Supplementary Material: ADVERSARIAL ROBUSTNESS GUARANTEES FOR CLASSIFICATION WITH GAUSSIAN PROCESSES

In the first Section of this Supplementary Material we present the proof of Propositions 1 and 2, as well as Theorem 1. Further technical results concerning multiclass classification are treated in Section B. In Section C we detail the case of binary classification using the probit likelihood function. In Section D we detail our approach for computing a lower bound of the predictive variance and mention how promising candidate points for the GPC bounding can be computed. We empirically analyse the computational complexity of the branch and bound methodology in a runtime analysis in Section E. In Section F we describe the datsets used and detail the experimental settings. Finally, in Section G, details for the interpretability metric we use in the experimental section are given.

A PROOFS FOR BINARY CLASSIFICATION BOUNDS

A.1 Proof of Proposition 1

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

$$\min_{x\in T}\pi(x|\mathcal{D})$$

(By definition)

$$=\min_{x\in T}\int_{-\infty}^{+\infty}\sigma(\bar{f})q(f(x)=\bar{f}|\mathcal{D})d\bar{f}$$

(By additivity of integrals)

$$=\min_{x\in T}\sum_{i=1}^{N}\int_{a_i}^{b_i}\sigma(\bar{f})q(f(x)=\bar{f}|\mathcal{D})d\bar{f}$$

(By monotonicity of σ and non-negativity of q)

$$\geq\min_{x\in T}\sum_{i=1}^{N}\int_{a_i}^{b_i}\sigma(a_i)q(f(x)=\bar{f}|\mathcal{D})d\bar{f}$$

(By definition of minimum and of q)

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

□

A.2 Proof of Proposition 2

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

$$\min_{x\in T}\int_a^b\mathcal{N}(\bar{f}|\mu(x),\Sigma(x))d\bar{f}\geq$$

$$\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\ \Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\int_a^b\mathcal{N}(\bar{f}|\mu,\Sigma)d\bar{f}=$$

$$\frac{1}{2}\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\ \Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\left(\text{erf}\left(\frac{\mu-a}{\sqrt{2\Sigma}}\right)-\text{erf}\left(\frac{\mu-b}{\sqrt{2\Sigma}}\right)\right):=$$

$$\frac{1}{2}\min_{\substack{\mu\in[\mu_T^L,\mu_T^U]\\ \Sigma\in[\Sigma_T^L,\Sigma_T^U]}}\Phi(\mu,\Sigma).$$

By looking at the partial derivatives we have that:

$$\frac{\partial\Phi(\mu,\Sigma)}{\partial\mu}=$$

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

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

$$\frac{\partial\Phi(\mu,\Sigma)}{\partial\Sigma}=$$

$$\frac{1}{\sqrt{2\pi\Sigma^3}}\left((\mu-b_i)e^{-\frac{(\mu-b_i)^2}{2\Sigma^2}}-(\mu-a_i)e^{-\frac{(\mu-a_i)^2}{2\Sigma^2}}\right)\geq 0$$

$$\Leftrightarrow\Sigma\leq\frac{(\mu-a)^2-(\mu-b)^2}{2\log\frac{\mu-a}{\mu-b}}:=\Sigma^m(\mu)$$

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

□

A.3 Proof of Theorem 1

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 $\bar{x}\in I$

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

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 can consider a uniform discretisation 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

$$|\pi(\bar{x}|\mathcal{D}) - \sum_{i=1}^{N} \sigma(a_i) \int_{a_i}^{b_i} \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i))d\bar{f}| \quad (11)$$

(By Definition)

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

(By additivity of integral and re-ordering terms)

$$= | \sum_{i=1}^{N} \Big(\int_{a_i}^{b_i} \sigma(\bar{f})\mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x}))d\bar{f} - \int_{a_i}^{b_i} \sigma(a_i)\mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i))d\bar{f}\Big)| \quad (13)$$

(As for any $\bar{f} \in S_i$, $\sigma(a_i) \leq \sigma(f_i) \leq \sigma(a_i) + \frac{1}{N}$)

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

(By Triangle Inequality)

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

(By Re-ordering terms and Triangle Inequality)

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

(By properties of integrals and $\sigma(f) \in [0, 1]$)

$$\leq \frac{1}{N} + \sum_{i=1}^{N} | \int_{a_i}^{b_i} (\mathcal{N}(\bar{f}|\mu(\bar{x}), \Sigma(\bar{x})) - \mathcal{N}(\bar{f}|\mu(x^i), \Sigma(x^i)))d\bar{f}| \quad (17)$$

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 $S_i \in S$

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

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 (17) is bounded by $\frac{\epsilon}{2}$.

□

B BOUNDS FOR MULTICALSS CLASSIFICATION

Proof of Proposition 3

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

$$\min_{x \in T} \pi^c(x|\mathcal{D})$$

(By definition)

$$= \min_{x \in T} \int \sigma^c(\bar{f}) q(f(x) = \bar{f}|\mathcal{D})d\bar{f}$$

(By additivity of integral)

$$= \min_{x \in T} \sum_{i=1}^{N} \int_{S_i} \sigma^c(\bar{f}) q(f(x) = \bar{f}|\mathcal{D})d\bar{f}$$

(Because q is non-negative)

$$\geq \min_{x \in T} \sum_{i=1}^{N} \int_{S_i} \min_{y \in S_i} \sigma^c(y) q(f(x) = \bar{f}|\mathcal{D})d\bar{f}$$

(By definition of infimum)

$$\geq \sum_{i=1}^{N} \min_{y \in S_i} \sigma^c(y) \min_{x \in T} \int_{S_i} q(f(x) = \bar{f}|\mathcal{D})d\bar{f}$$

(By Definition of q)

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

□

Proposition 3 in the main text implies that if we can compute infimum and supremum of the softmax over a set of the latent space (shown in Lemma 1) and the mean and covariance matrix that maximise a Gaussian integral (shown in Proposition 4), then upper and lower bounds on $\pi_{\min}(T)$ and $\pi_{\max}(T)$ can be derived.

Lemma 1. *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 optimisation problem for the maximisation case (minimisation case is equivalent) can be rewritten as follows:

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

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

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

Proof.

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

□

B.1 Proof of Proposition 4.

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

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

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 unidimensional Gaussian random variable, as a Gaussian random variable conditioned to a jointly Gaussian random variable is still Gaussian.

C BOUNDS FOR PROBIT BINARY CLASSIFICATION

For the case that the likelihood 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 $q(f(x) = \bar{f}|\mathcal{D})$ Bishop (2006). 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. This can be obtained by direct inspection of the derivatives of $\pi(x|\mathcal{D})$.

Lemma 3. *Let $T \subseteq \mathbb{R}^d$. Then, we have that*

$$\Phi\left(\frac{\mu_T^L}{\sqrt{\lambda^{-2} + \underline{\Sigma}}}\right) \leq \pi_{\min}(T) \tag{18}$$

and

$$\pi_{\max}(T) \leq \Phi\left(\frac{\mu_T^U}{\sqrt{\lambda^{-2} + \overline{\Sigma}}}\right) \tag{19}$$

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.

D BOUNDS ON LATENT MEAN AND VARIANCE

In this section of the Supplementary Material we briefly review how lower and upper bounds on the a-posteriori mean and variance can be computed, and further show how this give us candidate points for the evaluation of bounds (that is line 6 in Algorithm 1 of the main paper).

We obtain bounds on latent mean and variance by applying the framework presented in Cardelli et al. (2019b) for computation of μ_T^L, μ_T^U and Σ_T^U, and subsequently extend it for the computation of Σ_T^L. Briefly, assuming continuity and differentiability of the kernel function defining the GPC covariance, it is possible to find linear upper and lower bounds on the covariance vector, which can be propagated through the inference formula for $q(f(x) = \bar{f}|\mathcal{D})$. The bounding functions obtained in this way can be analytically optimised for μ_T^L and μ_T^U, while convex quadratic programming is used to obtain Σ_T^U (see Cardelli et al. (2019b) for details). Finally, we solve the concave quadratic problem that arises when computing Σ_T^L by adapting methods introduced in Rosen & Pardalos (1986), which reduces the problem to the solution of $2|\mathcal{D}| + 1$ linear programming problems. This is detailed in the following subsection.

As discussed in Section 4 in order to obtain $\pi_{\min}^U(T)$ and $\pi_{\max}^L(T)$ it suffice to evaluate the GPC in any point inside T. However, the more close $\pi_{\min}^U(T)$ and $\pi_{\max}^L(T)$ are to $\pi_{\min}(T)$ and $\pi_{\max}(T)$ respectively, the more quicker will be the convergence of the branch and bound algorithm (as per line 7 in Algorithm 1 in the main paper). Notice that, in solving the optimisation problems associated to μ_T^L, μ_T^U, Σ_T^U and Σ_T^L we obtain four extrema points in T on which the GPC assume the optimal values a-posteriori mean and variance values. As these points belong to T and provide extreme points for the latent function they make promising candidates for the evaluation of $\pi_{\min}^U(T)$ and $\pi_{\max}^L(T)$. Specifically in line 6 of Algorithm 1 (main paper), we evaluate the GPC on all four the extrema and select the one that gives the best bound among them.

D.1 Lower Bound on Latent Variance

Let $\mathbf{r}(x) = [r_1(x), \ldots, r_M(x)]$ 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 $\mathbf{r} = [r_1, \ldots, r_M]$, that is $-\mathbf{r}^T R \mathbf{r}$. Analogously to what is done in Cardelli et al. (2019b) 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:

$$\begin{aligned} &\min -\mathbf{r}^T R \mathbf{r} \\ &\text{Subject to:} \quad A_r \mathbf{r} + A_x x \leq b \\ &\qquad\qquad r_i^L \leq r_i \leq r_i^U \qquad i = 1, \ldots, M \\ &\qquad\qquad x_i^L \leq x_i \leq x_i^U \qquad i = 1, \ldots, m \end{aligned} \tag{20}$$

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 Rosen & Pardalos (1986) whose exact solution would make a branch and bound algorithm based on it impractical. Following the methods discussed in Rosen & Pardalos (1986), 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 $U = [\mathbf{u}_1, \ldots, \mathbf{u}_M]$ and a diagonal matrix of the associated eigenvalues λ_i for $i = 1, \ldots, M$, 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:

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

Implementing the change of variables into Problem 20 we obtain:

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

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 Cardelli et al. (2019b) 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 20 and can be hence used to compute a lower bound to the latent variance:

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

E RUNTIME ANALYSIS

In this section of the Supplementary Material we empirically analyse the CPU time required for convergence of Algorithm 1 in the MNIST38 dataset. For the first 50 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 analysed below 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.

E.1 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 10. The results are shown in terms of average runtime in Figure 7 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 almost 250 seconds beyond 7 dimensions. However, for $\gamma = 0.125$ the exponential curve seems to be shifted further to the right, as still for 10 dimensions Algorithm 1 terminates 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.

E.2 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 5. The results are shown in Figure 7 on the right. The behaviour seems to be roughly inversely exponential this time with lower error tolerance 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 terminates in under 2 seconds on average even for $\gamma = 0.25$.

F EXPERIMENTAL SETTINGS

F.1 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.

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.

MNIST38 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.

The subset of MNIST which we use to compile MNIST358 contains 5,715 samples of images of handwritten digits, of which roughly 36 % are 3s, 31 % are 5s and 34 % are 8s.

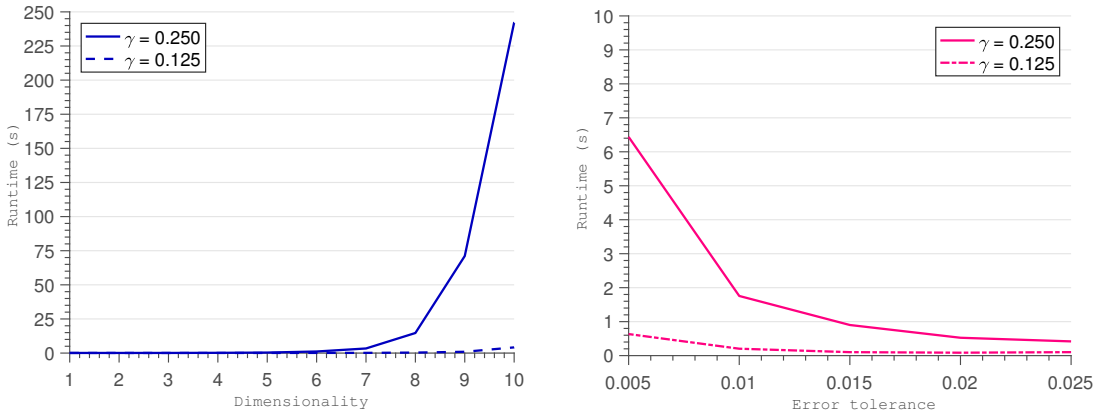

Figure 7: Average runtimes of Algorithm 1 to calculate $\pi_{\max}(T)$ up to specified error tolerance ϵ among the first 50 test points of MNIST38. **Left**: Average runtimes for increasing number of dimensions at $\epsilon = 0.025$. **Right**: Average runtimes for different values of ϵ with number of dimensions $d = 5$.

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.

F.2 Experimental Settings

For the binary experiments, we use 1,000 randomly selected points as a training set and 200 randomly selected points as a test set. For the multiclass experiments, scalability of GPs is even more of an issue so we just work with 500 randomly selected points as training set.

For the GP training of binary classification problems, we use the GPML package for Matlab. For the GP training of multiclass classification problems, we use the GPstuff package.

For the safety verification experiments in Section 6.1, we used a GPC model with a probit likelihood function and the Laplace approximation for the posterior. For the synthetic 2D data, the number of epochs (marginal likelihood evaluations) performed during hyper-parameter optimisation was restricted to 20. For the SPAM data, it was restricted to 40. Finally for the attacks on MNIST38 it was restricted to 10 and 20.

For the robustness experiments in Section 6.2, we give the specifications of training in the paper itself.

For the interpretability experiments in Section 6.3, we use a multiclass GPC model with softmax link function and the Laplace approximation for the posterior. We limit the number of iterations performed during hyper-parameter optimisation to 10.

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

G DETAILS ON INTERPRETABILITY METRIC

Below, we briefly derive our metric for interpretability analysis $\mathbf{\Delta}^i_\gamma$, which by using our bounds does not rely on local linearity , in a bit more detail.

For a testpoint x^* and dimension i, we define $T^i_\gamma(x^*) = [x^*, x^* + \gamma * e_i]$ like in the main paper. To analyse the impact of changes in dimension i, we propose to analyse how much the maximum of the assigned class probabilities can differ from the initial class probability $\pi(x^*)$ over such a one-sided interval compared to how much the minimum differs from that initial probability. In other words, we calculate

$$\Delta^i_\gamma(x^*) = \left(\pi_{\max}(T^i_\gamma(x^*)) - \pi(x^*)\right) \tag{21}$$
$$- \left(\pi(x^*) - \pi_{\min}(T^i_\gamma(x^*))\right). \tag{22}$$

If increasing the value of dimension i makes the model favor assigning lower class probabilities, we would expect this value to be negative and vice versa. To make it more robust, we center the analysis by calculating the proposed metric

$$\mathbf{\Delta}^{\mathbf{i}}_\gamma(x^*) = \Delta^i_\gamma(x^*) - \Delta^i_{-\gamma}(x^*) \tag{23}$$
$$= \left(\pi_{\max}(T^i_\gamma(x^*)) - \pi_{\max}(T^i_{-\gamma}(x^*))\right) \tag{24}$$
$$+ \left(\pi_{\min}(T^i_\gamma(x^*)) - \pi_{\min}(T^i_{-\gamma}(x^*))\right). \tag{25}$$

Finally, if instead of a local analysis a global analysis is desired, we suggest following LIME's approach in aggregating local insights to a global insight by averaging over a selection of test points M

$$\mathbf{\Delta}^{\mathbf{i}}_\gamma = \frac{1}{M} \sum_{j=1}^{M} \mathbf{\Delta}^{\mathbf{i}}_\gamma(x^j). \tag{26}$$

Ideally, M contains all test points; however, if for computational reasons a subselection is to be made, the SP algorithm in Ribeiro et al. (2016) could be used.