Adversarial Robustness Guarantees for Classification with Gaussian Processes

Arno Blaas*, Andrea Patane*, Luca Laurenti*,
Luca Cardelli, Marta Kwiatkowska and Stephen Roberts
University of Oxford
{arno,sjrob}@robots.ox.ac.uk
{andrea.patane,luca.laurenti}@cs.ox.ac.uk
{luca.cardelli,marta.kwiatkowska}@cs.ox.ac.uk

Abstract

Bayesian classification models have been conjectured to be more robust to adversarial attacks than deterministic ones [1]. Formal guarantees for the robustness of existing Bayesian models are however still outstanding. In this work, we consider Bayesian classification with Gaussian processes and derive theoretical and algorithmic results that can guarantee safety of a given Gaussian process classification model against adversarial examples for compact subsets of the input space in finite time. Applying our results on a synthetic dataset and two real-world datasets, we demonstrate that failure of adversarial attacks is not sufficient to guarantee safety.

1 Introduction

The impressive advancements in machine learning (ML) research in recent years have contributed to an increased adoption of ML methods in safety-critical application areas, such as healthcare [2] or autnomous driving [3]. Their application in such areas naturally calls for formal guarantees for their behaviour. One particularly important class of such guarantees is centered around the robustness of models when facing adversarial attacks, as successful adversarial attacks can lead to catastrophic outcomes in safety-critical applications [3]. In this work, we derive safety guarantees on the robustness against adversarial attacks for Bayesian classification with Gaussian processes (GPs) and demonstrate their validity and usefulness in practice.

More specifically, given a trained Gaussian process classification (GPC) model and a compact subset of the input space $T \subseteq \mathbb{R}^d$ (typically encompassing a correctly-classified test point), we pose the problem of computing the maximum and minimum of the class probabilities over all $x \in T$. An exact direct computation of these values is not possible, as it would require solving non-linear optimization problems, for which no general solving method exists [4]. We thus first derive suitable upper and lower bounds for the minimum and maximum classification probabilities of GPC models and then iteratively refine these approximations inside a branch and bound algorithmic scheme. We show that our algorithm converges to the true maximum and minimum values in finite time, ensuring that the results are ϵ-close to the true optima for any error $\epsilon > 0$.

The proposed framework is the first that exactly computes the range of *classification* probabilities for compact sets of input points (up to a quantifiable and controllable error $\epsilon > 0$) for GPC models and allows us to give $\epsilon-$exact guarantees against adversarial attacks. Our main contributions are:

- We pose the reachability problem for the Bayes optimal classifier for GP classification in order to obtain adversarial robustness guarantees.

NeurIPS 2019 Workshop on Machine Learning with Guarantees, Vancouver, Canada.

*Equal Contributions.

- We develop techniques for latent space discretisation and optimization, variance bounding, and a branch and bound algorithm with a finite-time convergence proof to solve the problem.
- We apply our approach to analyse GPC adversarial robustness on a synthetic dataset, as well as the MNIST38 and SPAM datasets, and compare it with adversarial attacks for GPCs [5].

2 Bayesian classification with Gaussian processes

Given a dataset $\mathcal{D} = \{(x, y) \,|\, x \in \mathbb{R}^d, y \in \{1, ..., C\}\}$ and a test point x^*, for a GPC model the probability $\pi^c(x^*|\mathcal{D})$ that x^* is asssigned to class c is given by:

$$\pi^c(x^*|\mathcal{D}) = \int \sigma^c(\bar{f}) p(f(x^*) = \bar{f}|\mathcal{D}) d\bar{f}, \tag{1}$$

where $f(x) = [f^1(x), ..., f^C(x)]$ is the latent function vector, $\sigma^c : \mathbb{R}^C \to [0, 1]$ is the link function for class c and $p(f(x^*) = \bar{f}|\mathcal{D})$ the predictive posterior distribution [6]. However, in general the posterior $p(f(x^*) = \bar{f}|\mathcal{D})$ is intractable and has to be approximated using either sampling methods or analytic approximations [6]. Most analytic approximations result in a Gaussian approximation $q(f(x^*) = \bar{f}|\mathcal{D})$ for $p(f(x^*) = \bar{f}|\mathcal{D})$ [7, 8, 9]. In what follows, we will thus work with $q(f(x^*) = \bar{f}|\mathcal{D})$ instead of $p(f(x^*) = \bar{f}|\mathcal{D})$, under the assumption that $q(f(x^*) = \bar{f}|\mathcal{D}) = \mathcal{N}(\bar{f} \,|\, \mu(x^*), \Sigma(x^*))$. The results presented in this paper do not depend on a particular approximation method as long as q is Gaussian.

Focus on the binary case. For the case $C = 2$ (binary classification) there are only two classes to distinguish and it suffices to compute $\pi(x^*|\mathcal{D}) = \int \sigma(\bar{f}) p(f(x^*) = \bar{f}|\mathcal{D}) d\bar{f} := \pi^1(x^*|\mathcal{D})$, with f being a univariate latent function and setting $\pi^2(x^*|\mathcal{D}) := 1 - \pi(x^*|\mathcal{D})$. As this leads to a significant reduction of computational complexity, the multiclass case ($C > 2$) is often reduced to the binary case via one-vs.-all classification [6],[10]. For simplicity of exposition, we thus focus on the binary classification problem; an extension to multiple classes is covered in Appendix E.

3 Adversarial robustness bounds for binary classification

3.1 Problem formulation

We focus the notion of adversarial robustness around the following definition of safety, which allows to decide whether adversarial examples exist in T, and can thus offer formal guarantees against attacks for GPC models:

Definition 1. *(Safety) Let $T \subseteq \mathbb{R}^d$ and $x^* \in T$. Then, we say that the classification of x^* is safe against adversarial attacks in T, iff* $\forall x \in T : \arg\max_{c \in \{1,2\}} \pi^c(x) = \arg\max_{c \in \{1,2\}} \pi^c(x^*)$.

3.2 Outline of proposed approach

To evaluate safety of a binary GPC model at $x^* \in T$, we want to compute the values of

$$\pi_{\min}(T) := \min_{x \in T} \pi(x \,|\, \mathcal{D}) \quad \pi_{\max}(T) := \max_{x \in T} \pi(x \,|\, \mathcal{D}), \tag{2}$$

as it is easy to see that safety according to Definition 1 is equivalent to either both $\pi(x^*) \geq 0.5$ and $\pi_{\min}(T) \geq 0.5$ or both $\pi(x^*) \leq 0.5$ and $\pi_{\max}(T) \leq 0.5$.

For a given $T \subseteq \mathbb{R}^d$, we can compute $\pi^L_{\min}(T)$, a lower bound for $\pi_{\min}(T)$, and $\pi^U_{\max}(T)$, an upper bound for $\pi_{\max}(T)$, by optimizing a sum of Gaussian integrals forming a lower (resp. upper) bound function (Proposition 1, Appendix B). Given lower and upper bounds for the latent mean and variance, each of these integrals can be bounded in constant time (Proposition 2, Appendix B). By evaluating the GPC on points in T, we can subsequently compute $\pi^U_{\min}(T)$ and $\pi^L_{\max}(T)$ such that

$$\pi^L_{\min}(T) \leq \pi_{\min}(T) \leq \pi^U_{\min}(T) \quad \text{and} \quad \pi^L_{\max}(T) \leq \pi_{\max}(T) \leq \pi^U_{\max}(T). \tag{3}$$

We iteratively refine these upper and lower bounds in a branch and bound optimization algorithm (Algorithm 1, detailed below) until $\pi^U_{\min}(T) - \pi^L_{\min}(T) \leq \epsilon$ for any chosen $\epsilon > 0$, guaranteeing that both these bounds are $\epsilon-$exact approximations of the true value of $\pi_{\min}(T)$ (analogously for $\pi_{\max}(T)$). This enables us to exactly evaluate safety according to Definition 1 up to error ϵ. The approach is visualized in Figure 1 for the computation of $\pi_{\min}(T)$ for $d = 1$.

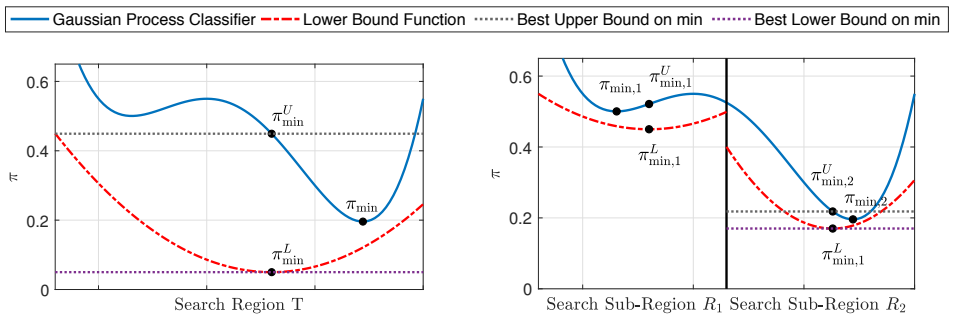

Figure 1: **Left:** Computation of upper and lower bounds on $\pi_{\min}(T)$, the unknown minimum of the classification ranges for the original search region T. **Right:** The search region is repeatedly partitioned into multiple sub-regions according to Algorithm 1 (only first partitioning visualised), reducing the gap between best lower and upper bounds until convergence (up to ϵ) is reached.

3.3 A branch and bound algorithm for convergence

Algorithm 1 sketches the computation of $\pi_{\min}(T)$; it works analogously for $\pi_{\max}(T)$. After initializing $\pi^L_{\min}(T)$, $\pi^U_{\min}(T)$ and the exploration regions stack $\mathbf{R}$, the main optimization loop is entered until convergence (lines 2–9). Among the regions in the current exploration stack, we select the region R with the most promising lower bound (line 3), and refine its lower bounds using Propositions 1 and 2 (lines 4–5) as well as its upper bounds through evaluation of promising candidate points (line 6). If further exploration of R is necessary for convergence (line 7), then the region R is partitioned into two smaller regions R_1 and R_2, which are added to the exploration regions stack and inherit R's bound values (line 8). Finally, the freshly computed bounds local to $R \subseteq T$ are used to update the global bounds for T (line 9). Namely $\pi^L_{\min}(T)$ is updated to the smallest among the $\pi^L_{\min}(R)$ values for $R \in \mathbf{R}$, while $\pi^U_{\min}(T)$ is set to the lowest observed value yet explicitly computed in line 6.

Algorithm 1 Branch and bound for computation of $\pi_{\min}(T)$

Input: Input space subset T; error tolerance $\epsilon > 0$; latent mean/variance functions $\mu(\cdot)$ and $\Sigma(\cdot)$
Output: Lower and upper bounds on $\pi_{\min}(T)$ with $\pi^U_{\min}(T) - \pi^L_{\min}(T) \leq \epsilon$

1: *Initialization:* Stack of regions $\mathbf{R} \leftarrow \{T\}$; $\pi^L_{\min}(T) \leftarrow -\infty$; $\pi^U_{\min}(T) \leftarrow +\infty$
2: **while** $\pi^U_{\min}(T) - \pi^L_{\min}(T) > \epsilon$ **do**
3: Select region $R \in \mathbf{R}$ with current lowest lower bound $\pi^L_{\min}(R)$ and delete it from stack
4: Find bounds $[\mu^L_R, \mu^U_R]$ and $[\Sigma^L_R \Sigma^U_R]$ for latent mean and variance functions over R
5: Compute $\pi^L_{\min}(R)$ from $[\mu^L_R, \mu^U_R]$ and $[\Sigma^L_R \Sigma^U_R]$ using Propositions 1 and 2
6: Find $\pi^U_{\min}(R)$ by evaluating GPC in points yielding boundary values in line 4
7: **if** $\pi^U_{\min}(R) - \pi^L_{\min}(R) > \epsilon$ **then**
8: Split R into two sub-regions R_1, R_2, add them to stack and use $\pi^L_{\min}(R), \pi^U_{\min}(R)$ as bounds for both sub-regions
9: Update current best bounds $\pi^L_{\min}(T), \pi^U_{\min}(T)$ as lowest (highest) $\pi^L_{\min}(R)$ ($\pi^U_{\min}(R)$)
10: **return** $[\pi^L_{\min}(T), \pi^U_{\min}(T)]$

For our approach to work, it is crucial that Algorithm 1 converges, i.e. that the while loop in lines 2-9 terminates. We prove this convergence in finitely many steps for any $\epsilon > 0$ (Theorem 1, Appendix B). A runtime analysis can be found in Appendix D. Lastly, the details on how to find lower and upper bounds of the mean and variance of $q(f(x) = \bar{f}|\mathcal{D})$ for $x \in T$ in line 4 are given in Appendix C.

4 Experimental results

We evaluate our methods on three datasets. The first, 2D-Toy, is generated by shifting two-dimensional standard-normals either along the first dimension (Class 1) or along the second dimension (Class 2). The second is the MNIST38 dataset [11]. The results for the last dataset, SPAM, are included together with further details on each dataset and the training procedures in Appendix A.

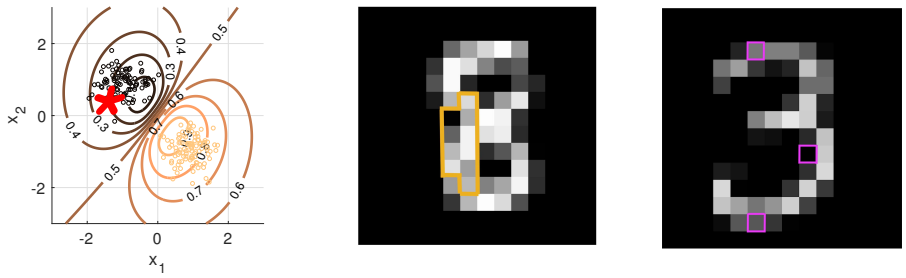

Figure 2: **Left to right**: Contour plot of trained GPC model for 2D-Toy dataset (red star marking selected test point); Sample of 8 from MNIST38 with 10 pixels selected by SIFT (orange); Sample of 3 from MNIST38 with the three pixels that have the shortest lengthscales after GP training (purple).

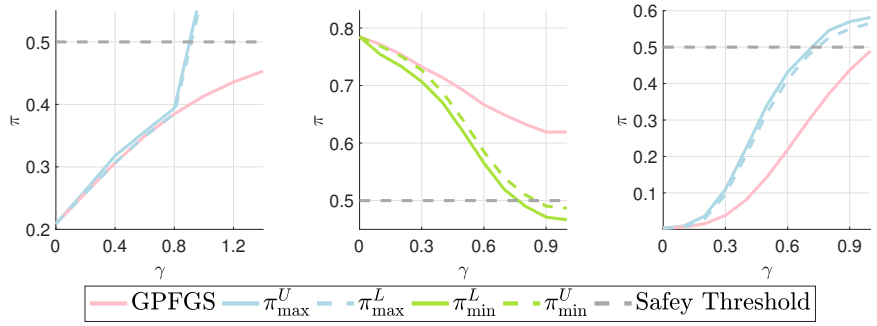

Figure 3: Safety analysis for the test points shown Figure 2 in corresponding order. Shown are the converged upper and lower bounds for $\epsilon = 0.02$ on either $\pi_{\max}(T)$ or $\pi_{\min}(T)$ (solid and dashed blue resp. green lines) and the GPFGS adversarial attack (pink line). The true values of $\pi_{\max}(T)$ or $\pi_{\min}(T)$ are guaranteed to be between the solid and dashed blue (green) lines.

In Figures 2 and 3, we use our method to compute $\pi_{\min}(T)$ and $\pi_{\max}(T)$ for safety verification (Definition 1) and compare it to GPFGS (described in Appendix A), a gradient based heuristic adversarial attack for GPC [5]. To this end, we let $T \subset \mathbb{R}^d$ be a $\gamma-$ball around the chosen test point w.r.t. the L_∞-norm and iteratively increase γ to find its largest value for which $\pi_{\min}(T)$ (or $\pi_{\max}(T)$ respectively) does not cross the decision boundary of 0.5 (dashed line in Figure 3). Note that, unlike analyses purely based on the latent mean, our bounds can be used for decision boundaries different from 0.5 (e.g. in one vs. all classification, robust decision making or cost-sensitive classification).

For MNIST38, we limit the dimensionality of T to 10 and 3 for scalability reasons. This is done using two different feature-relevance detection methods (SIFT [12] and the lengthscales of trained GPC model respectively) to stress that our methods work independently of which method is used.

As can be seen in Figure 3, heuristic adversarial attacks can give a false sense of safety. For the 2D-Toy example (left column), the classification of the GPC model is only safe up to $\gamma \approx 0.9$, at which both $\pi^U_{\max}(T)$ and $\pi^L_{\max}(T)$ cross 0.5. However, the gradient based GPFGS attack is fooled by local non-linearities near the test point and fails to succeed even for $\gamma > 1.2$. The MNIST38 examples (middle and right column) show that such non-linearities can also be present in real-world datasets. Here, safety is in both cases only given up to values of γ of around $0.7 - 0.8$, but the GPFGS attack fails to find a successful example even for $\gamma = 1$, which corresponds to complete pixel flips.

5 Conclusion

We developed results and algorithms for computing, for any compact set of input points, the class probability range of a GP classification model across all points in the set up to any desired precision $\epsilon > 0$. This allows us to analyse robustness and certify safety against adversarial attacks, which we have demonstrated on 3 binary datasets. We plan to extend the experiments to multiclass problems.

References

[1] Yarin Gal and Lewis Smith. Sufficient conditions for idealised models to have no adversarial examples: a theoretical and empirical study with Bayesian neural networks. *arXiv preprint arXiv:1806.00667*, 2018.

[2] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017.

[3] Tencent Keen Security Lab. Experimental security research of Tesla autopilot, 2019.

[4] Arnold Neumaier. Complete search in continuous global optimization and constraint satisfaction. *Acta numerica*, 13:271–369, 2004.

[5] Kathrin Grosse, David Pfaff, Michael T Smith, and Michael Backes. The limitations of model uncertainty in adversarial settings. *arXiv preprint arXiv:1812.02606*, 2018.

[6] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. The MIT Press, 2004.

[7] Christopher KI Williams and David Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

[8] Thomas P Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001.

[9] James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. *JMLR*, 2015.

[10] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[11] Yann LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[12] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[13] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

[16] V Balakrishnan, S Boyd, and S Balemi. Branch and bound algorithm for computing the minimum stability degree of parameter-dependent linear systems. *International Journal of Robust and Nonlinear Control*, 1(4):295–317, 1991.

[17] Luca Cardelli, Marta Kwiatkowska, Luca Laurenti, and Andrea Patane. Robustness guarantees for Bayesian inference with Gaussian processes. *arXiv preprint arXiv:1809.06452*, 2018.

[18] J Ben Rosen and Panos M Pardalos. Global minimization of large-scale constrained concave quadratic problems by separable programming. *Mathematical Programming*, 34(2):163–174, 1986.

[19] Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust optimization with Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 5760–5770, 2018.

[20] Hyun-Chul Kim and Zoubin Ghahramani. Outlier robust Gaussian process classification. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 896–905. Springer, 2008.

[21] Daniel Hernández-Lobato, José M Hernández-Lobato, and Pierre Dupont. Robust multi-class Gaussian process classification. In *Advances in neural information processing systems*, pages 280–288, 2011.

[22] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[23] Luca Cardelli, Marta Kwiatkowska, Luca Laurenti, Nicola Paoletti, Andrea Patane, and Matthew Wicker. Statistical guarantees for the robustness of Bayesian neural networks. *arXiv preprint arXiv:1903.01980*, 2019.

[24] Matthias Seeger. Pac-Bayesian generalisation error bounds for Gaussian process classification. *Journal of machine learning research*, 3(Oct):233–269, 2002.

[25] David McAllester. A PAC-Bayesian tutorial with a dropout bound. *arXiv preprint arXiv:1307.2118*, 2013.

APPENDIX: Adversarial Robustness Guarantees for Classification with Gaussian Processes

In the first section of the Appendix (Appendix A) we give insights into the experimental results on the SPAM dataset as well as more details on the experimental settings in Section 4, such as on the datasets and training procedures. The subsequent 3 sections of the Appendix contain the theoretical and algorithmic results that our method is built upon. In Appendix B we state and prove the results mentioned in Sections 3.2 and 3.3. In particular, we detail the results to compute upper and lower bounds on $\pi_{\max}(T)$ and $\pi_{\min}(T)$ and prove the convergence of Algorithm 1. In Appendix C we detail how the lower and upper bounds to the latent mean and variance that are required for Proposition 2 are computed. In Appendix D we discuss the resulting computational complexity of our approach and provide an empirical runtime analysis on 30 test points of MNIST38. Finally, we present the extension of our theoretical framework to the multiclass classification GP case in Appendix E before finishing with a section on related work in Appendix F.

A Further experiments and details on experimental settings

A.1 Experiments on SPAM dataset

We also conducted experiments on the SPAM dataset from the UCI database [13], but found the results less interesting as SPAM appears to be a dataset that can be separated linearly (see Figure 4 left).

An illustration of the behaviour of adversarial attacks on SPAM can be found in Figure 4 (right). For the SPAM dataset, the GPC model is locally linear near the analysed test point (see contour plot in Figure 4) and thus by following the gradient, GPFGS gets close to the actual worst-case sample in the explored region, and also to the obtained bound. It is in this case able to find a successful adversarial example efficiently. However, such a strongly linear behaviour is not guaranteed to be present in other real applications, as has been demonstrated in Section 4 and in general, adversarial attacks should not be trusted as sources for adversarial robustness guarantees.

A.2 Details on datasets

Our synthetic two-dimensional dataset contains 1,200 points, of which 50 % belong to Class 1 and 50 % belong to Class 2. The points were generated by shifting draws from a two-dimensional standard-normal random variable by 5, either along the first dimension (Class 1) or along the second dimension (Class 2). Subsequently, we normalise the data by subtracting its mean and dividing by its standard deviation.

MNIST38 is a subset of the original MNIST dataset [11] only consisting of 3s and 8s. It contains 8,403 samples of images of handwritten digits, of which roughly 50 % are 3s and 50 % are 8s. Each sample consists of a 28×28 pixel image in gray scale (integer values between 0 and 255) which following convention, we normalise by dividing by 255. For better scalability we then downsample to 14×14 pixels.

SPAM is a binary dataset that contains 4,601 samples, of which 60% are benign. Each sample consists of 54 real-valued and three integer-valued features. However, identical or better prediction accuracies can be achieved with models involving only 11 of those 57 variables, among them e.g. the frequency of the word 'free' in the email, the share of $ signs in its body, or the total number of capital letters, which is why we only use these 11 selected variables. We normalise the data by subtracting its mean and dividing by its standard deviation.

A.3 Training procedures

For the binary experiments, we use 1,000 randomly selected points as a training set and 200 randomly selected points as a test set. Training is done with the Laplace approximation [6] and the probit link function as implemented in the GPML package for Matlab. The number of epochs performed during hyper-parameter optimization was restricted to 20 for the synthetic dataset and MNIST38, and 40 for SPAM, yielding test set accuracies of 100%, 94%, and 93% on the synthetic two-dimensional dataset, MNIST38 and SPAM respectively.

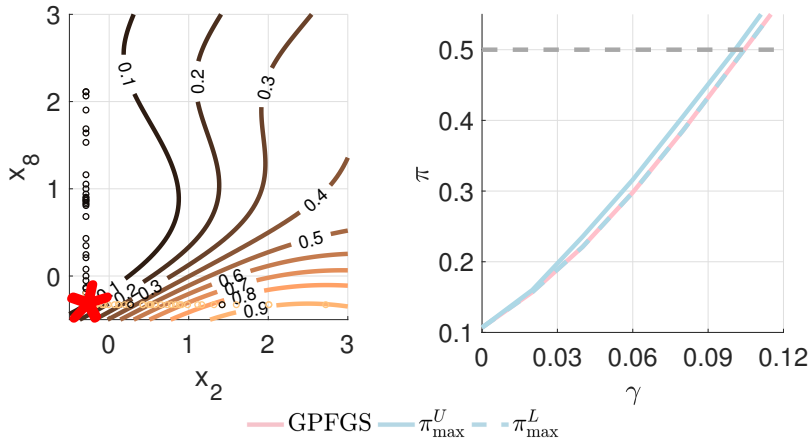

Figure 4: **Left**: Projected contour plot for 2 most influential inputs for SPAM dataset (dimensions 2 and 8) as selected by L_1-penalised logistic regression **Right**: Safety analysis using the refined upper (solid blue line) and lower (dashed blue line) bound on $\pi_{\max}(T)$ for test point highlighted by red star in contour plot on the left with $\epsilon = 0.02$. Comparison with heuristic adversarial attack method (pink line).

A.4 GPFGS attacks

GPFGS attacks have been developed in [5] as the GP analogue of the well-known fast gradient sign method (FGSM) attack for neural networks [14]. Given a perturbation budget γ, FGSM attacks a model at test point x^* by perturbing each dimension by γ in the direction of its gradient at x^*. Since for GPs, the integral in Eqn 1 makes calculating the gradient directly impossible, [5] propose to base the attack on the gradient of the latent mean function $\mu(\cdot)$ instead. The GPFGS attack is thus defined by

$$x_{adv} = x^* + \gamma \times sign(\nabla \mu(x^*)) \tag{4}$$

The code for the GPFGS attacks was implemented by us in Matlab according to the original Python code provided by the authors.

B Theoretical results for binary classification bounds

Proposition 1. *Let $\mathcal{S} = \{S_i, i \in \{1, ...N\}\}$ be a partition of $\mathbb{R}$ in a finite set of intervals and $\sigma(\cdot)$ a non-decreasing, continuous function link function. Call $a_i = \inf_{\bar{f} \in S_i} \bar{f}$ and $b_i = \sup_{\bar{f} \in S_i} \bar{f}$. Then, for any $x \in T$ it holds that:*

$$\pi_{\min}(T) = \min_{x \in T} \pi(x) \geq \sum_{i=1}^{N} \sigma(a_i) \min_{x \in T} \int_{a_i}^{b_i} \mathcal{N}(\bar{f} | \mu(x), \Sigma(x)) d\bar{f} \tag{5}$$

$$\pi_{\max}(T) = \max_{x \in T} \pi(x) \leq \sum_{i=1}^{N} \sigma(b_i) \max_{x \in T} \int_{a_i}^{b_i} \mathcal{N}(\bar{f} | \mu(x), \Sigma(x)) d\bar{f}, \tag{6}$$

where $\mu(x)$ and $\Sigma(x)$ are mean and variance of the predictive posterior $q(f(x) = \bar{f} | \mathcal{D})$.

Proof. We detail the proof for $\min_{x\in T}\pi(x)$. The $\max$ case follows similarly.

$$
\begin{aligned}
&\min_{x\in T}\pi(x)\\
&\quad\text{(By definition)}\\
=&\min_{x\in T}\int_{-\infty}^{+\infty}\sigma(\bar{f})q(f(x)=\bar{f}|\mathcal{D})d\bar{f}\\
&\quad\text{(By additivity of integrals)}\\
=&\min_{x\in T}\sum_{i=1}^{N}\int_{a_i}^{b_i}\sigma(\bar{f})q(f(x)=\bar{f}|\mathcal{D})d\bar{f}\\
&\quad\text{(By monotonicity of }\sigma\text{ and non-negativity of }q)\\
\geq&\min_{x\in T}\sum_{i=1}^{N}\int_{a_i}^{b_i}\sigma(a_i)q(f(x)=\bar{f}|\mathcal{D})d\bar{f}\\
&\quad\text{(By definition of minimum and of }q)\\
\geq&\sum_{i=1}^{N}\sigma(a_i)\min_{x\in T}\int_{a_i}^{b_i}\mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}
\end{aligned}
$$

□

Proposition 1 guarantees that Eqn (2) can be over-approximated by solving N optimization problems. Each of these problems seeks to find the mean and variance that maximize or minimize the integral of a Gaussian distribution over T. Proposition 2 gives constant time bounds to each of them, using the following notations for lower and upper bounds of latent mean and variance:

$$\mu_T^L\leq\min_{x\in T}\mu(x)\quad\mu_T^U\geq\max_{x\in T}\mu(x)\quad\Sigma_T^L\leq\min_{x\in T}\Sigma(x)\quad\Sigma_T^U\geq\max_{x\in T}\Sigma(x).\tag{7}$$

Proposition 2. *Let $T\subseteq\mathbb{R}^d$. Let $\mu^m=\frac{a+b}{2}$ and $\Sigma^m(\mu)=\frac{(\mu-a)^2-(\mu-b)^2}{2\log\frac{\mu-a}{\mu-b}}$, then it holds that:*

$$\max_{x\in T}\int_a^b\mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}\leq\int_a^b\mathcal{N}(\bar{f}|\overline{\mu},\overline{\Sigma})d\bar{f}=\frac{1}{2}\left(\operatorname{erf}\left(\frac{\overline{\mu}-a}{\sqrt{2\overline{\Sigma}}}\right)-\operatorname{erf}\left(\frac{\overline{\mu}-b}{\sqrt{2\overline{\Sigma}}}\right)\right)\tag{8}$$

$$\min_{x\in T}\int_a^b\mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}\geq\int_a^b\mathcal{N}(\bar{f}|\underline{\mu},\underline{\Sigma})d\bar{f}=\frac{1}{2}\left(\operatorname{erf}\left(\frac{\underline{\mu}-a}{\sqrt{2\underline{\Sigma}}}\right)-\operatorname{erf}\left(\frac{\underline{\mu}-b}{\sqrt{2\underline{\Sigma}}}\right)\right)\tag{9}$$

where: $\overline{\mu}=\arg\min_{\mu\in[\mu_T^L,\mu_T^U]}|\mu^m-\mu|$ and $\overline{\Sigma}$ is equal to Σ_T^L if $\overline{\mu}\in[a,b]$, otherwise $\overline{\Sigma}=\arg\min_{\Sigma\in[\Sigma_T^L,\Sigma_T^U]}|\Sigma^m(\overline{\mu})-\Sigma|$. Analogously, for the infimum we have: $\underline{\mu}=\arg\max_{\mu\in[\mu_T^L,\mu_T^U]}|\mu^m-\mu|$ and $\underline{\Sigma}=\arg\min_{\Sigma\in\{\Sigma_T^L,\Sigma_T^U\}}[\operatorname{erf}(b|\underline{\mu},\Sigma)-\operatorname{erf}(a|\underline{\mu},\Sigma)]$.

Proof. We provide the proof for the $\inf$ case, similar arguments hold for the $\sup$. By definition of μ_T^L, μ_T^U, Σ_T^L, Σ_T^U we have that:

$$
\begin{aligned}
\min_{x\in T}\int_a^b\mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}&\geq\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\\Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\int_a^b\mathcal{N}(\bar{f}|\mu,\Sigma)d\bar{f}\\
&=\frac{1}{2}\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\\Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\left(\operatorname{erf}\left(\frac{\mu-a}{\sqrt{2\Sigma}}\right)-\operatorname{erf}\left(\frac{\mu-b}{\sqrt{2\Sigma}}\right)\right)\\
&:=\frac{1}{2}\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\\Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\Phi(\mu,\Sigma).
\end{aligned}
$$

By looking at the partial derivatives we have that:

$$\frac{\partial \Phi(\mu,\Sigma)}{\partial \mu} = \frac{\sqrt{2}}{\sqrt{\pi\Sigma}}\left(e^{-\frac{(\mu-b)^2}{2\Sigma}} - e^{-\frac{(\mu-a)^2}{2\Sigma}}\right) \geq 0 \Leftrightarrow \mu \leq \frac{a+b}{2} =: \mu^c$$

and that if $\mu \notin [a,b]$:

$$\frac{\partial \Phi(\mu,\Sigma)}{\partial \Sigma} = \frac{1}{\sqrt{2\pi\Sigma^3}}\left((\mu-b_i)e^{-\frac{(\mu-b_i)^2}{2\Sigma^2}} - (\mu-a_i)e^{-\frac{(\mu-a_i)^2}{2\Sigma^2}}\right) \geq 0$$
$$\Leftrightarrow \Sigma \leq \frac{(\mu-a)^2-(\mu-b)^2}{2\log\frac{\mu-a}{\mu-b}} := \Sigma^c(\mu)$$

otherwise the last inequality has no solutions. As such μ^c and Σ^c will correspond to global maximum wrt to μ and Σ respectively. As Φ is symmetric wrt μ^c we have that the minimum value wrt to μ is always obtained for the point furthest away from μ^c, that is: $\underline{\mu} = \arg\max_{\mu\in[\mu_T^L,\mu_T^U]} |\mu^c - \mu|$. The minimum value wrt to Σ will hence be either for Σ_T^L or Σ_T^U, that is $\underline{\Sigma} = \arg\min_{\Sigma\in\{\Sigma_T^L,\Sigma_T^U\}} \Phi(\underline{\mu},\Sigma)$. □

Classification with probit link function. For the case that the link function σ is taken to be the probit function, that is, $\sigma(\bar{f}) = \Phi(\lambda\bar{f})$ is the cdf of the univariate standard Gaussian distribution scaled by $\lambda > 0$, it holds that $\pi(x \,|\, \mathcal{D}) = \Phi\left(\frac{\mu(x)}{\sqrt{\lambda^{-2}+\Sigma(x)}}\right)$, where $\mu(x)$ and $\Sigma(x)$ are the mean and variance of $q(f(x) = \bar{f}|\mathcal{D})$ [15]. We can use this result to derive analytic upper and lower bounds for Eqn (2) without the need to apply Proposition 1, by relying on upper and lower bounds for the latent mean and variance functions.

Lemma 1. *Let $T \subseteq \mathbb{R}^d$. Then, we have that $\Phi\left(\frac{\mu_T^L}{\sqrt{\lambda^{-2}+\underline{\Sigma}}}\right) \leq \pi_{\min}(T)$ and $\pi_{\max}(T) \leq \Phi\left(\frac{\mu_T^U}{\sqrt{\lambda^{-2}+\overline{\Sigma}}}\right)$ with $\underline{\Sigma} = \Sigma_T^U$ if $\mu_T^L \geq 0$ and Σ_T^L otherwise, while $\overline{\Sigma} = \Sigma_T^L$ if $\mu_T^U \geq 0$ and Σ_T^U otherwise.*

Proof of convergence of Algorithm 1. Finally, given any a-priori specified error threshold $\epsilon > 0$, the following theorem ensures that there exists a latent space discretization such that the bounding error vanishes, thus guaranteeing convergence of the method in finite time due to the properties of branch and bound algorithms [16].

Theorem 1. *Assume $\mu : \mathbb{R}^d \to \mathbb{R}$ and $\Sigma : \mathbb{R}^d \to \mathbb{R}$ are Lipschitz continuous in $T \subseteq \mathbb{R}^d$. Then, for any $\epsilon > 0$, there exists a partition of the latent space $\mathcal{S}$ and $r > 0$ such that, for every $R \subseteq T$ of side length of less than r, it holds that $|\pi_{\min}^U(R) - \pi_{\min}^L(R)| \leq \epsilon$ and $|\pi_{\max}^U(R) - \pi_{\max}^L(R)| \leq \epsilon$.*

Proof. We consider the $\min$ case. The $\max$ case follows similarly.

In order to show the convergence of the branch and bound, we need to show that for any test point x there exists $r > 0$ and a partition of the latent space $\mathcal{S} = \{S_i, i = \{1, ..., N\}\}$ such that for the interval $I = [x - rI, x + rI]$ we have that for any $\bar{x} \in I$

$$\left|\pi(\bar{x}) - \sum_{i=1}^{N} \sigma(a_i) \min_{x\in I} \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}\right| \leq \epsilon.$$

In order to do that, we first observe that by the Lipschitz continuity of mean and variance we have that for $x_1, x_2 \in I$, it holds that

$$|\mu(x_1) - \mu(x_2)| \leq K^\mu r$$
$$|\Sigma(x_1) - \Sigma(x_2)| \leq K^\Sigma r,$$

for certain $K^\mu, K^\Sigma > 0$. Now, for $S_i \in \mathcal{S}$, consider x^i such that $\int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x^i),\Sigma(x^i))d\bar{f} = \min_{x\in I} \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}$. Further, due to the monotonicity and continuity of σ, we consider

a uniform discretization of the y-axis for σ in N intervals. That is, for all $S_i \in \mathcal{S}$, we have that $\sigma(b_i) = \sigma(a_i) + \frac{1}{N}$. At this point, for any $\bar{x} \in I$ the following calculations follow

$$\left| \pi(\bar{x}) - \sum_{i=1}^{N} \sigma(a_i) \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f} \right| \tag{10}$$

(By Definition)

$$= \left| \int \sigma(\bar{f}) \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} - \sum_{i=1}^{N} \sigma(a_i) \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f} \right| \tag{11}$$

(By additivity of integral and re-ordering terms)

$$= \left| \sum_{i=1}^{N} \left(\int_{a_i}^{b_i} \sigma(\bar{f}) \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} \; - \; \int_{a_i}^{b_i} \sigma(a_i) \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f} \right) \right| \tag{12}$$

(As for any S_i, $\int_{a_i}^{b_i} \sigma(\bar{f}) \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} \geq \int_{a_i}^{b_i} \sigma(a_i) \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f}$)

$$\leq \left| \sum_{i=1}^{N} \left(\int_{a_i}^{b_i} (\sigma(a_i) + \frac{1}{N}) \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) \; - \; \sigma(a_i) \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f} \right) \right| \tag{13}$$

(By Triangle Inequality)

$$\leq \left| \sum_{i=1}^{N} \int_{a_i}^{b_i} \frac{1}{N} \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} \right|$$
$$+ \left| \sum_{i=1}^{N} \left(\int_{a_i}^{b_i} (\sigma(a_i)) \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) \; - \; \sigma(a_i) \mathcal{N}(\bar{f}\mu(x^i), \Sigma(x^i)) d\bar{f} \right) \right| \tag{14}$$

(By Re-ordering terms and Triangle Inequality)

$$\leq \left| \frac{1}{N} \int \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} \right|$$
$$+ \sum_{i=1}^{N} \sigma(a_i) \left| \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) \; - \; \mathcal{N}(\bar{f}\mu(x^i), \Sigma(x^i)) d\bar{f} \right| \tag{15}$$

(By basic Inequalities of integrals and the fact that $\sigma(f) \in [0, 1]$)

$$\leq \frac{1}{N} + \sum_{i=1}^{N} \left| \int_{a_i}^{b_i} (\mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) \; - \; \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i))) d\bar{f} \right| \tag{16}$$

Now, as $|\mu(\bar{x}), -\mu(x^i), | \leq K^{\mu} r$ and $|\Sigma^2(\bar{x}) - \Sigma^2(x^i)| \leq K^{\Sigma} r$, we have that as $r \to 0$ both mean and variance converge to the same value. Hence, this implies that for each $S_i \in S$

$$\lim_{r \to 0} \left(\int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) d\bar{f} \; - \; \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)) d\bar{f} \right) = 0.$$

As a consequence, for any $\epsilon > 0$, we can choose $N = \lceil \frac{2}{\epsilon} \rceil$ and then select r such that the second term in Eqn 16 is bounded by $\frac{\epsilon}{2}$.

□

C Bounds on latent mean and variance

Note that Proposition 2 relies on Eqn (7), that is, it requires lower and upper bounds of the mean and variance of $q(f(x) = \bar{f}|\mathcal{D})$ for $x \in T$. We obtain these by applying the framework presented in [17] for computation of μ_T^L, μ_T^U and Σ_T^U. Briefly, assuming continuity and differentiability of the kernel function defining the prior covariance, it is possible to find linear upper and lower bounds on the covariance vector, which can be propagated through the inference formula for $q(f(x) = \bar{f}|\mathcal{D})$.

However, for Σ_T^L, simply applying the same framework often results in excessively loose bounds that defer convergence. Hence we solve the concave quadratic problem that arises when computing Σ_T^L by adapting methods introduced in [18]. The details are given in Appendix C.

Calculation of lower variance bound. Let $\mathbf{r}(x) = [r_1(x), \dots, r_M(x)]$ be the vector of covariance between a test point and the training set $\mathcal{D}$ with $|\mathcal{D}| = M$, and let R be the inverse covariance matrix computed in the training set, and Σ_p be the (input independent) self kernel value. By explicitly using the variance inference formula, we are interested in finding a lower bound for: $\min_{x \in T} \left(\Sigma_p - \mathbf{r}(x)^T R \mathbf{r}(x)\right) = \Sigma_p + \min_{x \in T} \left(-\mathbf{r}(x)^T R \mathbf{r}(x)\right)$. We proceed by introducing the M auxiliary variables $r_i = \mathbf{r}_i(x)$, yielding a quadratic objective function on the auxiliary variable vector $\mathbf{r} = [r_1, \dots, r_M]$, that is $-\mathbf{r}^T R \mathbf{r}$. Analogously to what is done in [17] we can compute two matrices A_r, A_x and a vector b such that $\mathbf{r} = \mathbf{r}(x)$ implies $A_r \mathbf{r} + A_x x \leq b$, hence obtaining the quadratic program:

$$\begin{aligned} &\min -\mathbf{r}^T R \mathbf{r} \\ &\text{Subject to:} \quad A_r \mathbf{r} + A_x x \leq b \\ &\qquad\qquad r_i^L \leq r_i \leq r_i^U \qquad i = 1, \dots, M \\ &\qquad\qquad x_i^L \leq x_i \leq x_i^U \qquad i = 1, \dots, d \end{aligned} \tag{17}$$

whose solution provides a lower bound (and hence a safe approximation) to the original problem $\min_{x \in T} \left(-\mathbf{r}(x)^T R \mathbf{r}(x)\right)$. Unfortunately, as R is positive definite, we have that $-R$ is negative definite; hence the problem posed is a concave quadratic program for which a number of local optima may exist. As we are instead dealing with worst-case scenario analyses, we are actually interested in computing the global minimum. This however is an NP-hard problem [18] whose exact solution would make a branch and bound algorithm based on it impractical. Following the methods discussed in [18], we instead proceed to compute a safe lower bound to that. The main observation is that, being R symmetric positive definite, there exist a matrix of eigenvectors $U = [\mathbf{u}_1, \dots, \mathbf{u}_M]$ and a diagonal matrix of the associated eigenvalues λ_i for $i = 1, \dots, M$, Λ such that $R = U \Lambda U^T$. We hence define $\hat{r}_i = \mathbf{u}_1^T r_i$ for $i = 1, \dots, M$ to be the rotated variables and compute their ranges $[\hat{r}_i^L, \hat{r}_i^U]$ by solution of the following $2M$ linear programming problems:

$$\begin{aligned} &\min / \max \quad \mathbf{u}_i^T r_i \\ &\text{Subject to:} \quad A_r \mathbf{r} + A_x x \leq b \\ &\qquad\qquad r_j^L \leq r_i \leq r_j^U \qquad j = 1, \dots, M \\ &\qquad\qquad x_j^L \leq x_i \leq x_j^U \qquad j = 1, \dots, d. \end{aligned}$$

Implementing the change of variables into Problem 17 we obtain:

$$\begin{aligned} &\min -\hat{\mathbf{r}}^T \Lambda \hat{\mathbf{r}} \\ &\text{Subject to:} \quad \hat{A}_{\hat{r}} \hat{\mathbf{r}} + A_x x \leq b \\ &\qquad\qquad \hat{r}_i^L \leq \hat{r}_i \leq \hat{r}_i^U \qquad i = 1, \dots, M \\ &\qquad\qquad x_i^L \leq x_i \leq x_i^U \qquad i = 1, \dots, d \end{aligned}$$

where we set $\hat{A} = AU$. We then notice that $\hat{\mathbf{r}}^T \Lambda \hat{\mathbf{r}} = \sum_i \lambda_i \hat{r}_i^2$. By using the methods developed in [17] it is straightforward to find coefficients of a linear under approximations α_i and β_i such that: $\alpha_i + \beta_i \hat{r}_i \leq -\lambda_i \hat{r}_i^2$ for $i = 1, \dots, M$. Defining $\beta = [\beta_1, \dots, \beta_M]$, and $\hat{\alpha} = \sum_{i=1}^M \alpha_i$ we then have that the solution to the following linear programming problem provides a valid lower bound to Problem 17 and can be hence used to compute a lower bound to the latent variance:

$$\begin{aligned} &\min \left(\hat{\alpha} + \beta^T \hat{\mathbf{r}}\right) \\ &\text{Subject to:} \quad \hat{A}_{\hat{r}} \hat{\mathbf{r}} + A_x x \leq b \\ &\qquad\qquad \hat{r}_i^L \leq \hat{r}_i \leq \hat{r}_i^U \qquad i = 1, \dots, M \\ &\qquad\qquad x_i^L \leq x_i \leq x_i^U \qquad i = 1, \dots, d. \end{aligned}$$

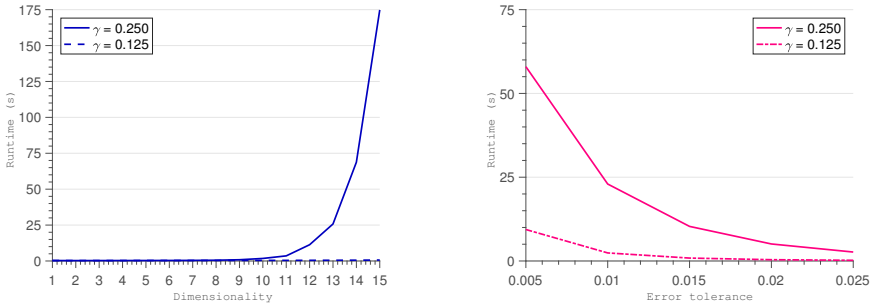

Figure 5: Average runtimes for calculation of $\pi_{\max}(T)$ among the 30 test points. **Left**: Average runtimes for increasing number of dimensions at $\epsilon = 0.025$. **Right**: Average runtimes for different values of ϵ with number of dimensions $d = 10$.

D Computational complexity and runtime analysis of Algorithm 1

D.1 Computational complexity.

Proposition 2 implies that the bounds in Proposition 1 can be obtained in $\mathcal{O}(N)$, with N being the number of intervals the real line is being partitioned into. For the particular case of the probit link function, this can be done in constant time by applying Lemma 1. Computation of μ_T^L and μ_T^U is performed in $\mathcal{O}(|\mathcal{D}|)$ while obtaining Σ_T^U involves the solution of a convex quadratic problem in $m+|\mathcal{D}|$ variables [17]. Solving for Σ_T^L requires the solution of $2|\mathcal{D}|+1$ linear programming problems in $m + |\mathcal{D}|$ dimensions. Refining through branch and bound has a worst-case cost exponential in the number of non-trivial dimensions of T.

D.2 Runtime analysis.

To shed some light on what finite time convergence looks like in practice, we include results of a runtime experiment conducted on the more complex data set under consideration, MNIST38. For 30 test points and a $\gamma-$ball T of dimensionality d, we calculated $\pi_{\max}(T)$ up to a pre-specified error tolerance ϵ. We use $\gamma = 0.125$ and $\gamma = 0.25$, corresponding to up to 50% of the normalised input domain. All runtimes shown were obtained on a MacBook Pro with a 2.5 Ghz Intel Core i7 processor and 16GB RAM running on macOS Mojave 10.14.6.

Runtime depending on dimension of compact subset. First, we analysed the effect of increasing d, by fixing $\epsilon = 0.025$ and increasing the number of pixels selected by SIFT to define T from 1 to 15. The results are shown in terms of average runtime in Figure 5 on the left. For $\gamma = 0.25$, we can observe the exponential behaviour of the computational complexity in terms of number of dimensions, as the runtime quickly grows from below 5 seconds to 175 seconds beyond 10 dimensions. However, for $\gamma = 0.125$ the exponential curve seems to be shifted further to the right, as still for 15 dimensions Algorithm 1 converges in only a few seconds. Given that for $\gamma = 0.125$, T spans up to 25% of the input domain (on the selected pixels), we consider this quite fast.

Runtime depending on error tolerance Second, we analysed the effect of the error tolerance ϵ, by calculating the bounds for each $\epsilon \in \{0.005, 0.01, 0.015, 0.02, 0.025\}$ with the number of pixels selected by SIFT (i.e. d) fixed to 10. The results are shown in Figure 5 on the right. The behaviour seems to be roughly inversely exponential this time with lower error tollerance ϵ naturally demanding higher runtimes. In practice, one would seldom expect to require precision of $\epsilon < 0.01$ though, at which point Algorithm 1 still converges in under 25 seconds even for $\gamma = 0.25$.

E Extension to multiclass classification

In Proposition 3 and 4 we show that, similarly to the binary case, Eqn (1) can be written as a summation of multi-dimensional Gaussian integrals. Building on this result, Algorithm 1 can also be applied to the multiclass setting.

Proposition 3. *Let $\mathcal{S} = \{S_i, i \in \{1, ...N\}\}$ be a fine partition of $\mathbb{R}^C$. Then, for $c \in \{1, ..., N\}$:*

$$\pi_{\min}(T) \geq \sum_{i=1}^{N} \min_{x \in S_i} \sigma^c(x) \min_{x \in T} \int_{S_i} \mathcal{N}(\bar{f}|\mu(x), \Sigma(x)) d\bar{f} \tag{18}$$

$$\pi_{\max}(T) \leq \sum_{i=1}^{N} \max_{x \in S_i} \sigma^c(x) \max_{x \in T} \int_{S_i} \mathcal{N}(\bar{f}|\mu(x), \Sigma(x)) d\bar{f} \tag{19}$$

Proof. We detail the proof for $\min_{x \in T} \pi^c(x)$. The $\max$ case follows similarly.

$$\begin{aligned}
&\min_{x \in T} \pi^c(x) \\
&\quad \text{(By definition)} \\
&= \min_{x \in T} \int \sigma^c(\bar{f}) q(f(x) = \bar{f}|\mathcal{D}) d\bar{f} \\
&\quad \text{(By additivity of integral)} \\
&= \min_{x \in T} \sum_{i=1}^{N} \int_{S_i} \sigma^c(\bar{f}) q(f(x) = \bar{f}|\mathcal{D}) d\bar{f} \\
&\quad \text{(Because } q \text{ is non-negative)} \\
&\geq \min_{x \in T} \sum_{i=1}^{N} \int_{S_i} \min_{y \in S_i} \sigma^c(y) q(f(x) = \bar{f}|\mathcal{D}) d\bar{f} \\
&\quad \text{(By definition of minimum)} \\
&\geq \sum_{i=1}^{N} \min_{y \in S_i} \sigma^c(y) \min_{x \in T} \int_{S_i} q(f(x) = \bar{f}|\mathcal{D}) d\bar{f} \\
&\quad \text{(By Definition of } q\text{)} \\
&= \sum_{i=1}^{N} \min_{y \in S_i} \sigma^c(y) \min_{x \in T} \int_{S_i} \mathcal{N}(\bar{f}|\mu(x), \Sigma(x)) d\bar{f}
\end{aligned}$$

□

Proposition 3 guarantees that, for all $x \in T$, $\pi^c(x)$ can be upper and lower bounded by solving $2N$ optimization problems. As shown below in Lemma 2, under the assumption that σ is the softmax function and that S_i is an axis-parallel hyper-rectangle, $\sup_{x \in S_i} \sigma^c(x)$ and $\inf_{x \in S_i} \sigma^c(x)$ can be computed by simply evaluating the vertices of S_i. Further, in Proposition 4, we show that upper and lower bounds for the integral of a multi-dimensional Gaussian distribution, such as those appearing in Eqns (18) and (19), can be obtained by optimizing uni-dimensional integrals over both the input and latent space. Each of these integrals can be optimized by employing Proposition 2.

In what follows, we call $\mu_{i:j}(x)$ the subvector of $\mu(x)$ containing only the components from i to j and similarly we define $\Sigma_{i:k,j:l}(x)$ the submatrix of $\Sigma(x)$ containing rows from i to k and columns from j to l. Then, the following proposition follows.

Proposition 4. *Let $S = \prod_{i=1}^{C} [k_i^1, k_i^2]$ be an axis-parallel hyper-rectangle. For $i \in \{1, \ldots, C-1\}$ and $f \in \mathbb{R}^{C-1-i}$, define $\mathcal{I} := i+1 : C$ and:*

$$\mu_i^f(x) = \mu_i(x) - \Sigma_{i,\mathcal{I}}(x) \Sigma_{\mathcal{I},\mathcal{I}}^{-1} (f - \mu_{\mathcal{I}}(x)) \quad \Sigma_i^f(x) = \Sigma_{i,i}(x) - \Sigma_{i,\mathcal{I}}(x) \Sigma_{\mathcal{I},\mathcal{I}}^{-1} \Sigma_{i,\mathcal{I}}^T(x).$$

Let $S^{i+1} = \prod_{j=i+1}^{C}[k_j^1, k_j^2]$*, then we have that:*

$$\max_{x\in T}\int_S \mathcal{N}(z|\mu(x),\Sigma(x)) \leq \max_{x\in T}\int_{k_C^1}^{k_C^2}\mathcal{N}(z|\mu_C(x),\Sigma_{C,C}(x))dz\prod_{i=1}^{C-1}\max_{\substack{x\in T\\ f\in S^{i+1}}}\int_{k_i^1}^{k_i^2}\mathcal{N}(z|\mu_i^f(x),\Sigma_i^f(x))dz$$

$$\min_{x\in T}\int_S \mathcal{N}(z|\mu(x),\Sigma(x)) \geq \min_{x\in T}\int_{k_C^1}^{k_C^2}\mathcal{N}(z|\mu_C(x),\Sigma_{C,C}(x))dz\prod_{i=1}^{C-1}\min_{\substack{x\in T\\ f\in S^{i+1}}}\int_{k_i^1}^{k_i^2}\mathcal{N}(z|\mu_i^f(x),\Sigma_i^f(x))dz$$

Proposition 5. *We consider the maximum case. The minimum follows similarly. Let* $\mathbf{y}(x)$ *be a normal random variable with mean* $\mu(x)$ *and covariance matrix* $\Sigma(x)$*. Then, we have*

$$\begin{aligned}
&\max_{x\in T}\int_S \mathcal{N}(z|\mu(x),\Sigma(x))dz\\
=&\max_{x\in T} P(\mathbf{y}(x)\in S)\\
=&\max_{x\in T} P(\wedge_{i=1}^{C} k_i^1 \leq \mathbf{y}_i(x) \leq k_i^2)\\
=&\max_{x\in T}\prod_{i=1}^{C} P(k_i^1 \leq \mathbf{y}_i(x) \leq k_i^2 |\wedge_{j=i+1}^{C} k_j^1 \leq \mathbf{y}_j(x) \leq k_j^2)\\
&\quad \textit{(By Lemma 3)}\\
\leq&\max_{x\in T}\prod_{i=1}^{C}\max_{f\in S^{i+1}} P(k_i^1 \leq \mathbf{y}_i(x) \leq k_{i,2} |\wedge_{j=i+1}^{C} \mathbf{y}_j(x) = f_{j-i})\\
\leq&\prod_{i=1}^{C}\max_{x\in T, f\in S^{i+1}} P(k_i^1 \leq \mathbf{y}_i(x) \leq k_i^2 |\wedge_{j=i+1}^{C} \mathbf{y}_j(x) = f_{j-i})
\end{aligned}$$

Notice that for each $i \in \{1,...,C\}$*,* $P(k_i^1 \leq \mathbf{y}_i(x) \leq k_i^2 |\wedge_{j=i+1}^{C} \mathbf{y}_j(x) = f_{j-i})$ *is the integral of a uni-dimensional Gaussian random variable, as a Gaussian random variable conditioned to a jointly Gaussian random variable is still Gaussian.*

Similarly to the binary case, our method is guaranteed to compute a safe over-approximation of the class probability ranges with a quantifiable error also for the multiclass case. However, due to the discretization of the latent space, the resulting approach has a computational complexity that is exponential in the number of classes.

Auxiliary Lemmata. Proposition 3 in implies that if we can compute infimum and supremum of the softmax over a set of the latent space (shown in Lemma 2) and the mean and covariance matrix that maximize a Gaussian integral (shown in Proposition 4), then upper and lower bounds on $\pi_{\min}(T)$ and $\pi_{\max}(T)$ can be derived.

Lemma 2. *Let* $S \subset \mathbb{R}^{|C|}$ *be an axis-parallel hyper-rectangle. Call* $f^{max} = \arg\max_{f\in S}\sigma^c(f)$ *and* $f^{min} = \arg\min_{f\in S}\sigma^c(f)$*. Assume* σ *is the softmax function. Then,* f^{max} *and* f^{min} *are vertices of* S*.*

Proof. S is an axis-parallel hyper-rectangle. As a consequence, it can be written as intersection of constraints of the form $-f_i \leq -k_{i,1}$ and $f_i \leq k_{i,2}$, where f_i is the i-th component of vector f. Hence, the optimization problem for the maximization case (minimization case is equivalent) can be rewritten as follows:

$$\begin{aligned}
&\max \sigma^c(f)\\
&\text{such that } \forall i \in \{1,...,|C|\} - f_i \leq -k_{i,1}, \quad f_i \leq k_{i,2}.
\end{aligned}$$

In order to solve this problem we can apply the Karush-Kuhn-Tucker (KKT) conditions. Being the constraints independent of f, the KKT conditions imply that in order to conclude the proof we just need to show that for all $f \in S, c \in \{1,...,|C|\}$, $\frac{d\sigma^c(f)}{df_c} \neq 0$. This is shown in what follows.

For $f \in \mathbb{R}^n$ and $c \in \{1, ...n\}$ We have

$$\sigma^c(f) = \frac{\exp(f_c)}{\sum_{j=1}^{C} \exp(f_j)}.$$

Then, we obtain

$$\frac{d\sigma^c(f)}{df_c} = \frac{\exp(f_c)(\sum_{j \neq c} \exp(f_j))}{(\sum_{j=1}^{C} \exp(f_j))^2},$$

while for $i \neq c$ we have

$$\frac{d\sigma^c(f)}{f_i} = -\frac{\exp(f_c)\exp(f_i)}{(\sum_{j=1}^{C} \exp(f_j))^2}.$$

This implies that for $f \in \mathbb{R}^n$ and $i \neq c$ we always have

$$\frac{d\sigma^c(f)}{df_c} > 0 \qquad \frac{d\sigma^c(f)}{df_i} < 0.$$

□

Note that in Lemma 2 we assumed that S is an hyper-rectangle. However, the lemma can be trivially extended to more general sets given by the intersection of arbitrarily many half-spaces generated by hyper-planes perpendicular to one of the axis.

The following Lemma is needed to prove Proposition 4.

Lemma 3. *Let X and Y be random variables with joint density function f. Consider measurable sets A and B. Then, it holds that*

$$P(X \in A | X_2 \in B) \leq \sup_{y \in B} P(X \in A | X_2 = y).$$

Proof.

$$\begin{aligned}
&P(X \in A | X_2 \in B) \\
&= \frac{P(X \in A \wedge X_2 \in B)}{P(X_2 \in B)} \\
&= \frac{\int_{x \in A} \int_{y \in B} f(X_1 = x \wedge X_2 = y) dx dy}{P(X_2 \in B)} \\
&= \frac{\int_{x \in A} \int_{y \in B} f(X_1 = x | X_2 = y) f(X_2 = y) dx dy}{P(X_2 \in B)} \\
&\leq \frac{\int_{x \in A} \int_{y \in B} \sup_{\bar{y} \in B} f(X_1 = x | X_2 = \bar{y}) f(X_2 = y) dx dy}{P(X_2 \in B)} \\
&= \frac{\int_{x \in A} \sup_{\bar{y} \in B} f(X_1 = x | X_2 = \bar{y}) dx \int_{y \in B} f(X_2 = y) dy}{P(X_2 \in B)} \\
&= \frac{\sup_{y \in B} P(X_1 \in A | X_2 = y) P(X_2 \in B)}{P(X_2 \in B)} \\
&= \sup_{y \in B} P(X_1 \in A | X_2 = y),
\end{aligned}$$

□

F Related work

Different notions of robustness for GPs have been studied in the literature [19, 20, 21, 17]. In this work we consider robustness against adversarial input perturbations, which is similar to settings considered in [19, 17], but differs from [20, 21], where robustness against labelling errors and outliers is studied.

Several papers address the problem of quantifying the robustness of the predictions of a Bayesian model with respect to input perturbations. One direction of work considers heuristic approaches based on studying adversarial examples on GPs and Bayesian neural networks [5, 22]. Formal guarantees are developed in [17], but only considering probabilistic reachability prior to decision making for *regression* problems. Their probabilistic approach can not be applied to the *classification* setting studied in this work. Some works for classification problems consider statistical guarantees [23]. In contrast, the approach we develop in this paper offers stronger (i.e., non-statistical) guarantees for classification problems by relying on the properties of GPs, which do not require any sampling for their computation.

Another relevant direction of work focuses on deriving *Probably Approximately Correct (PAC)* bounds on the generalization error for Bayesian models [24, 25]. However, these bounds are not directly applicable to our robustness estimation problem, as our focus is on analysing how, for a given test point, a perturbation applied to that point causes a prediction change, independently of the point ground truth.