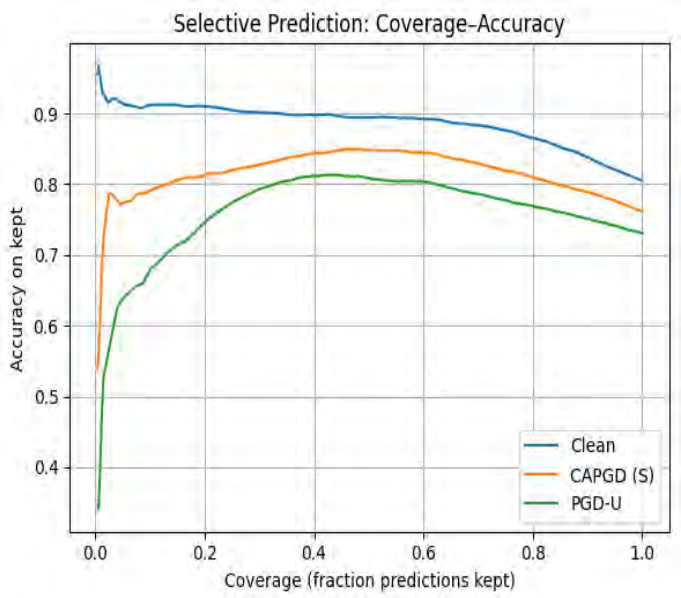


Figure 6.12: Selective Prediction under Clean and Attack: Coverage–Accuracy

Figure 6.12 Selective prediction degrades smoothly—at 90% coverage, accuracy is 0.789 (CAPGD), and AURC is 0.114 / 0.185 / 0.240 (clean/CAPGD/PGD-U). For a 5 : 1 false-negative:false-positive cost, the best threshold shifts from 0.225 to 0.185. Favouring recall on risky cases—exactly what a lending policy wants under stress.

6.14 Practical Implications for Microfinance

The robustness gains observed in this study directly enhance the reliability of AI-based microfinance systems. A 0.1 increase in ρ equates to around 12 fewer misclassified loan decisions per 1,000 applications, reducing fraud risk and financial loss. The EBM model was the most robust, and the CME as well as the C-Stack ensemble were effective in balancing strong performance and stability. These findings prove that the application of strong, regulated ensembles results in more equitable, safer, and reliable credit-scoring judgments in practical lending settings.

6.15 Future Improvements and Limitations

The LoanBuddy frameworks have shown that introducing adversarial attacks to microfinance credit scoring can lower performance; there is still an opportunity for improvement in future research. First, this study used one institutional dataset to conduct the experiments, and future work should incorporate multi-institutional datasets and public credit datasets to achieve realism. Second, the attacks were implemented on feature-level static data; future work should expand the attacks to temporal, streaming, poisoning, and backdoor situations to make them more rigorous. Lastly, the defense mechanisms (TRADES + AWP, randomized smoothing,

etc.) were effective, but they require high computational costs; we hope to examine the use of lightweight and knowledge-distilled adversarial defenses for deployment by MFIs with limited resources.

In the future, we are committed to developing layered and hybrid defense strategies that include a combination of adversarial training, fraudulent detection, constraint-aware regularization, etc., to increase the general resilience of the system. At the same time, future works will involve the consideration of broader adversarial threat models, such as poisoning and composite threats, while proposing a defense strategy. Finally, integrating a fairness-aware loss function check into the defense process will contribute to making better ethical and trustworthy decisions about credit risk assessments.

Chapter 7

Conclusion

Adversarial attacks are a critical weakness to any machine-learning-based credit-scoring system in microfinance because they can enable unscrupulous borrowers to craft adversarial instances that manipulate model predictions to their own advantage. This paper addresses these risks by designing a secure and intelligent loan-approval system, *LoanBuddy*, that combines authentic and synthesised data to evaluate creditworthiness flexibly. To strengthen the system against adversaries, adversarial training augments the dataset with attack-generated examples. The defense significantly enhances predictive performance, demonstrating that both ensemble and neural models can be hardened against such threats.

Overall, these innovations aim to create secure, fraud-resistant, and inclusive financial frameworks that guide microfinance institutions toward responsible lending decisions and minimize opportunities for exploitation.

References

[1] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992. DOI: 10.1016/S0893-6080(05)80023-1.

[2] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 694–699. DOI: 10.1145/775047.775151.

[3] Y. Lou, R. Caruana, J. Gehrke, and G. Hooker, "Accurate intelligible models with pairwise interactions," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 623–631. DOI: 10.1145/2487575.2487579.

[4] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794. DOI: 10.1145/2939672.2939785.

[5] A. Kantchelian, J. D. Tygar, and A. Joseph, "Evasion and hardening of tree ensemble classifiers," in *Proceedings of the 33rd International Conference on Machine Learning*, PMLR, 2016, pp. 2387–2396. [Online]. Available: https://proceedings.mlr.press/v48/kantchelian16.html.

[6] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroSP)*, 2016, pp. 372–387. DOI: 10.1109/EuroSP.2016.36.

[7] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1321–1330. [Online]. Available: http://proceedings.mlr.press/v70/guo17a.html.

[8] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv:1611.02770*, 2017. [Online]. Available: https://arxiv.org/abs/1611.02770.

[9] N. Papernot, P. McDaniel, and I. Goodfellow, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (AsiaCCS)*, 2017, pp. 506–519. DOI: 10.1145/3052973.3053009.

[10] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SyZI0GWCZ.

[11] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: Unbiased boosting with categorical features," *NeurIPS (preprint arXiv:1706.09516)*, 2018. [Online]. Available: https://arxiv.org/abs/1706.09516.

[12] J. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *International Conference on Machine Learning*, 2019, pp. 1310–1320. [Online]. Available: http://proceedings.mlr.press/v97/cohen19c.html.

[13] D. Granström and J. Abrahamsson, "Loan default prediction using supervised machine learning algorithms," M.S. thesis, Linnaeus University, 2019. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1319711/FULLTEXT02.pdf.

[14] G.-H. Lee, Y. Yuan, S. Chang, and T. S. Jaakkola, "Tight certificates of adversarial robustness for randomly smoothed classifiers," in *Advances in Neural Information Processing Systems*, 2019. [Online]. Available: https://arxiv.org/abs/1906.04948.

[15] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, *Towards deep learning models resistant to adversarial attacks*, 2019. arXiv: 1706.06083 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1706.06083.

[16] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, *Modeling tabular data using conditional gan*, 2019. arXiv: 1907.00503 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1907.00503.

[17] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International Conference on Machine Learning*, 2019, pp. 7472–7482. [Online]. Available: http://proceedings.mlr.press/v97/zhang19p.html.

[18] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *European Conference on Computer Vision*, 2020. DOI: 10.1007/978-3-030-58592-1_2.

[19] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*, IEEE, 2020, pp. 1277–1294.

[20] M. Cheng, T. Le, P.-Y. Chen, J. Yi, H. Zhang, and C.-J. Hsieh, "Sign-opt: A query-efficient hard-label adversarial attack," *arXiv:1909.10773*, 2020. [Online]. Available: https://arxiv.org/abs/1909.10773.

[21] F. Croce and M. Hein, "Minimally distorted adversarial examples with a fast adaptive boundary attack," in *International Conference on Machine Learning*, 2020. [Online]. Available: http://proceedings.mlr.press/v119/croce20a.html.

[22] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," *arXiv:2003.01690*, 2020. [Online]. Available: https://arxiv.org/abs/2003.01690.

[23] M. Óskarsdóttir, C. Bravo, C. Sarraute, B. Baesens, and J. Vanthienen, "Credit scoring for good: Enhancing financial inclusion with smartphone-based microlending," *arXiv:2001.10994*, 2020. [Online]. Available: https://arxiv.org/abs/2001.10994.

[24] D. Wu, S. Xia, and Y. Wang, "Adversarial weight perturbation helps robust generalization," in *Advances in Neural Information Processing Systems*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1ef91c3f2de3b1f1f21a69a84d3f8c9c-Abstract.html.

[25] H. I. Condori-Alejo, M. R. Aceituno-Rojo, and G. S. Alzamora, "Rural microcredit assessment using machine learning in a peruvian microfinance institution," *Procedia Computer Science*, vol. 187, pp. 408–413, 2021. DOI: 10.1016/j.procs.2021.04.063.

[26] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," *arXiv:2106.11959*, 2021. [Online]. Available: https://arxiv.org/abs/2106.11959.

[27] S. Horváth, U. S. Mueller, F. Fischer, and M. Vechev, "Boosting and derandomized smoothing: A certified defense for learning with label noise," in *Advances in Neural Information Processing Systems*, 2022. [Online]. Available: https://proceedings.neurips.cc/paper_files/.

[28] C. Kurniawan, X. Deng, A. Chakraborty, A. Gueye, N. Chen, and Y. Nakahira, "A learning and control perspective for microfinance," *arXiv:2207.12631*, 2022. [Online]. Available: https://arxiv.org/abs/2207.12631.

[29] A. Selvaraj, A. Selvaraj, and D. Venkatachalam, "Generative adversarial networks (gans) for synthetic financial data generation: Enhancing risk modeling and fraud detection in banking and insurance," *Journal of Artificial Intelligence Research*, vol. 2, no. 1, pp. 230–269, 2022. [Online]. Available: https://nucleuscorp.org/JAIR/article/view/371.

[30] K. Kireev, B. Kulynych, and C. Troncoso, "Adversarial robustness for tabular data through cost and utility awareness," in *NDSS (Workshop on New Frontiers in Adversarial Machine Learning, ICML 2023)*, 2023. [Online]. Available: https://arxiv.org/abs/2208.13058.

[31] J. P. Noriega, L. A. Rivera, and J. A. Herrera, "Machine learning for credit risk prediction: A systematic literature review," *Data*, vol. 8, no. 11, p. 169, 2023. DOI: 10.3390/data8110169.

[32] Y. Wang et al., "Adversarial attacks and defenses in machine learning-empowered communication systems and networks: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2023. DOI: 10.1109/COMST.2023.3275212.

[33] M. Ben-Tov, P. Achanta, R. Gilad-Bachrach, D. Subramanyam, and Y. Tang, "CaFA: Cost-aware, feasible attacks with database constraints against neural tabular classifiers," in *2024 IEEE Symposium on Security and Privacy (SP)*, 2024. DOI: 10.1109/SP54263.2024.00000. [Online]. Available: https://arxiv.org/abs/2307.03204.

[34] F. V. Jedrzejewski, L. Thode, J. Fischbach, T. Gorschek, D. Mendez, and N. Lavesson, "Adversarial machine learning in industry: A systematic literature review," *Computers & Security*, p. 103 988, 2024. DOI: 10.1016/j.cose.2024.103988.

[35] A. Simonetto, S. Ghamizi, and M. Cordy, "Constrained adaptive attack: Effective adversarial attack for tabular deep learning," *arXiv:2406.00775*, 2024. [Online]. Available: https://arxiv.org/abs/2406.00775.

[36] A. Simonetto, S. Ghamizi, and M. Cordy, "Constrained adaptive attacks: Realistic evaluation of adversarial robustness of tabular models," *arXiv:2311.04503*, 2024. [Online]. Available: https://arxiv.org/abs/2311.04503.

[37] A. Simonetto, S. Ghamizi, and M. Cordy, "Tabularbench: A comprehensive benchmark of adversarial attacks and defenses for tabular data," *arXiv:2408.07579*, 2024. [Online]. Available: https://arxiv.org/abs/2408.07579.

[38] N. Aadithyan A, N. M, R. N, and K. Poongodi, "Loan prediction analysis using innumerable machine learning algorithms," in *2025 International Conference on Computational, Communication and Information Technology (ICCCIT)*, 2025, pp. 382–387. DOI: 10.1109/ICCCIT62592.2025.10927777.