Universiteit Leiden
The Netherlands

Data Science & AI

Investigating Local Robustness of TabPFN on Small Numerical Binary Classification Tasks

Willem Scholten

Supervisors:
Dr. Jan N. van Rijn & Prof. Dr. Holger H. Hoos

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

17/07/2026

Abstract

Adversarial robustness has been extensively studied for image classifiers, but remains critically underexplored for tabular models, despite their broad deployment in safety-sensitive domains such as clinical diagnostic support and fraud detection. TabPFN is a recently proposed foundation model for classification problems that leverages in-context learning, obtaining state-of-the-art performance without task-specific training. Existing work on TabPFN's robustness is limited to single perturbation budgets and non-gradient-based attack methods. This thesis investigates the local adversarial robustness of TabPFN V2.0 on numerical binary classification tasks, comparing it against logistic regression, SVMs, and both standard and adversarially trained MLPs. We evaluate robustness using FGSM and PGD attacks across a range of perturbation budgets on five synthetic datasets and the Wisconsin Breast Cancer dataset, and additionally compute empirical robustness distributions to provide a finer-grained vulnerability assessment. Our central finding is that TabPFN exhibits notably higher robustness than the baselines under all direct attack methods on the Wisconsin dataset, but that this apparent advantage disappears once we utilize adversarial examples crafted on logistic regression and SVM, which induce misclassification in TabPFN at comparable rates. To assess the generalizability of the transfer findings, we repeat the transferability experiment on four additional real-world datasets, showing the observed pattern is not universal, but does recur in several settings. We further discover that TabPFN is more robust than all baselines on synthetic data under single-step FGSM attacks for moderate-to-large perturbation budgets, but that this advantage largely disappears under iterative PGD, suggesting that TabPFN's gradient landscape obstructs single-step attacks. Robustness distributions further reveal that vulnerability increases with feature dimensionality. These findings deliver the first distribution-level robustness analysis of TabPFN, guiding further investigation and highlighting the need for formal verification techniques.

Contents

1 Introduction

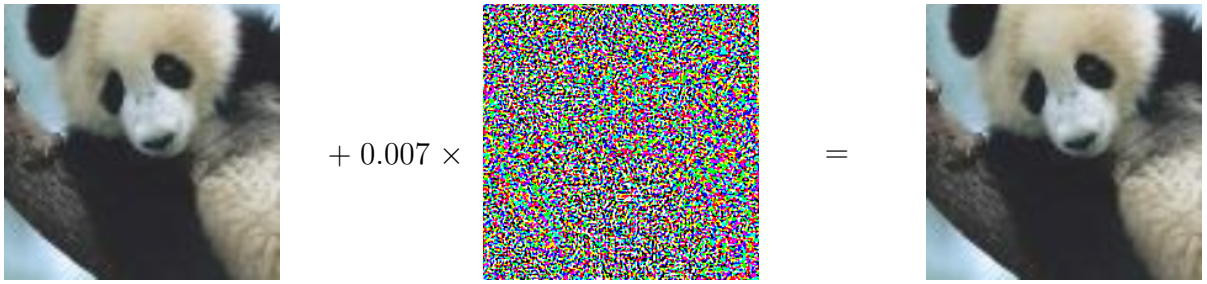
Performance of automated classifiers has become increasingly important, with applications in autonomous driving [CJB+24], medical imaging and diagnostics [LFK+19], and face recognition [OC21]. Deep neural networks have achieved human-level accuracy in such classification tasks. However, minimal perturbations to an input can lead to misclassifications by the network. Models are vulnerable to such perturbations, called adversarial attacks [SZS+14]. For example, in autonomous driving systems, minor perturbations to a stop sign can lead to misclassifications, potentially resulting in dangerous traffic violations and accidents.

Existing research on adversarial attack vulnerability has focused on image classifiers, where several methods have been developed to construct adversarial examples heuristically. Two prominent examples are: the Fast Gradient Sign Method (FGSM) [GSS15], a single-step gradient-based approach, and Projected Gradient Descent (PGD) [MMS+18], an iterative extension of FGSM. Models that achieve strong performance on standard benchmarks are susceptible to such attacks, as illustrated in Figure 1. Alongside heuristic methods, formal verification techniques have been proposed to characterize a model's vulnerability to adversarial perturbations with guarantees (e.g., [KBD+17, TXT19]). These approaches encode the model and its input constraints as a formal optimization problem and determine whether any perturbation within a specified bound can induce misclassification. For instance, piecewise-linear models can be encoded as a mixed-integer linear program, solvable via the simplex method. Unlike empirical attacks, such methods can provide provable certificates of robustness, though they typically scale poorly to large or architecturally complex models.

While research has focused on image classification, robustness guarantees are equally critical for automated classifiers operating on other data types. In particular, tabular classifiers remain underexplored from a robustness perspective. Consider, for example, clinical decision-support systems based on electronic health records, where minor perturbations in laboratory values or patient attributes could lead to inappropriate treatment decisions due to incorrect classifications.

Motivated by the disparity between the practical importance of adversarial robustness for tabular models and the limited attention it has received in the literature, we investigate the local robustness properties of TabPFN V2.0, a recently proposed transformer-based classifier designed for tabular data [HMEH23]. Unlike conventional supervised learning approaches that train a separate model for each dataset, TabPFN is pre-trained once on a large collection of synthetically generated tabular datasets. The model uses in-context learning, in which it receives both labeled training and unlabeled test samples as input to perform training and inference simultaneously.

TabPFN V2.0 constitutes an advance in tabular machine learning by obviating dataset-specific hyperparameter tuning while attaining state-of-the-art predictive performance. Its recommended operating regime is limited to datasets with at most 10,000 samples, 500 features, and 10 classes, beyond which performance may deteriorate. Within this regime, its predictive performance has been validated on TabArena, a continuously maintained benchmark for tabular machine learning, where TabPFN has been shown to perform particularly well on smaller datasets.



x
"panda"
57.7% confidence

$\text{sign}\big(\nabla_x J(\theta, x, \lambda(x))\big)$
"nematode"
8.2% confidence

$x + \varepsilon\ \text{sign}\big(\nabla_x J(\theta, x, \lambda(x))\big)$
"gibbon"
99.3% confidence

Figure 1: Example of an FGSM perturbation: original input, signed gradient perturbation, and resulting adversarial example, obtained from [GSS15].

Before exploring the robustness of TabPFN via FGSM and PGD attacks on real and synthetic data, this work first investigates the computational cost of these attacks on TabPFN and other machine classification models. Having established the computational limits of our experimental setup, we proceed to perform a comparative robustness analysis of TabPFN against SVMs, logistic regression, and both adversarially and non-adversarially trained MLPs on synthetic data. Then, to probe the effect of feature dimensionality on robustness, we find robustness distributions of TabPFN across different feature dimensions. To conclude our analysis, we turn to a real-world dataset, the Wisconsin Breast Cancer dataset [WMS93], and compare TabPFN against SVMs and logistic regression, by inspecting the robustness distribution of TabPFN in this more realistic setting. Furthermore, we examine the transferability of adversarial examples between these models on the same dataset, as well as on Haberman's Survival [Hab76], Connectionist Bench [GS88], Spambase [HRFS99], and Ionosphere [SWHB89].

Organization Section 2 outlines relevant research to this thesis. Section 3 introduces all necessary concepts, starting by formally defining local robustness, followed by a formal introduction of FGSM and PGD, robustness distributions, and all automated classifiers used in this work. Section 4 details the experimental setup. Section 5 follows by reporting on two experiments regarding computational scaling. The results of the robustness experiments are written in Section 6. Finally, Section 7 concludes with a concise summary of the results and suggests directions for further research.

Main Results Our experiments show that TabPFN consistently exhibits lower vulnerability than logistic regression, SVMs, and MLPs under single-step FGSM attacks on synthetic data, but that this advantage largely disappears under the stronger iterative PGD attack, suggesting that TabPFN's gradient landscape obstructs single-step attacks without conferring true robustness. On the Wisconsin Breast Cancer dataset [WMS93], TabPFN retains a substantial robustness advantage under both attack types. However, transferability analysis reveals that adversarial examples crafted on logistic regression and SVM induce misclassification in TabPFN at near-perfect rates, demonstrating that this apparent advantage reflects a limitation of direct gradient-based attacks rather than a genuine structural property of the model. Robustness distributions further reveal that vulnerability increases with feature dimensionality, suggesting that TabPFN's robustness

advantage under direct attack methods on the real-world dataset is not a consequence of its lower dimensionality relative to the synthetic datasets. To assess the generalizability of these transfer findings, we repeat the transferability experiment on four additional real-world datasets. The transfer pattern is not universal, but recurs in several settings: Spambase closely replicates the Wisconsin result, Connectionist Bench shows a similar pattern mainly at larger perturbation budgets, and Ionosphere produces more mixed and relatively symmetric transfer behavior.

Code Availability All code used to produce the experiments and figures in this thesis is publicly available at https://github.com/WillemScholtenLeiden/robustness-of-TabPFN.

2 Related Work

Because TabPFN has only recently been proposed, research on its robustness remains limited. Work on the subject focuses on tree-based ensembles, leaving gradient-based machine learning methods unexplored. Below, we present the most current research.

Djilani et al. [DST+25] present the first comprehensive evaluation of the adversarial robustness of tabular foundation models, evaluating TabPFNv2 and TabICL. They use the TabularBench benchmark with different attack mechanisms on datasets from finance, cybersecurity, and healthcare to demonstrate that both foundation models are more vulnerable to evasion attacks than the best-performing specialized models. The authors show that most classical tree-based models are effective surrogates for attacking tabular foundation models, and that on some datasets, transfer attacks from classical models are even more effective than direct white-box attacks on the foundation model. Conversely, tabular foundation models can also serve as surrogates, with transfer to tree-based models such as XGBoost and random forests sometimes exceeding what is achieved by other classical surrogates. However, both directions of transfer exhibit high variance across datasets and random seeds.

To mitigate these vulnerabilities, the authors propose Adversarial In-Context Learning (AICL), a training method that exploits tabular foundation models' in-context learning by iteratively replacing context samples with adversarially perturbed instances, while leaving the model weights unchanged. They show that AICL consistently outperforms standard adversarial fine-tuning for tabular foundation models. However, adversarially trained foundation models do not achieve the robustness of similarly trained specialized models.

Although this work is closely related to ours, it differs in several points. We constrain our analysis to binary classification of numerical data and consider gradient-based attack methods rather than tree-based ones. Most importantly, where their work evaluates robustness at a single perturbation budget, we extend the analysis to a range of budgets and additionally compute empirical robustness distributions across multiple datasets. This provides a more complete picture of how TabPFN's adversarial vulnerability varies with perturbation strength, rather than a single snapshot at one fixed budget.

3 Background

We now formally introduce all concepts required for this thesis. We begin with a mathematical definition of adversarial robustness, followed by the narrower formulation we adopt throughout this work. We then describe the FGSM and PGD attack methods, accompanied by pseudocode and a geometric illustration. Finally, we introduce robustness distributions and the classifiers considered in our experiments.

3.1 Local Robustness

We first present a formal definition of adversarial robustness, following the notation in [TXT19] closely. Consider a deep neural network classifier $f : \mathbb{R}^n \to \mathbb{R}^m$ with $n, m \in \mathbb{Z}_{\geq 1}$, that takes an input $x \in \mathbb{R}^n$, and outputs a score vector $s = f(x) \in \mathbb{R}^m$, where s_k is the score for class label $k \in \{1, \ldots, m\}$. Let $\mathcal{A}(x)$ denote the set of all allowable perturbations for an input x, let $\lambda(x)$ denote the true class label of input x, and let $\mathcal{X}_{valid}$ be the domain of valid inputs for f. The network f is said to be locally robust to perturbations on x_0 iff the following holds:

$$\forall x \in (\mathcal{A}(x_0) \cap \mathcal{X}_{valid}) : \arg\max_i f_i(x) = \lambda(x_0). \tag{1}$$

This definition is general in that it leaves open the choice of $\mathcal{A}(x)$ and $\mathcal{X}_{valid}$. Throughout this work, we take the domain of valid inputs $\mathcal{X}_{valid}$ to be all of $\mathbb{R}^n$, though we discuss alternatives in Section 7. For the allowed perturbation set, we consider ℓ_∞ balls of varying radius ε around x_0, so that (1) reduces to

$$\forall x \in \{x \in \mathbb{R}^n : \|x - x_0\|_\infty \leq \varepsilon\} : \arg\max_i f_i(x) = \lambda(x_0). \tag{2}$$

3.1.1 Heuristic Attack Methods

Various methods have been developed to assess local robustness in automated classifiers. Adversarial attacks constitute a relatively straightforward approach for assessing the robustness of gradient-based classifiers, whereby a small perturbation is applied to an input to induce a misclassification [SZS+14]. Such attacks leverage the gradient of the loss function with respect to the input to identify directions in the input space along which small perturbations yield the largest increase in classification error.

FGSM attack Among the simplest of such methods is the Fast Gradient Sign Method (FGSM) [GSS15]. Given a model with parameters θ, input x, target classification $\lambda(x)$, and cost function $J(\theta, x, \lambda(x))$, this process produces an adversarial example x_{adv} by calculating

$$x_{\text{adv}} = x + \varepsilon \cdot \text{sign}\left(\nabla_x J(\theta, x, \lambda(x))\right), \tag{3}$$

where $\nabla_x J(\theta, x, \lambda(x))$ is the gradient of the loss with respect to the input x, and $\varepsilon > 0$ controls the magnitude of the perturbation. The sign operator is applied element-wise, yielding a perturbation that maximally increases the loss under an ℓ_∞-norm constraint. The example in Figure 1 was created using FGSM, while the left panel in Figure 2 shows the process geometrically. The procedure is also described in Algorithm 1.

In the context of automated image classification, this method reliably produces adversarial examples that induce misclassification, as illustrated in Figure 1. Goodfellow et al. [GSS15] demonstrate that, for $\varepsilon = 0.25$, this procedure causes a shallow softmax classifier to exhibit an error rate of 99.9% on the MNIST test set, with an average confidence of 79.3% assigned to the incorrect labels. Under the same perturbation magnitude, a maxout network misclassifies 89.4% of the adversarial examples, with an average confidence of 97.6%. Similarly, for $\varepsilon = 0.1$, they report an error rate of 87.15% and an average confidence of 96.6% on adversarial examples generated for a convolutional maxout network evaluated on a preprocessed version of the CIFAR-10 dataset.

Algorithm 1: Fast Gradient Sign Method (FGSM)

Input: Model parameters θ, input x, true label $\lambda(x)$, perturbation magnitude $\varepsilon > 0$, loss function J

Output: Adversarial example x_{adv}

$g \leftarrow \nabla_x J(\theta, x, \lambda(x))$;

$x_{\text{adv}} \leftarrow x + \varepsilon \cdot \text{sign}(g)$;

return x_{adv}

Note: $\nabla_x J(\theta, x, \lambda(x))$ *denotes the gradient of the loss w.r.t. input* x

PGD attack A second procedure that builds on the same concepts as FGSM is the Projected Gradient Descent (PGD) attack [MMS+18]. PGD is an iterative extension of FGSM, and refines an adversarial input by repeatedly ascending the loss landscape while projecting the proposed perturbation onto the allowed region.

Given a network with parameters θ, an input $x \in \mathcal{X}_{valid}$ with true label $\lambda(x)$, and a perturbation budget $\varepsilon > 0$, the PGD attack seeks an adversarial example via iterative projected gradient ascent. Starting from an initial point $x^0 \in \mathcal{A}(x)$, the update rule at iteration t is given by

$$x^{t+1} = \Pi_{\mathcal{A}(x) \cap \mathcal{X}_{\text{valid}}} \left(x^t + \alpha \cdot \text{sign} \left(\nabla_x J \left(\theta, x^t, \lambda \left(x \right) \right) \right) \right), \tag{4}$$

where $\alpha > 0$ is the step size, and $\Pi_{\mathcal{S}}(\cdot)$ denotes the projection operator onto the set $\mathcal{S}$. The projection ensures that each iterate remains both within the allowable perturbation region and the domain of valid inputs. After a fixed number of iterations T, the final iterate x^T is returned as an adversarial example. The procedure is geometrically represented in the right panel of Figure 2. To reduce sensitivity to the starting point, we repeat this iterative procedure over R random restarts, where each initial point x^0 is drawn uniformly at random from the allowable perturbation region $\mathcal{A}(x)$, and retain the adversarial example that achieves the highest loss. This complete process is described in Algorithm 2.

While adversarial attacks are effective at demonstrating the existence of robustness violations, they do not provide formal guarantees regarding the absence of adversarial examples and therefore motivate the development of more rigorous verification-based approaches. Such techniques have been proposed for various linear [TXT19] and transformer-based architectures [BDBV21], but they remain computationally expensive, rendering them infeasible for large-scale models. Consequently, in this work, we focus on the aforementioned heuristic attack methods.

Algorithm 2: Projected Gradient Descent (PGD) Attack with Random Restarts

Input: Model parameters θ, input $x \in \mathcal{X}_{\text{valid}}$, true label $\lambda(x)$, perturbation budget $\varepsilon > 0$, step size $\alpha > 0$, number of iterations T, number of restarts R, loss function J

Output: Adversarial example x^*

$x^* \leftarrow x$;

for $r = 1$ **to** R **do**

  $\delta \sim \mathcal{U}(-\varepsilon, \varepsilon)^d$;

  $x^0 \leftarrow \Pi_{\mathcal{A}(x) \cap \mathcal{X}_{\text{valid}}}(x + \delta)$;

  for $t = 0$ **to** $T - 1$ **do**

    $g \leftarrow \nabla_x J(\theta, x^t, \lambda(x))$;

    $\tilde{x}^{t+1} \leftarrow x^t + \alpha \cdot \text{sign}(g)$;

    $x^{t+1} \leftarrow \Pi_{\mathcal{A}(x) \cap \mathcal{X}_{\text{valid}}}(\tilde{x}^{t+1})$;

  if $J(\theta, x^T, \lambda(x)) > J(\theta, x^*, \lambda(x))$ **then**

    $x^* \leftarrow x^T$;

return x^*

Note: $\nabla_x J(\theta, x, \lambda(x))$ *denotes the gradient of the loss w.r.t. input* x

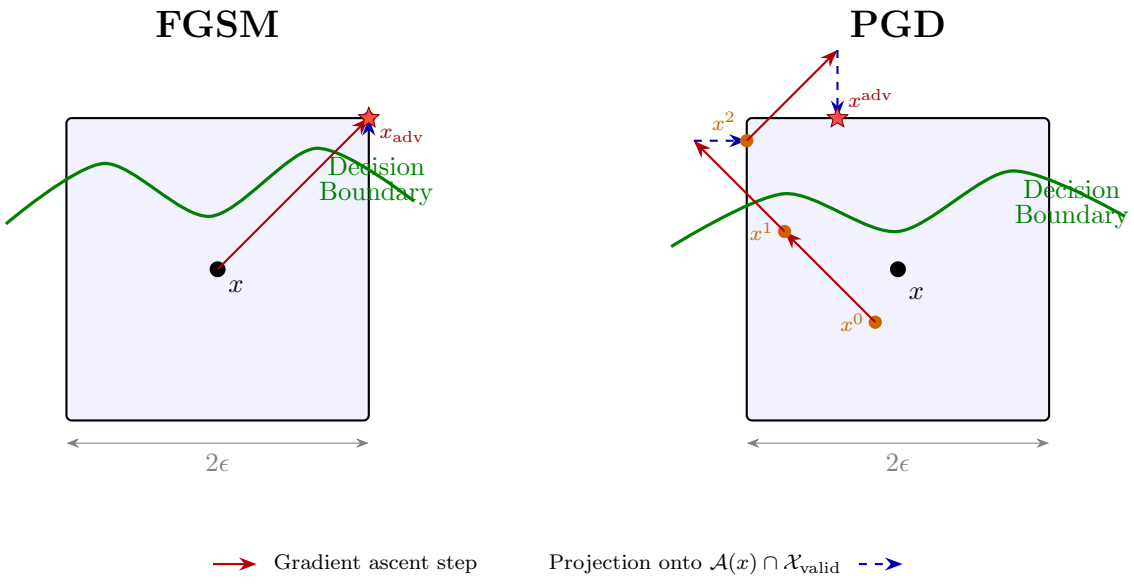


Figure 2: Geometric illustration of the FGSM and PGD methods in a two-dimensional input space. The blue region denotes $\mathcal{A}(x)$, which we assume fully lies in $\mathcal{X}_{\text{valid}}$. The green lines represent the decision boundary.

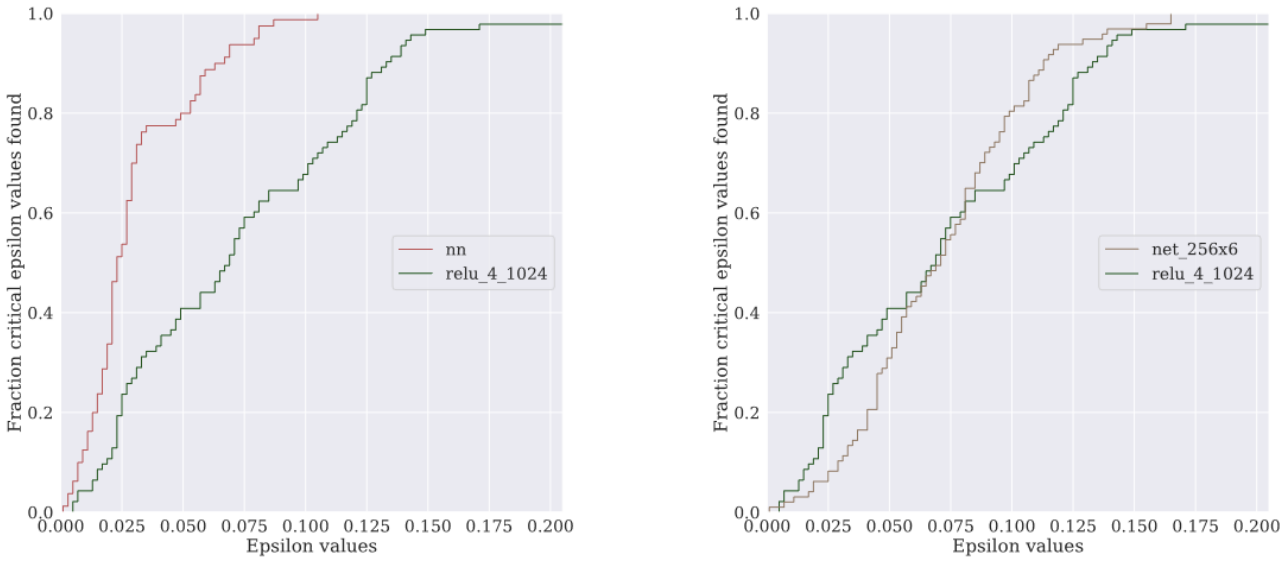


Figure 3: Example of empirical cumulative robustness distributions, obtained from [BBHvR25], plotted as the cumulative fraction of test samples for which an adversarial example is found below perturbation budget ε. The left panel shows that `nn` is substantially more vulnerable than `relu_4_1024`, as its distribution curve rises sharply at small ε. The right panel compares `net_256x6` and `relu_4_1024`, which exhibit broadly similar robustness profiles.

3.1.2 Robustness Distributions

Robustness distributions provide a way to visualize a classifier's adversarial robustness across an entire test set. We closely follow the definition and procedure introduced in [BBHvR25], where we obtained the example in Figure 3. For a given sample x_0, we define its robustness as the minimum perturbation budget ε^* for which an attack succeeds in inducing a misclassification. For samples that are misclassified on the clean input, we set $\varepsilon^* = 0$, since no perturbation is required to induce a misclassification. The robustness distribution is then obtained by computing ε^* for every sample in the test set and plotting the empirical cumulative distribution function. A curve that rises steeply at small ε indicates a highly vulnerable model, while one that remains low for large ε signals greater robustness.

In this work, we approximate ε^* by performing a binary search over a predefined search space. At each step, we run an adversarial attack at the current budget and narrow the interval accordingly, repeating until the search interval falls below a specified stopping margin δ, yielding an approximate $\tilde{\varepsilon} \approx \varepsilon^*$. Because we use heuristic attack methods rather than complete verifiers, it is not guaranteed that $\tilde{\varepsilon} - \varepsilon^* \leq \delta$, since the attack may fail to find an adversarial example even when one exists within the current budget. Consequently, $\tilde{\varepsilon}$ provides an upper bound on the true robustness ε^*, and the approximation error may exceed δ.

3.2 Machine Learning Classifiers

This thesis considers several automated classification methods, which we formally introduce below. We begin with TabPFN, describing its mathematical foundations, training procedure, and architecture. We then provide concise introductions to logistic regression, SVMs, and neural network classifiers [BWL+24].

3.2.1 TabPFN

TabPFN (Tabular Prior-Data Fitted Network) [HMEH23, HMP+25] is a transformer-based classifier designed specifically for learning from tabular data. Unlike conventional supervised learning approaches that train a separate model for each dataset, TabPFN is pre-trained once on a large collection of synthetically generated tabular datasets. The model leverages in-context learning, in which it receives both labeled training and unlabeled test samples as input and jointly performs training and inference.

Bayesian Inference The mathematical foundation for TabPFN lies in Bayesian modeling [MHP+22, Nag23]. In this framework, we assume there is a set of data-generating processes Φ that possibly generate our data $\mathcal{D}$, where the prior $p(\phi),\ \phi \in \Phi$ encodes the researcher's belief over which process gave rise to the observed data $\mathcal{D}$. Under this paradigm, prediction for a new data point x is obtained by marginalizing over all possible processes, yielding the posterior predictive distribution (PPD)

$$p(y \mid x, \mathcal{D}) \propto \int_{\Phi} p(y \mid x, \phi) p(\mathcal{D} \mid \phi) p(\phi) d\phi, \tag{5}$$

where the terms $p(\mathcal{D} \mid \phi)$ and $p(\phi)$ together form the posterior over ϕ via Bayes' rule. In most cases, it is impossible to find $p(y \mid x, \mathcal{D})$ in closed form. Classical approaches to approximate it, such as Markov Chain Monte Carlo or Variational Inference, are computationally expensive, particularly when repeated for each new dataset. This cost motivates the development of methods that can approximate the posterior predictive distribution more efficiently [MHP+22, Nag23].

PFNs TabPFN is a Prior-Data Fitted Network (PFN), a class of neural architectures specifically designed for predicting the PPD efficiently. A PFN is trained once on a large collection of synthetic datasets sampled from a prior $p(\phi)$. During training, a data-generating process ϕ is sampled from the prior, and a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ is then drawn from $p(\mathcal{D} \mid \phi)$. The network receives a subset of the dataset as context and is trained to predict the labels of the held-out query points, effectively learning to map any context-query pair $(\mathcal{D}_{\text{train}}, x)$ directly to an approximation of $p(y \mid x, \mathcal{D}_{\text{train}})$. The training loss is cross-entropy on the held-out set.

SCMs Two distinct data-generating paradigms were used for training the first version of TabPFN: Structural Causal Models (SCM) and Bayesian Neural Networks (BNN). TabPFN V2, however, relies exclusively on SCMs for data generation. An SCM consists of a directed acyclic graph (DAG) $\mathcal{G}$ over a collection of structural assignments $Z = \{z_1, \ldots, z_k\}$, where each variable is defined by a mechanism [Pea09]

$$z_i = f_i(z_{PA_{g(i)}}, \epsilon_i), \tag{6}$$

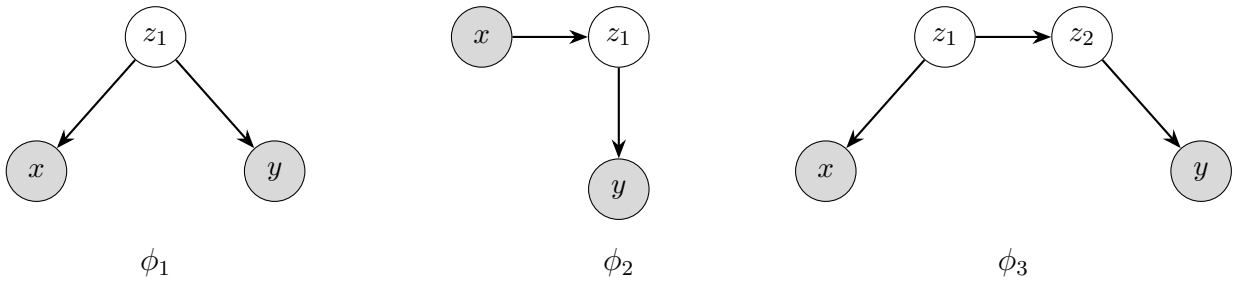


Figure 4: Three example SCMs. Shaded nodes denote observed variables, while white nodes represent latent variables. ϕ_1: a common latent cause generates both x and y. ϕ_2: x influences y through a latent mediator z_1. ϕ_3: a latent chain connects x and y indirectly through two hidden variables. Figure inspired by Figure 2 in [HMEH23].

where $PA_g(i)$ denotes the parents of node i in $\mathcal{G}$, f_i is a deterministic function and ϵ_i is a noise variable. To generate a synthetic dataset from this prior, a random SCM is first sampled, including its DAG structure, activation functions, edge weights, and noise distributions. A subset of nodes Z_X is assigned as observed features and a single node z_y as the target. For each sample in the dataset, noise variables are drawn and propagated through the graph.

As an example, consider the three SCMs from a hypothesis space Φ, illustrated in Figure 4. All three SCMs produce datasets $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ in which x and y are correlated, yet the underlying generating process, and therefore the optimal prediction rule, differs in each case. In the Bayesian framework underlying PFNs, ϕ_1, ϕ_2, and ϕ_3 are all elements of Φ, each weighted by its prior probability $p(\phi)$. Given a training dataset $\mathcal{D}$, the posterior predictive distribution implicitly assigns higher weight to whichever SCM, or family of SCMs, best explains the observed data.

Architecture The architecture of TabPFN, illustrated in Figure 5, is based on a modified transformer encoder. The input to the model consists of a tabular context: N labeled training samples $(\mathbf{x}_i, y_i)_{i=1}^N$ each with d features and a binary label, together with a query point $\mathbf{x}^*$ whose label is to be predicted. Each sample is represented as a row of $d+1$ values: the d features plus the label. A feature tokenizer maps each of these $d+1$ input columns independently into a k-dimensional embedding, producing one token per column per sample. This yields a set of $N+1$ token sequences that are fed into a stack of self-attention layers. The self-attention mechanism operates jointly over all $N+1$ samples, allowing the model to learn dependencies between the training context and the query point within a single forward pass. The final representation corresponding to the query point $\mathbf{x}^*$ is passed through a multilayer perceptron classification head, which outputs a predictive distribution over the binary class labels.

3.2.2 Baseline Models

We compare TabPFN against several classical classifiers, which we briefly introduce below.

Logistic regression To establish the robustness of TabPFN, we compare the model against various other methods. A baseline model for binary classification is logistic regression [Cox58, Bis06].

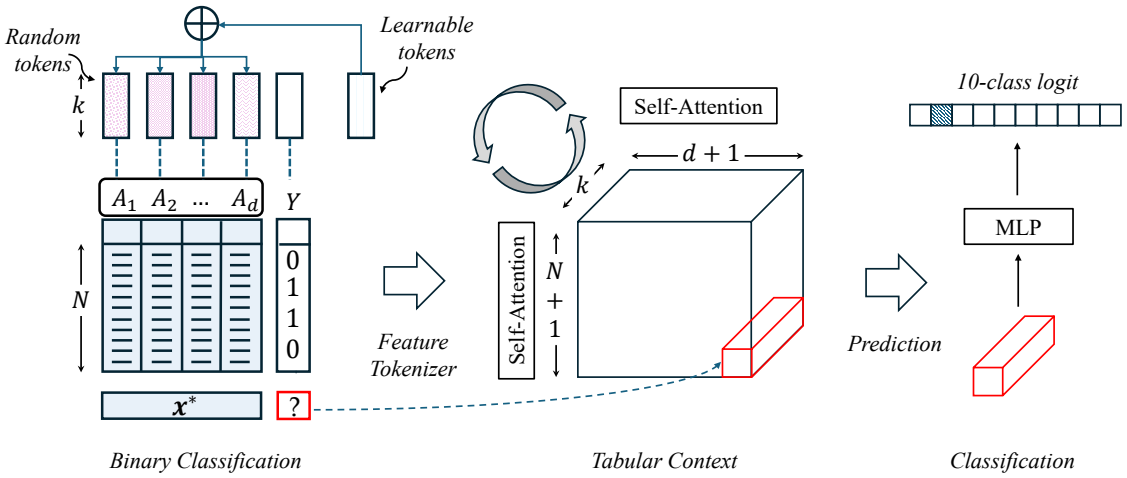


Figure 5: Overview of the TabPFN architecture. Input tabular data consisting of N training samples with d features and labels Y forms the tabular context. Each sample, along with a query point $\mathbf{x}^*$, is passed through a feature tokenizer that maps the d + 1 input columns into k-dimensional token embeddings. These tokens are then processed by a stack of self-attention layers, allowing the model to attend over both training samples and the query. The final representation of the query point is passed through an MLP to produce a classification prediction. This figure is obtained from [YLC25].

Logistic regression is a linear probabilistic classification model that estimates the conditional probability of a binary outcome given an input feature vector. Formally, given an input $x \in \mathbb{R}^d$, logistic regression models the probability of class membership as

$$P(y = 1 \mid x) = \sigma\left(w^\top x + b\right), \tag{7}$$

where $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ denote the model parameters, and $\sigma(\cdot)$ is the logistic sigmoid function. The predicted class label is obtained by thresholding this probability. Model parameters are learned by minimizing the negative log-likelihood of the observed labels, which is equivalent to minimizing the binary cross-entropy loss over the training data. As a linear classifier, logistic regression induces a linear decision boundary in the input space, making it computationally efficient and well understood, while also serving as a strong baseline for evaluating the robustness properties of more complex models. For logistic regression, FGSM yields the exact optimal ℓ_∞-bounded adversarial perturbation [GSS15].

Support Vector Classifier A different model to compare against is the support vector classifier (SVC), a margin-based discriminative model that seeks to separate data points of different classes by constructing a decision boundary with maximum geometric margin [CV95]. In its linear form, given training data $\{(x_i, y_i)\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, the SVC determines a hyperplane parameterized by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ by solving a convex optimization problem that maximizes the margin between classes while penalizing misclassifications via a regularization parameter. The resulting classifier assigns labels based on the sign $w^\top x + b$.

Neural Network Classifier A simple neural network classifier extends linear models by introducing one or more nonlinear transformations of the input through the use of hidden layers [GBC16]. We employ a two-layer neural network comprising an input layer, two hidden layers, and an output layer that produces class scores or probabilities. Given an input $x \in \mathbb{R}^d$, the network computes a sequence of affine transformations followed by element-wise nonlinearities, enabling it to learn nonlinear decision boundaries in the input space. Model parameters are optimized by minimizing cross-entropy, using gradient-based optimization methods. These models can be hardened through adversarial training [GSS15], providing a meaningful baseline for assessing whether TabPFN exhibits comparable resilience without explicit robustness interventions.

4 Experimental Setup

In this section, we describe the hardware and software specifications used throughout our experiments. We also specify the standard model configurations and formally define the metrics used to quantify robustness.

Attack Model Across all experiments, we perform untargeted attacks under an ℓ_∞ perturbation constraint. Unless stated otherwise, the adversary maximizes the binary cross-entropy loss with respect to the true label. For the linear SVM baseline, FGSM and PGD are applied by perturbing in the direction of the weight vector. Unless stated otherwise, the FGSM attack uses the specified ε value directly, while the PGD attack is configured with $T = 5$ iterations and 5 random restarts, where we choose $\alpha = 2\varepsilon/T$.

Metrics Let $\hat{f}(x) = \arg\max_i f_i(x)$ denote the predicted class of model f, and let $\lambda(x_0)$ denote the true label. We quantify robustness with three metrics. First, Attack Success Rate (ASR), which quantifies among inputs the clean model classifies correctly, the fraction whose adversarial counterpart is misclassified:

$$\mathrm{ASR} = \Pr\Big[\hat{f}(x_{\mathrm{adv}}) \neq \lambda(x_0) \;\Big|\; \hat{f}(x_0) = \lambda(x_0)\Big]. \tag{8}$$

Second, Adversarial Accuracy (AA), which denotes the fraction of samples that are both correctly classified and robust to the attack:

$$\mathrm{AA} = \Pr\Big[\hat{f}(x_{\mathrm{adv}}) = \lambda(x_0)\Big]. \tag{9}$$

Last, transfer rate (TR), which calculates among adversarial examples that successfully induce misclassification in the source model $f^{(s)}$ (restricted to inputs originally classified correctly by it), the fraction that also induces a misclassification on the target model $f^{(t)}$:

$$\mathrm{TR} = \Pr\Big[\hat{f}^{(t)}(x_{\mathrm{adv}}) \neq \lambda(x_0) \;\Big|\; \hat{f}^{(s)}(x_{\mathrm{adv}}) \neq \lambda(x_0) \wedge \hat{f}^{(s)}(x_0) = \lambda(x_0)\Big]. \tag{10}$$

Robustness Distributions For each test sample, the robustness ε^* is approximated via binary search over the interval $[0, 4]$ with some heuristic attack function. The search terminates once the interval width falls below a stopping margin of $\delta = 1 \cdot 10^{-3}$, yielding an approximation $\widetilde{\varepsilon} \approx \varepsilon^*$. Samples misclassified on the clean input are assigned $\varepsilon^* = 0$. Because the attack methods are heuristic, $\tilde{\varepsilon}$ constitutes an upper bound on the true robustness. The robustness curve is then obtained by plotting the empirical cumulative distribution function of $\tilde{\varepsilon}$ over the test set.

Experimental Environment The computational scaling experiments are performed using the ALICE compute resources provided by Leiden University, under a SLURM allocation of 8 CPUs and 16 GiB RAM, on a node equipped with an NVIDIA L4 GPU (23 GB VRAM), running RHEL 9.6 with NVIDIA driver 580.82.07. All other experiments are conducted on the shared vibranium.liacs.nl server, equipped with an Intel Xeon Silver 4214R CPU (48 cores, 2.40 GHz), 251 GB RAM, and two NVIDIA GeForce RTX 3090 GPUs (24 GB VRAM each), running CUDA 13.1 and Python 3.10.20. Models were implemented using PyTorch 2.10.0 (CUDA 12.8), scikit-learn 1.7.2, TabPFN 6.3.2,

and NumPy 2.4.2. TabPFN V2.0 is used in all experiments. We chose this version over more recent releases for two reasons. First, it is the most widely validated version: the introductory paper has accumulated over 400 citations, and the repository has been downloaded more than 2,000,000 times. This broad adoption, combined with its more permissive license, makes it the most suitable choice for reproducible and accessible research. Second, given the rapid pace of development, TabPFN V2.6 was released as recently as March 24, 2026; we leave the evaluation of newer versions to future work.

5 Computational Scaling Experiments

Before turning to the main objective of this thesis, we first investigate the computational limits of our setup. PGD attacks are notoriously expensive on image classifiers [MMS+18], and it is therefore important to establish which experiments are feasible within practical time constraints. To this end, we detail two scaling experiments. The first examines how computation time grows with feature dimension, the second how it scales with dataset size. These experiments serve to establish the computational limits to guide the architecture of the other experiments.

Feature dimensionality scaling As a preliminary experiment, we assess the computational feasibility of our attack methods given the available resources. Specifically, we measure how the runtime of FGSM and PGD attacks scales with the dimensionality of the feature space. For each attack method, we report the per-sample attack time, total runtime, and the ratio of FGSM to PGD execution time, alongside the corresponding TabPFN fitting time across all feature dimensions. We conduct this analysis across multiple runs on a single dataset and report mean runtimes with error bars to indicate variability.

We evaluate performance across feature dimensionalities of $2, 10, 20, 30, 40$, and 50 on a synthetic dataset comprising 60 samples. For the FGSM attack, we set $\varepsilon = 0.5$. For the PGD attack, we use a step size of $\alpha = 2\varepsilon/T$, with $T = 5$ iterations and 5 random restarts. All experiments are repeated 3 times, and the results are averaged to reduce variance. On the described hardware configuration, the full experiment completes in approximately 144 minutes. The results are shown in Figure 6.

Across all experiments, FGSM runtime at the smallest problem size deviates from the otherwise smooth trend and exhibits markedly larger error bars. We attribute this to one-time startup costs, which are paid on the first call to the model and therefore inflate FGSM's measured time disproportionately. The wide error bars at this point are consistent with this interpretation, as startup overhead varies between runs. Panel (a) reveals that per-sample attack time is approximately linear across all feature dimensionalities for both FGSM and PGD. Panel (b) confirms the same increasing trend for total runtime, indicating that higher feature dimensionality increases attack cost at the full-test-set level as well. Panel (c) shows that the empirical PGD/FGSM runtime ratio hovers around 16.5, well below the theoretical ratio of $steps \times restarts = 25$. We believe this discrepancy suggests that each standalone FGSM call incurs fixed overhead, e.g., model initialization or memory allocation, that is reduced when gradient steps are performed iteratively within PGD. Finally, panel (d) shows that TabPFN's fitting time is low, but varies slightly. Overall, these results suggest that feature dimensionality is a limiting factor for the computational feasibility of gradient-based adversarial attacks on TabPFN, and should be considered when designing experiments.

Dataset size scaling Second, we perform an analogous experiment to investigate how computational demand scales with dataset size. We report the same metrics as above, now varying the numbers of training and test samples while holding the feature dimensionality fixed, again averaging across three runs for each dataset and displaying error bars. We vary the number of samples across $24, 48, 72$, and 96, with a fixed feature dimensionality of 5. The FGSM attack uses $\varepsilon = 0.5$, while the PGD attack is configured with a step size of $\alpha = 2\varepsilon/T$, with $T = 5$ iterations, and 5 random restarts. On the described hardware configuration, the full experiment completes in 109 minutes

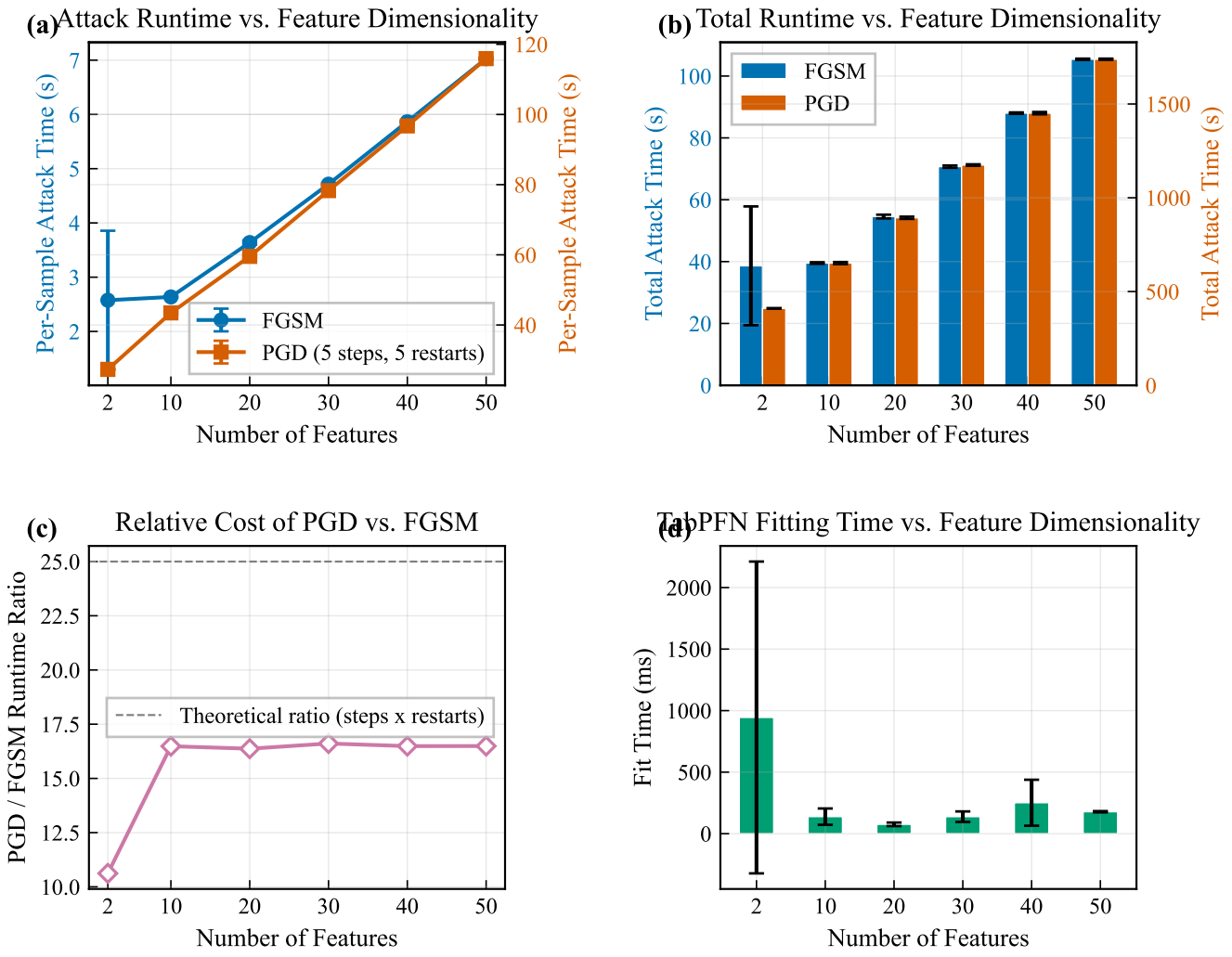


Figure 6: Runtime analysis of adversarial attacks on TabPFN across varying feature dimensionalities. (a) Per-sample attack time for FGSM and PGD (5 steps, 5 restarts), showing that both methods exhibit approximately linear cost with respect to the number of features. (b) Total attack time over the full test set, reflecting the same trend at scale. (c) Observed PGD/FGSM runtime ratio compared to the theoretical ratio of steps $\times$ restarts $= 25$. The empirical ratio remains consistently below this bound at approximately 16.5, suggesting non-negligible fixed overhead per FGSM call, which is reduced across PGD iterations. (d) TabPFN fitting time as a function of feature dimensionality, which is small, but varies. Error bars denote standard deviation over 3 repetitions.

and 27 seconds. The results are shown in Figure 7.

Again, FGSM runtime at the smallest problem size deviates from the otherwise smooth trend, which we attribute to the same reason. Panel (a) shows that the FGSM and PGD per-sample attack times grow steadily across dataset sizes. We believe this increase is attributable to the TabPFN design, which conditions on the entire training set at each forward pass, so increased context size increases computational demand. Panel (b) reflects both this increase and the increase in test samples to be attacked, both of which inflate the total runtime. Panel (c) shows that the PGD/FGSM runtime ratio remains below the theoretical bound of 25. Panel (d) reveals that TabPFN fitting time is again low, but appears to rise steadily after an initial peak at 24. The error bars are too large, however, for any definitive claims. Taken together, these results indicate that dataset size, such as feature dimensionality, has a meaningful impact on attack cost, and should be considered when designing larger-scale experiments.

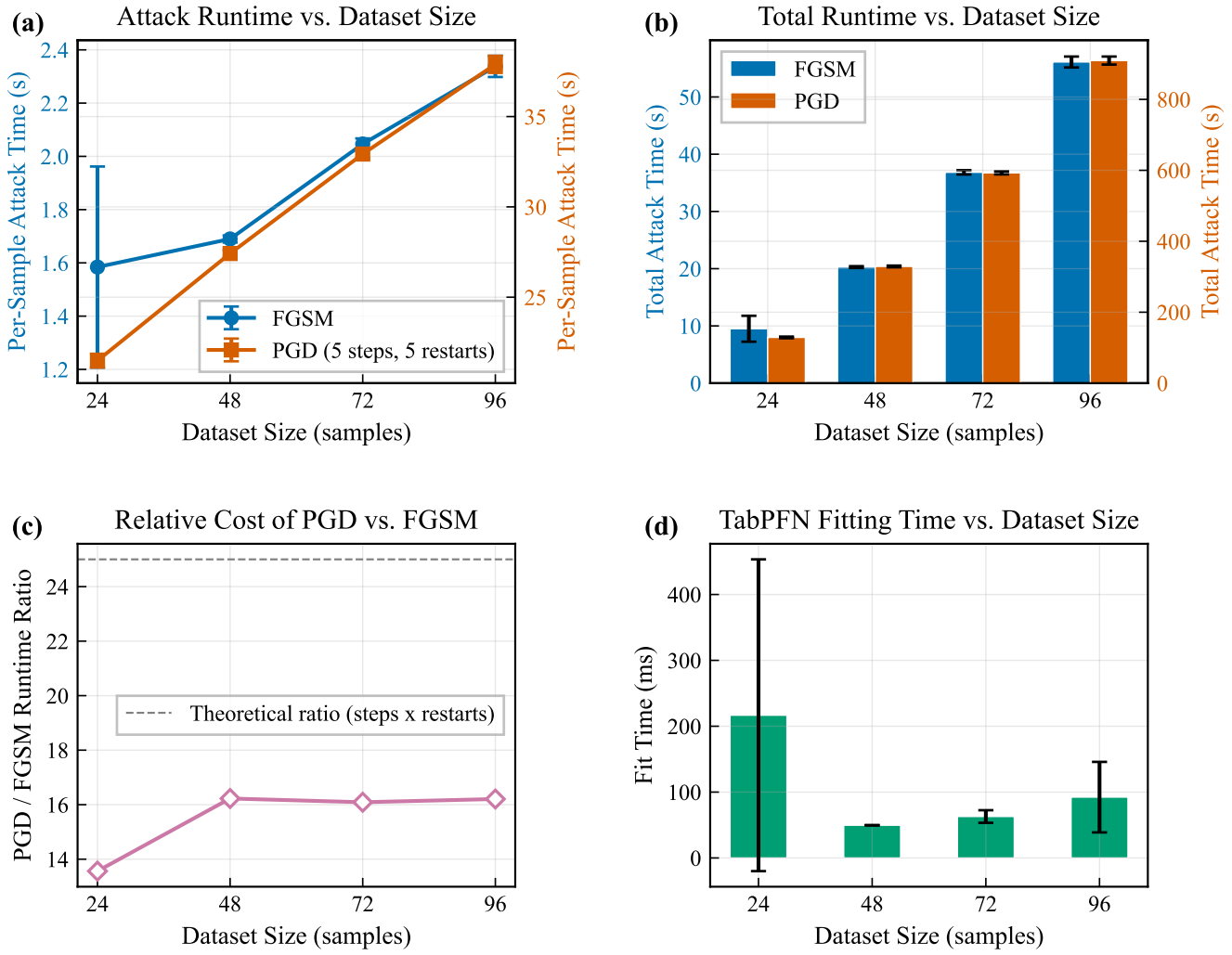


Figure 7: Runtime analysis of adversarial attacks on TabPFN across varying dataset sizes with a fixed feature dimensionality of 5. (a) Per-sample attack time for FGSM and PGD (5 steps, 5 restarts), showing that FGSM remains approximately constant while PGD exhibits a moderate increase for larger datasets. (b) Total attack time over the full test set, which grows with dataset size due to the increasing number of test samples. (c) Observed PGD/FGSM runtime ratio compared to the theoretical ratio of steps $\times$ restarts $= 25$. The empirical ratio remains below this bound, with a notable spike at 72 samples. (d) TabPFN training time as a function of dataset size, which remains stable for smaller datasets but increases sharply at 96 samples, suggesting a transition in the model's internal computation. Error bars denote standard deviation over 3 repetitions.

6 Robustness Evaluation Experiments

Having established the computational limits of our setup, we proceed to the central aim of this thesis. We first describe a comparative robustness analysis of TabPFN against SVMs, logistic regression, and both adversarially and non-adversarially trained MLPs on synthetic data, followed by an investigation of the robustness distribution of TabPFN across different feature dimensions. We then turn to real-world datasets, where we again compare TabPFN against SVMs and logistic regression by inspecting robustness distributions in this more realistic setting. Additionally, we investigate how well adversarial examples generated to induce misclassification in one model also succeed in inducing misclassification in a different model on these datasets.

6.1 Synthetic Datasets

In this section, we specify all experiments conducted on the synthetic datasets. All datasets are generated with scikit-learn's make classification method: `make_classification`. For each dataset, all features are informative (`n_informative = n_features`), with no redundant or repeated features (`n_redundant = 0`, `n_repeated = 0`), one cluster per class, and class separation `class_sep = 2.0`. Features are standardized to zero mean and unit variance using `StandardScaler`, fit on the training split only. Train-test splits are stratified by class label using a 75/25 split. Unless stated otherwise, we use 5 features.

Comparison With Classical Methods We now compare adversarial robustness across TabPFN, Logistic Regression, and a linear SVM. For different values of ε, we investigate the fraction of test samples where a misclassification is induced under both FGSM and PGD attacks for each model. This allows us to assess whether TabPFN is more or less vulnerable to adversarial perturbations than conventional classifiers. We report mean attack success rates across datasets for all five models, accompanied by error bars. Per-dataset breakdowns are provided in Appendix A.1. On the described hardware configuration, the full experiment completes in approximately 15 hours and 58 minutes. The results are shown in Figure 8.

At small perturbation budgets ($\varepsilon \leq 0.5$), all three models exhibit comparably low attack success rates under both FGSM and PGD, suggesting that minor input perturbations are insufficient to cross any of the learned decision boundaries. As ε increases beyond this threshold, however, a clear divergence emerges under FGSM: logistic regression and the linear SVM become substantially more vulnerable, while TabPFN maintains a notably lower ASR, indicating greater robustness to single-step gradient attacks. Under PGD, this robustness gap disappears, with all three models reaching similar ASR levels at large ε. This suggests that TabPFN's apparent resilience under FGSM may stem from suboptimal single-step attack directions, as the stronger iterative PGD attack is able to overcome this advantage.

Finally, we note that in Dataset 3 in Appendix A.1 under PGD, the monotonicity assumption is violated: ASR decreases for certain increases in ε. This non-monotonic behavior may result from PGD converging to suboptimal local perturbations at particular budget levels, which is expected since we use a heuristic attack method.

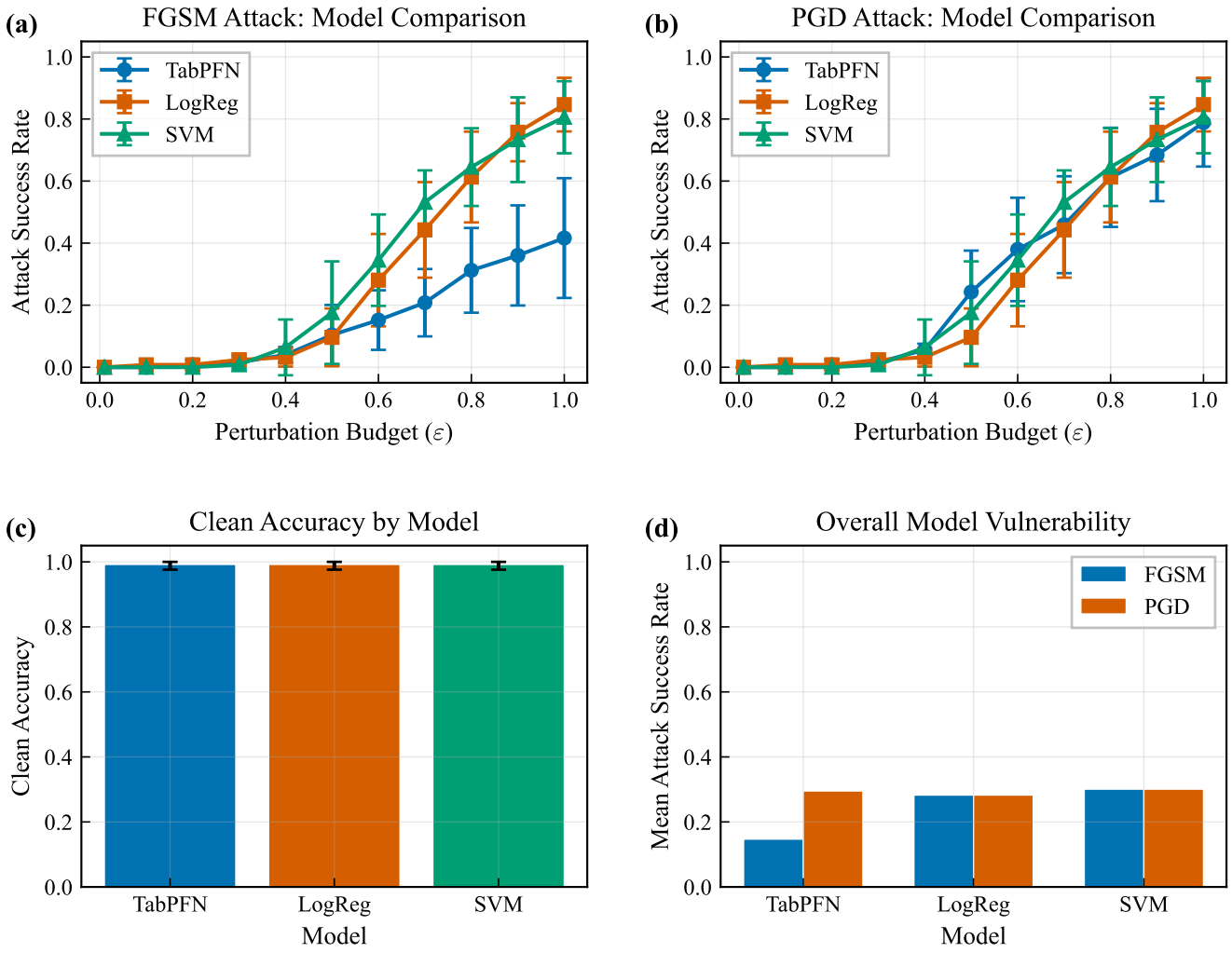


Figure 8: Comparative adversarial robustness of TabPFN, logistic regression, and a linear SVM on synthetic datasets. Panels (a) and (b) show the mean attack success rate as a function of the perturbation budget ε under FGSM and PGD, respectively, with error bars indicating one standard deviation across datasets. Panel (c) presents the FGSM results as a grouped bar chart for each ε value. Panel (d) summarizes overall model vulnerability by averaging the attack success rate across all perturbation budgets for both attack types. At small perturbation budgets ($\varepsilon \leq 0.5$), all three models exhibit comparably low attack success rates under both FGSM and PGD. For larger ε, TabPFN remains notably more robust than logistic regression and the linear SVM under FGSM, but this advantage disappears under the stronger iterative PGD attack, where all three models reach similar ASR levels.

Comparison With Neural Networks Next, we compare TabPFN against both a standard MLP and an adversarially trained MLP on three synthetic datasets. The MLP architecture consists of three linear layers with hidden dimensions 64 and 32, interleaved with ReLU activations, mapping 5 input features to 2 output classes. Both MLPs are trained for 100 epochs using the Adam optimizer with a learning rate of 10^{-3} and cross-entropy loss. The adversarially trained variant follows the approach of Madry et al. [MMS+18]. At each epoch, training samples are replaced by adversarial examples generated via 10-step PGD with perturbation budget $\varepsilon = 0.1$, step size $\alpha = 0.01$, and the ℓ_∞ norm. TabPFN is used in its pretrained form without any fine-tuning. PGD attacks use 10 steps, 5 random restarts with uniform initialization within the ℓ_∞-ball, and a step size of $\alpha = 2\varepsilon/10$. A dataset with 500 training and 25 test instances was used to allow for on-par performance of the neural networks. This experiment investigates whether the robustness of TabPFN, which is not explicitly trained to resist adversarial perturbations, is comparable to that of a model specifically hardened through adversarial training. We report mean attack success rates with error bars, with per-dataset results deferred to AppendixA.2. The results are shown in Figure 9.

Under FGSM, TabPFN again exhibits the lowest ASR for larger perturbation budgets, while the standard MLP and adversarially trained MLP follow similar trajectories. Notably, adversarial training provides only a marginal benefit, with the adversarially trained MLP performing comparably to its standard counterpart, suggesting that the adversarial training regime may not have been sufficiently aggressive to confer meaningful robustness on these synthetic datasets. Under PGD, all three models become more vulnerable at large ε, but TabPFN displays a clear disadvantage up to approximately $\varepsilon = 0.7$. Beyond this point, the gap narrows, and the error bars overlap substantially. Panel (d) summarizes the overall picture: TabPFN exhibits the lowest mean ASR under FGSM, while the opposite is true under PGD.

Robustness Distribution of TabPFN To obtain a more fine-grained picture of TabPFN's adversarial vulnerability, we examine its robustness distribution as defined in Section 3.1.2. Specifically, we investigate how the distribution changes with feature dimension, considering datasets with 5, 15, and 25 features under both FGSM and PGD attacks, where we restrict the binary search to $[0, 4]$. On the described hardware configuration, the full experiment completes in approximately 19 hours and 6 minutes.

The results are shown in Figure 10. Panels (a) through (c) display the robustness distributions for 5, 15, and 25 features respectively, with each panel overlaying FGSM and PGD curves for both datasets. Panel (d) merges all PGD curves across feature counts and datasets to facilitate direct comparison. Under PGD, we see the curves shifting leftward as the number of features increases, indicating that TabPFN becomes more vulnerable to iterative attacks in higher dimensions, consistent with the larger perturbation space available to the attacker. FGSM does not exhibit the same trend. For the second dataset, the 15-feature curve is shifted further left than the 5-feature curve. However, all FGSM curves lie well to the right of their PGD counterparts, indicating that the single-step method often fails to find effective attack directions and is therefore unreliable for assessing the dimensional effect.

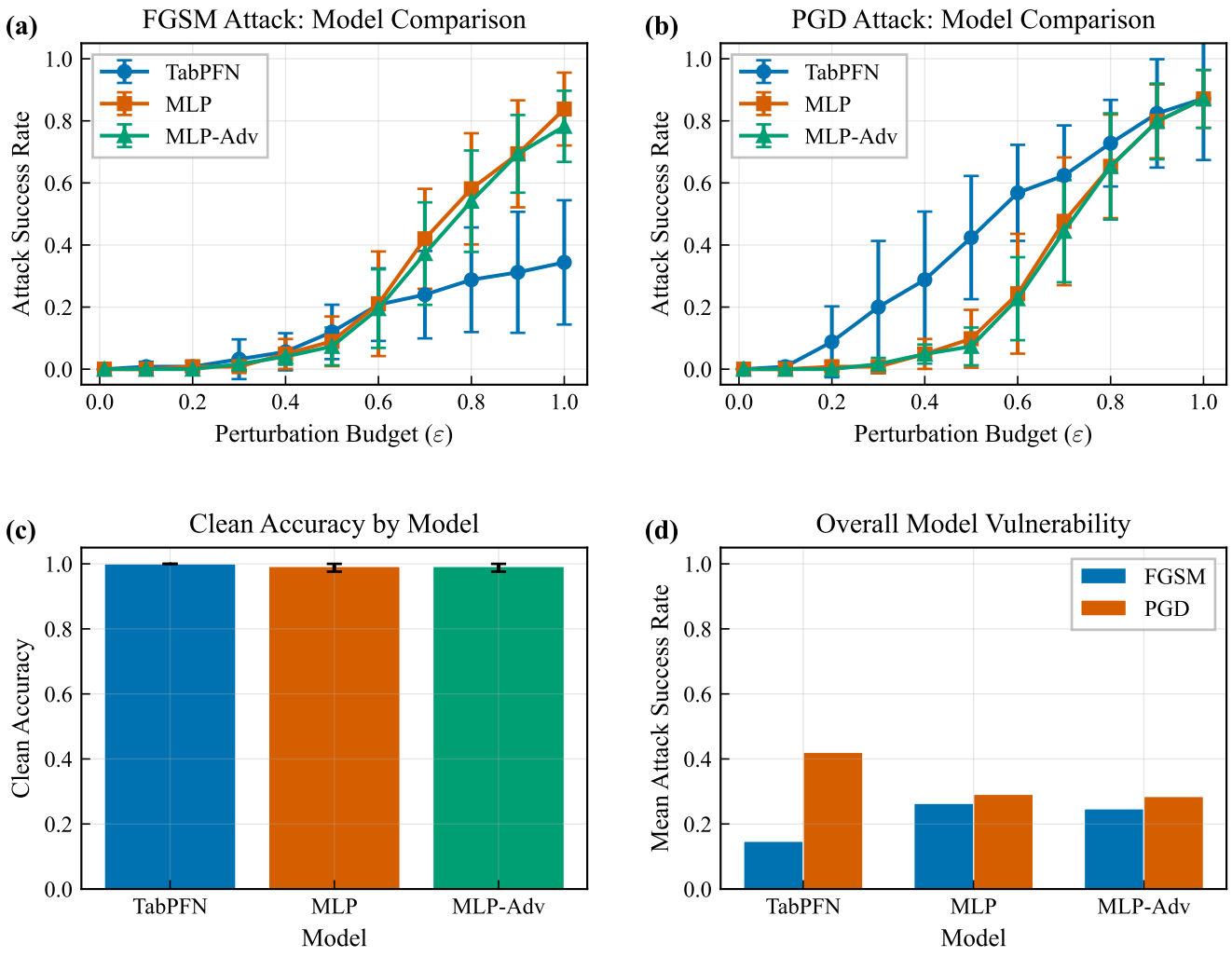


Figure 9: Comparative adversarial robustness of TabPFN, a standard MLP, and an adversarially trained MLP (MLP-Adv) on synthetic datasets. Panels (a) and (b) show the mean attack success rate as a function of the perturbation budget ε under FGSM and PGD, respectively, with error bars indicating one standard deviation across datasets. Panel (c) presents the FGSM results as a grouped bar chart for each ε value. Panel (d) summarizes overall model vulnerability by averaging the attack success rate across all perturbation budgets for both attack types.

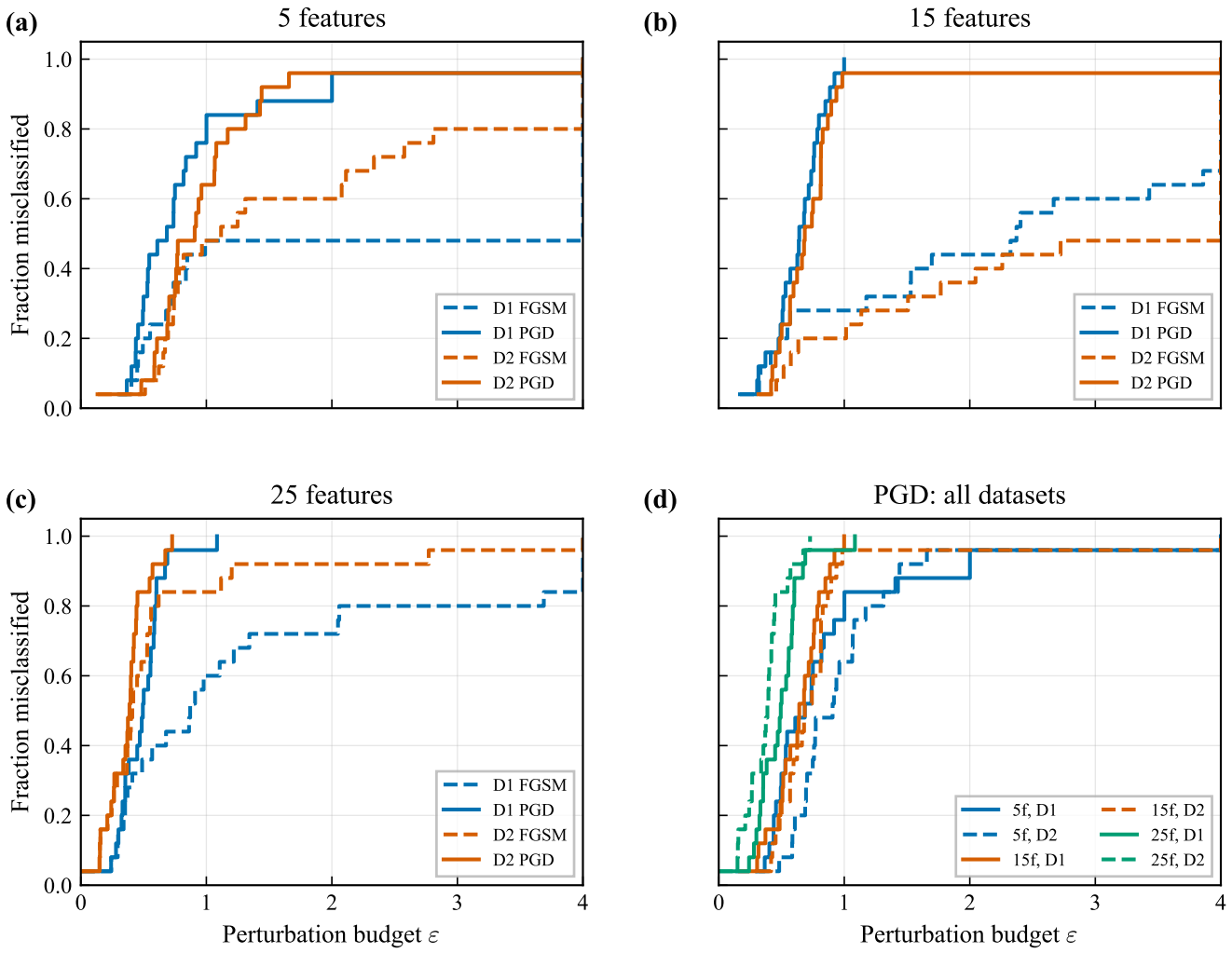


Figure 10: Robustness distribution of TabPFN for datasets with 5, 15, and 25 features. Panels (a) through (c) show the fraction of test samples misclassified as a function of the perturbation budget ε under FGSM and PGD attacks for both datasets D1 and D2. Panel (d) overlays all PGD curves across feature counts and datasets for direct comparison.

6.2 Real World Datasets

To see whether the results found on the artificial dataset hold for real-world data, we perform comparison experiments on several real-world datasets. We first use the Wisconsin Breast Cancer dataset [WMS93] to conduct a transferability experiment and investigate empirical robustness distributions across various setups. This dataset comprises 569 instances with 30 continuous features and binary class labels indicating benign or malignant tumors. Classifiers typically achieve near-perfect performance [Aga18] on this dataset, and the dataset is therefore useful for evaluating adversarial robustness in a real-world tabular classification setting where small perturbations might lead to clinically significant misclassifications. Next, we conduct preliminary transferability investigations on four additional real-world datasets: Haberman's Survival [Hab76], Spambase [HRFS99], Connectionist Bench [GS88], and Ionosphere [SWHB89], to inform future work aimed at establishing whether the findings from the Wisconsin experiments generalize.

6.2.1 In-Depth Analysis on the Wisconsin Breast Cancer Dataset

As a first experiment, we investigate the extent to which adversarial examples crafted to induce misclassification in one classifier transfer to the remaining classifiers. Specifically, we compute pairwise transfer rates between TabPFN, the linear SVM, and logistic regression on the Wisconsin Breast Cancer dataset, using the data processing regime described above and the transfer rate metric defined in Section 4. The results are shown in Table 1 for perturbation budgets $\varepsilon \in \{0.25, 0.5, 0.75, 1.0\}$.

Across all budgets, we observe extremely high transfer rates from both logistic regression and the linear SVM to TabPFN, consistently reaching into the nineties. Since adversarial examples crafted on these linear models reliably induce misclassifications on TabPFN, the perturbations required to induce misclassification in TabPFN clearly exist within the allowed perturbation region and are straightforward to identify via surrogate models, even when direct gradient-based attacks on TabPFN fail to locate them. This means a lower direct attack success rate does not necessarily reflect a structural property of the model, but may instead reflect a limitation of gradient-based attack methods when applied directly to TabPFN's loss landscape. In the reverse direction, transfer rates from TabPFN to the linear models are considerably lower across all budgets. Together with the high transfer rates from the classical models to TabPFN, this suggests that TabPFN may be less robust on this dataset than its classical counterparts: adversarial examples that induce misclassification in the classical models often also induce misclassification in TabPFN, whereas the converse does not necessarily hold, despite TabPFN's lower direct attack success rates.

Next, we extend the comparative robustness evaluation of TabPFN against classical methods to a real-world setting. By inspecting the empirical distribution curves of TabPFN, SVM, and logistic regression on the Wisconsin Breast Cancer dataset, we assess whether the robustness patterns observed on synthetic datasets generalize to a naturally occurring feature space. We use the previous experiment, however, to introduce two new attack methods. Guided by the high transferability rate from the two classical methods to TabPFN, we use adversarial examples generated to induce misclassification in one of these models on TabPFN. We use this method, PGD with 10 steps and 10 restarts for SVM, logistic regression, and TabPFN, and PGD with varying steps and restarts for TabPFN, to obtain empirical robustness distributions. The results are shown in Figure 11.

Wisconsin Breast Cancer ($n_{\text{test}} = 142$)					
ε	Source	TabPFN	LogReg	SVM	n_{succ}
0.25	TabPFN	100.0	58.3	75.0	12
	LogReg	94.9	100.0	100.0	39
	SVM	91.8	75.5	100.0	49
0.5	TabPFN	100.0	70.8	87.5	24
	LogReg	96.8	100.0	100.0	94
	SVM	95.0	91.1	100.0	101
0.75	TabPFN	100.0	69.0	83.3	42
	LogReg	95.9	100.0	100.0	122
	SVM	93.8	95.3	100.0	128
1	TabPFN	100.0	78.6	87.5	56
	LogReg	97.0	100.0	100.0	133
	SVM	96.3	99.3	100.0	134
	Accuracy	97.9	96.5	95.8	

Table 1: PGD (20 steps, 10 restarts) adversarial transfer rates (%) between TabPFN, logistic regression, and a linear SVM across perturbation budgets ε. Adversarial examples crafted on the linear models transfer to TabPFN at very high rates, consistently exceeding 90%, showing that successful perturbations exist within the allowed region even when direct attacks on TabPFN are less effective. In contrast, adversarial examples crafted on TabPFN transfer less reliably to the linear models. The column n_{succ} denotes the number of successful source-model attacks used to compute the corresponding transfer rates.

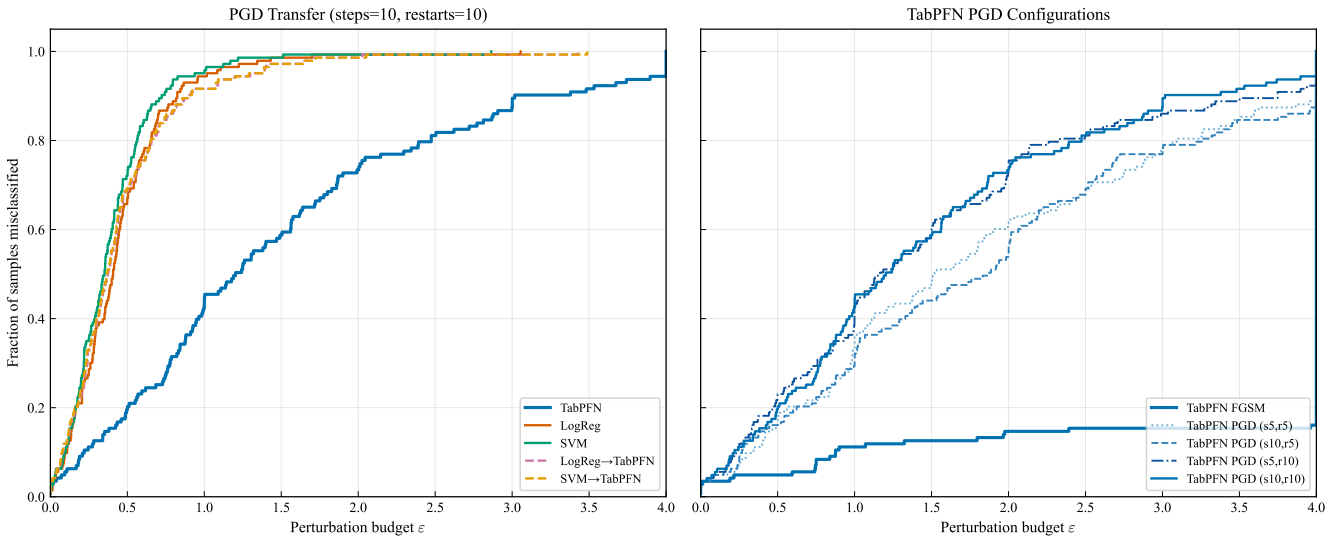


Figure 11: Empirical robustness distributions of TabPFN, logistic regression, and a linear SVM on the Wisconsin Breast Cancer dataset [WMS93], evaluated under a range of attack methods. Each curve shows the fraction of test samples misclassified as a function of the perturbation budget ε, restricted to the search space $[0, 4]$.

The left panel shows the robustness distributions under PGD (10 steps, 10 restarts) for all three models, as well as two transfer attack configurations in which adversarial examples crafted on logistic regression and SVM, respectively, are evaluated directly on TabPFN. TabPFN's direct PGD curve rises considerably more slowly than those of logistic regression and the linear SVM, indicating a substantially lower fraction of misclassified samples at any given perturbation budget. From the transfer attack curves, we see that adversarial examples generated to induce misclassification in logistic regression or the SVM succeed in misclassifying TabPFN at rates similar to those of the direct attacks on the linear models themselves. This suggests that adversarial perturbations sufficient to misclassify TabPFN inputs exist and are readily found indirectly, even when direct gradient-based attacks on TabPFN fail to locate them. This highlights the importance of formal verification techniques: attack-based robustness curves provide only an upper bound on true robustness, since failure to find an adversarial example does not certify that none exists within the perturbation region.

The right panel isolates TabPFN and compares several attack configurations. The FGSM curve rises most slowly and plateaus well below the PGD curves, confirming that a single gradient step is insufficient to find an adversarial example. Among the PGD configurations, increasing the number of restarts from 5 to 10 yields a more consistent improvement than increasing the number of steps alone, suggesting that TabPFN's attack landscape is sensitive to initialization and that random restarts are a more effective means of escaping suboptimal local perturbations. Even the strongest configuration does not reach the misclassification rates achieved by the transfer attacks in the left panel, further underscoring that the gradient landscape of TabPFN constitutes a structural obstacle for direct iterative attacks rather than a reflection of genuine adversarial robustness.

6.2.2 Preliminary Transferability on Additional Datasets

Next, we validate the generalizability of the results found on the Wisconsin dataset by repeating the transferability experiment on four different real-world datasets: Haberman's Survival [Hab76], Spambase [HRFS99], Connectionist Bench [GS88], and Ionosphere [SWHB89]. Because of the size of the datasets and the limited computational resources available, we use FGSM rather than PGD to attack TabPFN. This makes the preliminary transferability experiments computationally feasible, while still allowing us to assess whether adversarial directions found for one model induce misclassifications in the others. We again use a 75/25 train/test split, with the results reported in Table 2.

For Haberman's Survival, the transfer results should be interpreted cautiously because several source-model attacks succeed on only very few test points. Interestingly, this is the only dataset where, based solely on the direct attacks, TabPFN appears less robust than the classical models. At $\varepsilon = 0.25$, logistic regression yields only one successful attack and the SVM yields none, making the corresponding transfer rates uninformative. At larger budgets, adversarial examples crafted on the linear models often transfer to TabPFN, but the small values of n_{succ} again prevent strong conclusions. In contrast, attacks crafted on TabPFN transfer only weakly to the linear models, especially at $\varepsilon = 1$, where the transfer rates to logistic regression and the SVM are 16.7% and 19.4%, respectively. Thus, Haberman shows the same asymmetry as the Wisconsin experiment, but with substantially weaker empirical support due to the small number of successful source-model attacks.

Spambase provides the clearest replication of the earlier Wisconsin results. Across all budgets, adversarial examples crafted on logistic regression and the linear SVM transfer to TabPFN at very high rates, ranging from 96.3% to 99.9%. At the same time, adversarial examples crafted directly on TabPFN transfer much less reliably to the linear models, with transfer rates only in the mid-fifties to mid-sixties. This again suggests that adversarial perturbations that induce misclassification in TabPFN are readily found through simple linear surrogate models, even though direct attacks on TabPFN produce fewer successful attacks than direct attacks on the linear models. As in the Wisconsin experiment, the low direct attack success rate for TabPFN therefore should not be interpreted as evidence of greater robustness.

The Connectionist Bench results are more budget-dependent. At $\varepsilon = 0.25$, transfer from the linear models to TabPFN is only around 50%, which is much lower than in the Wisconsin and Spambase experiments. However, this changes rapidly as the perturbation budget increases: at $\varepsilon = 0.5$, both linear models transfer to TabPFN at approximately 88%, and at $\varepsilon = 1$, both reach 100%. The reverse direction is harder to interpret because direct attacks on TabPFN succeed on very few points, especially for the smaller budgets. Nevertheless, the results suggest that TabPFN is not consistently protected against transferred linear-model attacks. Rather, the transferability of such attacks becomes evident once the perturbation budget is sufficiently large.

For Ionosphere, transfer from the linear models to TabPFN is weaker than in Spambase and Wisconsin at small budgets, but increases substantially with ε. At $\varepsilon = 0.25$, the transfer rates from logistic regression and the SVM to TabPFN are 45.5% and 31.3%, respectively. These increase to 80.3% and

73.7% at $\varepsilon = 0.5$, and to 84.8% and 82.3% at $\varepsilon = 1$. Unlike the other datasets, however, Ionosphere does not exhibit a strongly asymmetric transfer pattern. Transfer from TabPFN to the linear models is of comparable magnitude, ranging from 54.2% to 78.8%, and in some cases even exceeds transfer in the reverse direction. Thus, Ionosphere provides a more mixed picture: while linear-model attacks become increasingly effective against TabPFN as the budget grows, the transfer behavior does not clearly support the same one-sided vulnerability observed on Wisconsin and Spambase. A possible factor could be the gap in performance; the difference between the accuracy of TabPFN and the other two models exceeds ten percentage points in both cases, but further investigation is needed.

Taken together, these four datasets show that the strong asymmetry observed on the Wisconsin dataset appears in several settings, but is not universal. Spambase most closely matches the Wisconsin behavior, with consistently high transfer from the linear models to TabPFN and substantially weaker transfer in the reverse direction. Connectionist Bench and Ionosphere show the same pattern mainly at larger perturbation budgets, but Connectionist Bench and Haberman are too small, in terms of successful source-model attacks, to support a strong conclusion.

ε	Source	Haberman ($n_{\text{test}} = 77$)				Spambase ($n_{\text{test}} = 1151$)			
		TabPFN	LogReg	SVM	n_{succ}	TabPFN	LogReg	SVM	n_{succ}
0.25	TabPFN	100.0	20.0	40.0	5	100.0	55.9	55.4	404
	LogReg	100.0	100.0	100.0	1	96.5	100.0	98.7	825
	SVM	–	–	–	0	96.3	98.4	100.0	831
0.5	TabPFN	100.0	50.0	33.3	6	100.0	61.8	61.4	479
	LogReg	80.0	100.0	60.0	5	98.4	100.0	99.7	971
	SVM	100.0	100.0	100.0	1	98.0	99.0	100.0	981
1	TabPFN	100.0	16.7	19.4	36	100.0	66.5	66.2	556
	LogReg	100.0	100.0	100.0	7	99.9	100.0	99.9	1056
	SVM	100.0	83.3	100.0	6	99.9	100.0	100.0	1059
	Accuracy	70.1	71.4	68.8		96.4	93.1	93.2	

ε	Source	Connectionist Bench ($n_{\text{test}} = 52$)				Ionosphere ($n_{\text{test}} = 88$)			
		TabPFN	LogReg	SVM	n_{succ}	TabPFN	LogReg	SVM	n_{succ}
0.25	TabPFN	100.0	100.0	100.0	1	100.0	54.2	70.8	24
	LogReg	50.0	100.0	100.0	38	45.5	100.0	98.5	66
	SVM	46.3	90.2	100.0	41	31.3	91.0	100.0	67
0.5	TabPFN	100.0	50.0	50.0	4	100.0	66.0	66.0	50
	LogReg	88.4	100.0	100.0	43	80.3	100.0	98.7	76
	SVM	88.1	100.0	100.0	42	73.7	98.7	100.0	76
1	TabPFN	100.0	62.5	87.5	8	100.0	74.2	78.8	66
	LogReg	100.0	100.0	100.0	43	84.8	100.0	100.0	79
	SVM	100.0	100.0	100.0	42	82.3	100.0	100.0	79
	Accuracy	88.5	80.8	90.4		97.7	86.4	87.5	

Table 2: FGSM adversarial transfer rates (%) between TabPFN, logistic regression, and a linear SVM on Haberman's Survival, Spambase, Connectionist Bench, and Ionosphere datasets across perturbation budgets ε. Transfer from the linear models to TabPFN is very high on Spambase, becomes high at larger budgets on Connectionist Bench and Ionosphere, and is suggestive for Haberman. It is difficult to interpret Haberman and Connectionist Bench due to the small values of n_{succ}. The column n_{succ} denotes the number of successful source-model attacks used to compute the corresponding transfer rates.

7 Conclusions and Further Research

This thesis investigated the local adversarial robustness of TabPFN on numerical binary classification tasks. Using FGSM and PGD attacks, we compared TabPFN with logistic regression, a linear SVM, and both standard and adversarially trained MLPs on synthetic datasets. We also computed empirical robustness distributions for TabPFN on synthetic datasets with varying feature dimensions. Moving to real-world data, we studied transferability between classical models and TabPFN on the Wisconsin dataset and plotted empirical robustness distributions for both direct and transfer-based attacks. Finally, we repeated the transferability experiment on four additional datasets to assess the generalizability of the findings.

Findings Our experiments revealed three main findings. First, on synthetic data, TabPFN consistently exhibited lower attack success rates than all baselines under FGSM, but this advantage largely disappeared under the stronger iterative PGD attack. This suggested that TabPFN's gradient landscape obstructs single-step attacks without necessarily providing genuine adversarial robustness. Second, the empirical robustness distributions showed that vulnerability increases with feature dimensionality, consistent with the larger perturbation space available to the attacker in higher dimensions. Third, the real-world experiments showed that TabPFN's apparent robustness under direct gradient-based attacks should be interpreted with caution. On the Wisconsin Breast Cancer dataset, direct attacks suggest a substantial robustness advantage for TabPFN over logistic regression and the linear SVM. However, transferability analysis revealed that adversarial examples crafted on the classical models induce misclassification in TabPFN at high rates, indicating that perturbations capable of misclassifying TabPFN often exist within the allowed perturbation region even when direct attacks fail to find them. This conclusion was further supported by the empirical robustness distributions, where transfer-based attacks against TabPFN were substantially more effective than direct PGD.

The additional real-world datasets showed that this transfer pattern is not universal, but does recur in several settings. Spambase closely replicated the Wisconsin result, with consistently high transfer from the linear models to TabPFN and weaker transfer in the reverse direction. Connectionist Bench showed a similar pattern mainly at larger perturbation budgets, while Ionosphere produced a more mixed and relatively symmetric transfer behavior. Haberman's Survival and Connectionist Bench were difficult to interpret due to the small number of successful source-model attacks. These results emphasize that low direct attack success rates on TabPFN should not be taken as reliable evidence of superior robustness and therefore underscore the need for formal verification techniques.

Limitations Several limitations should be noted. Our analysis relies on heuristic attacks, which provide upper bounds on robustness rather than formal guarantees. The transferability experiments highlight that these can be misleading, indicating the need for formal approaches. The adversarially trained MLP baseline used a relatively mild training regime, which may explain its limited improvement over the standard MLP. Finally, we assumed strictly numerical data and no constraints on the input, whereas in practice, data can be of mixed types and valid inputs may be constrained to a bounded domain, which could affect both the attack strategies and the resulting robustness estimates. Furthermore, our analysis is restricted to smaller datasets because of limitations to computational resources.

Directions for future work Several directions for future research follow naturally from these results. The transferability experiments on the four additional real-world datasets should be repeated using PGD with the same configuration as in the Wisconsin experiment. This would improve comparability across datasets and make the results more robust. Second, it would be interesting to see how AICL, an adversarial training mechanism introduced in [DST+25], would affect the results. Once formal verification methods become available and computationally feasible for transformer architectures, they could complement heuristic attack-based analyses with provable robustness bounds, which the transferability experiments show are necessary.

Relaxing the constraints imposed on the data would bring the robustness analysis closer to practical deployment settings. This extension, however, requires two methodological adjustments. First, standard PGD is no longer sufficient once domain constraints or categorical variables are introduced, since perturbations must remain valid with respect to the underlying data. A natural next step is therefore to adopt CAPGD or CAA [SGC24], attack methods specifically designed for tabular models, which incorporate these constraints into the adversarial search procedure. Second, logistic regression and linear SVMs do not naturally operate on categorical features without an appropriate representation. Future work could address this either by applying a fixed encoding scheme, such as one-hot encoding, and constraining attacks to preserve valid category assignments, or by replacing these baselines with models that handle mixed numerical and categorical data more directly, such as tree-based classifiers. This would allow the transferability analysis to be extended to less idealized tabular settings while maintaining a meaningful comparison between TabPFN and classical baselines.

It would also be interesting to repeat the experiments using the most recent version of TabPFN, TabPFN-3 [GFK+26], to determine whether the observed robustness and transferability patterns persist under the updated architecture and inference procedure. In particular, future work could investigate whether TabPFN-3's improved predictive performance and scalability also translate into improved adversarial robustness, or whether the same discrepancy between direct and transfer-based attacks remains. If access to TabPFN's Fast Inference Mode is available, future work could extend the transferability experiments to its distilled MLP and tree-ensemble variants.

References

[Aga18] Abien Fred M. Agarap. On breast cancer detection: an application of machine learning algorithms on the wisconsin diagnostic dataset. In *Proceedings of the 2nd international conference on machine learning and soft computing*, pages 5–9, 2018.

[BBHvR25] Annelot W. Bosman, Aaron Berger, Holger H. Hoos, and Jan N. van Rijn. Robustness distributions in neural network verification. *Journal of Artificial Intelligence Research*, 83:20:1–20:41, 2025.

[BDBV21] Gregory Bonaert, Dimitar I. Dimitrov, Maximilian Baader, and Martin Vechev. Fast and precise certification of transformers. In *Proceedings of the 42nd ACM SIGPLAN international conference on programming language design and implementation*, pages 466–481, 2021.

[Bis06] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[BWL+24] Mitra Baratchi, Can Wang, Steffen Limmer, Jan N. van Rijn, Holger Hoos, Thomas Bäck, and Markus Olhofer. Automated machine learning: past, present and future. *Artificial intelligence review*, 57(5):122, 2024.

[CJB+24] Marcelo Contreras, Aayush Jain, Neel P. Bhatt, Arunava Banerjee, and Ehsan Hashemi. A survey on 3d object detection in real time for autonomous driving. *Frontiers in Robotics and AI*, 11:1212070, 2024.

[Cox58] David R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 20(2):215–232, 1958.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[DST+25] Mohamed Djilani, Thibault Simonetto, Karim Tit, Florian Tambon, Paul Récamier, Salah Ghamizi, Maxime Cordy, and Mike Papadakis. On the robustness of tabular foundation models: Test-time attacks and in-context defenses. *CoRR*, abs/2506.02978, 2025.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press Cambridge, 2016.

[GFK+26] Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, et al. TabPFN-3: Technical report. *CoRR*, abs/2605.13986, 2026.

[GS88] R. Paul Gorman and Terrence J. Sejnowski. Connectionist Bench (Sonar, Mines vs. Rocks). UCI Machine Learning Repository, 1988.

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[Hab76] S. Haberman. Haberman's Survival. UCI Machine Learning Repository, 1976.

[HMEH23] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

[HMP+25] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin T. Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025.

[HRFS99] Mark Hopkins, Erik Reeber, George Forman, and Jaap Suermondt. Spambase. UCI Machine Learning Repository, 1999.

[KBD+17] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International conference on computer aided verification*, pages 97–117. Springer, 2017.

[LFK+19] Xiaoxuan Liu, Livia Faes, Aditya U. Kale, Siegfried K. Wagner, Dun Jack Fu, Alice Bruynseels, Thushika Mahendiran, Gabriella Moraes, Mohith Shamdas, Christoph Kern, et al. A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis. *The lancet digital health*, 1(6):e271–e297, 2019.

[MHP+22] Samuel Müller, Noah Hollmann, Sebastian Pineda-Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*. OpenReview.net, 2022.

[MMS+18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[Nag23] Thomas Nagler. Statistical foundations of prior-data fitted networks. In *International Conference on Machine Learning*, pages 25660–25676. PMLR, 2023.

[OC21] Alice J. O'Toole and Carlos D. Castillo. Face recognition by humans and machines: three fundamental advances from deep learning. *Annual Review of Vision Science*, 7(1):543–570, 2021.

[Pea09] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge university press, 2009.

[SGC24] Thibault Simonetto, Salah Ghamizi, and Maxime Cordy. Constrained adaptive attack: Effective adversarial attack against deep neural networks for tabular data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[SWHB89] Vincent G. Sigillito, Simon P. Wing, Larrie V. Hutton, and Kile B. Baker. Ionosphere. UCI Machine Learning Repository, 1989.

[SZS+14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations, Conference Track Proceedings*, 2014.

[TXT19] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*. OpenReview.net, 2019.

[WMS93] William H. Wolberg, Olvi L. Mangasarian, and W. Nick Street. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 1993.

[YLC25] Han-Jia Ye, Si-Yang Liu, and Wei-Lun Chao. A closer look at TabPFN v2: Understanding its strengths and extending its capabilities. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2025.

A Detailed Results for Synthetic Robustness Experiments

This appendix presents the per-dataset results underlying the aggregated figures in Section 6.1. Each figure shows the attack success rate as a function of the perturbation budget ε for every individual synthetic dataset, complementing the mean and standard deviation reported in the main text.

A.1 Classical Methods Comparison

Figure 12 shows the per-dataset breakdown of the classical methods comparison. The aggregate trends reported in Section 6 are consistent across most datasets: TabPFN exhibits lower ASR than logistic regression and the linear SVM under FGSM, while all three models converge under PGD. Individual datasets occasionally deviate from this pattern, most notably Dataset 3, where a non-monotonic ASR under PGD is visible.

A.2 Neural Network Comparison

Figure 13 presents the per-dataset results for the MLP comparison. The individual datasets confirm the aggregate finding that TabPFN is less vulnerable to FGSM attacks, while it consistently performs worse under PGD. The adversarially trained MLP does not show a clear advantage over the standard MLP on any individual dataset.

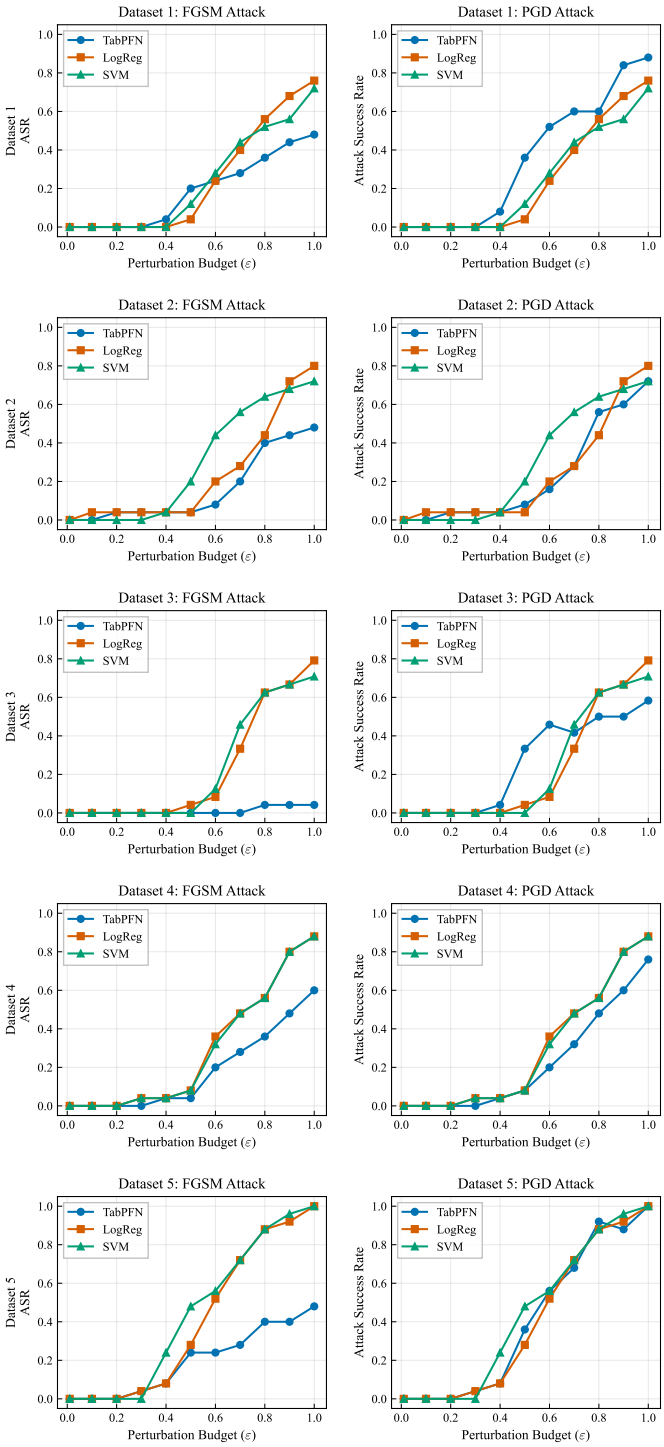


Figure 12: Attack success rate (ASR) as a function of perturbation budget (ε) for FGSM (left column) and PGD (right column, 5 steps, 5 restarts) attacks on TabPFN, logistic regression, and a linear SVM, evaluated across five synthetic datasets. TabPFN appears more robust for larger ε under FGSM, but all three models appear similar for all ε under PGD.

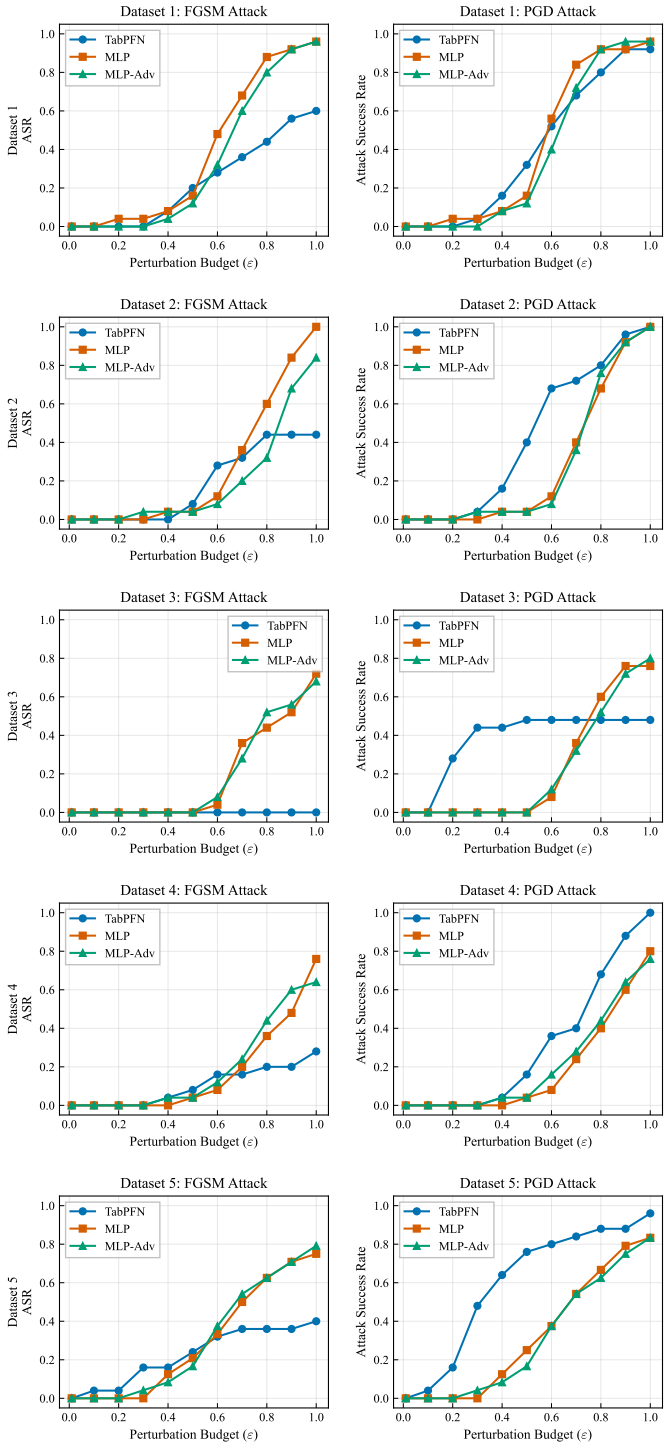


Figure 13: Attack success rate (ASR) as a function of perturbation budget (ε) for FGSM (left column) and PGD (right column, 5 steps, 5 restarts) attacks on TabPFN, MLP, and adversarially trained MLP evaluated across three synthetic datasets. TabPFN appears more robust for larger ε under FGSM, but less robust for smaller ε under PGD. There is no clear difference between the adversarially trained MLP and the normal MLP, possibly indicating that the training set is not large enough.