Local Robustness Quantification for Naive Bayes Classifiers and Generative Forests: a General Approach

Adrián Detavernier ADRIAN.DETAVERNIER@UGENT.BE
Jasper De Bock JASPER.DEBOCK@UGENT.BE
Foundations Lab for imprecise probabilities
Ghent University
Ghent, Belgium

Editors: Gustau Camps-Valls, Manuele Leonelli and Gherardo Varando

Abstract

We provide methods for calculating the robustness of the predictions of two types of generative classifiers whose underlying distribution is a Probabilistic Graphical Model (PGM): naive Bayes classifiers and generative forests (a probabilistic extension of random forests). Following the paradigm of robustness quantification, we define the robustness of a prediction as the extent to which the distribution of the classifier can be perturbed without changing this prediction. We consider perturbations obtained by varying the local models of the PGMs within general neighborhoods and focus in particular on epsilon-contamination, total variation distance and chi-squared divergence balls. We test our methods on benchmark datasets, demonstrate that the robustness value of a prediction serves as an indicator for its trustworthiness and compare our approach with other such indicators.

Keywords: trustworthy classification, generative forests, naive Bayes classifier, imprecise probabilities

1. Introduction

Say that you are writing a paper and the deadline is approaching, so you have to quickly finish the paper and submit it. However, the central proof of the paper is not working yet, so, to save time, you decide to consult a Large Language Model (LLM) to write the proof for you. You get the proof from the LLM, but you realize that time's up and that you are not able to check the proof before submitting the paper. [1] Would you trust the proof of the LLM? Would you take the risk? Or would you only do it if the LLM is extremely confident about the proof? If such a (hopefully unrealistic) situation arises, it would be very useful if the LLM were to give us some indication of the trustworthiness of its output. Similarly for other high-stakes settings, such as medical diagnosis, it would be very useful if a model were able to assess the trustworthiness of its own predictions such that the user could take this information into account when making decisions based on these predictions.

In this paper, we consider one possible approach that attempts to accompany the individual predictions of a classifier with an indication of their trustworthiness: Robustness Quantification (RQ) (Detavernier and De Bock, 2025). RQ in particular quantifies the so-called *robustness* of a classifier's predictions, which is the extent to which the model can be perturbed without changing the prediction, the idea being that predictions with high robustness are more trustworthy because they are less sensitive to errors made while training

1. Disclaimer: this situation is purely hypothetical and does not reflect the actual practices of the author(s).

© 2026 A. Detavernier & J. De Bock.

the model. RQ has so far been studied for classifiers based on several types of Probabilistic Graphical Models (PGMs): the Naive Bayes Classifier (NBC) (Detavernier and De Bock, 2025), for Generative Forests (GeFs) (Correia et al., 2020b), and for Bayesian networks and Markov random fields (De Bock et al., 2014). These results have furthermore shown that the robustness of a prediction, defined in this way, can be a useful indicator for its trustworthiness, in the sense that predictions with a higher robustness value tend to be correct more often than predictions with a lower robustness value.

In this work, we show how RQ can be reduced to an optimization problem. Additionally, we use ideas from imprecise probability theory (Augustin et al., 2014) and credal classification (Corani et al., 2014) to provide general methods for reducing this optimization problem to local optimizations if the PGM is an NBC or GeF that is perturbed by local perturbations of its local probability or weight functions, respectively. In doing so, we generalize existing methods that focussed on ε-contamination perturbations. We illustrate this generality by considering as perturbations total variation distance and chi-squared divergence balls, and we show in our experiments that the resulting robustness values still correlate with accuracy, and moreover that they are competitive with those obtained with ε-contamination and with indicators of trustworthiness that are based on uncertainty.

2. Classification: Setting and Notation

To formalize the setting, namely classification with discrete features, we introduce some notation. The goal of a classifier is to predict the class c of an instance, which is an element of a finite set of classes $\mathcal{C}$. When the class is unknown, we refer to it as the class variable C. To make this prediction, we typically have some information about the instance in the form of a set of features $\boldsymbol{f}$. Each of the features is denoted by a variable F_i and takes values f_i in a finite set $\mathcal{F}_i$, with $i \in \{1, \ldots, N\}$, where N is the number of features. We will collect all the features of an instance in a vector, and refer to this vector as the feature vector or just the features of the instance. The feature vector of an instance is given by $\boldsymbol{f} = (f_1, \ldots, f_N)$, where f_i is the value of the i-th feature for this instance. All such possible feature vectors are collected in the set $\boldsymbol{\mathcal{F}} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_N$. Every instance is thus determined by the combination of a class c and its features $\boldsymbol{f}$. A classifier $h : \boldsymbol{\mathcal{F}} \to \mathcal{C}$ can therefore be seen as a function that maps feature vectors to classes. In the ideal scenario, the predicted class $\hat{c} := h(\boldsymbol{f})$ is the same as the true class of the instance. In practice, however, this is not always the case. That is exactly why assessing the trustworthiness of the predictions of a classifier is important.

This contribution focusses on generative classifiers that first estimate a joint probability mass function p on $\mathcal{C} \times \boldsymbol{\mathcal{F}}$, and then predict the class $\hat{c}$ that maximizes the conditional probability $p(c|\boldsymbol{f}) = {}^{p(c,\boldsymbol{f})}/_{p(\boldsymbol{f})}$, with $p(\boldsymbol{f}) := \sum_{c' \in \mathcal{C}} p(c', \boldsymbol{f})$. This $\hat{c}$ may not be unique though, or $p(c|\boldsymbol{f})$ might be ill-defined if $p(\boldsymbol{f}) = 0$, which is why formally, given a set of features $\boldsymbol{f}$, a generative classifier $h_p : \boldsymbol{\mathcal{F}} \to \mathcal{C}$ chooses the predicted class $\hat{c}$ from the set

$$\hat{\mathcal{C}}_{p,\boldsymbol{f}} := \begin{Bmatrix} \arg\max_{c \in \mathcal{C}} p(c|\boldsymbol{f}) & \text{if } p(\boldsymbol{f}) > 0 \\ \mathcal{C} & \text{if } p(\boldsymbol{f}) = 0 \end{Bmatrix} = \arg\max_{c \in \mathcal{C}} p(c, \boldsymbol{f}). \tag{1}$$

In practice, $\hat{\mathcal{C}}_{p,\boldsymbol{f}}$ will of course typically be a singleton, in which case $\hat{c}$ is completely determined by p, but our methods also work for features for which this is not the case.

3. Robustness of a Prediction

The distribution p learned by a probabilistic classifier is typically not the true data generating distribution; data subject to distribution shift, the use of a specific model architecture, or randomness in the training process/model parameters can all lead to deviations of p from the true distribution, and hence to a suboptimal prediction. So, what if we would slightly change the distribution p that the model learned? Would the prediction change? How much would we have to change the distribution before the prediction changes? RQ answers this last question by quantifying how much the learned distribution p of the model can be changed without changing the resulting prediction of the classifier. To formalize this idea, we consider perturbations of the mass function p, which are simply sets of mass functions that contain p.

Let Ω be a finite set of outcomes and let Σ_Ω be the set of all probability mass functions on Ω: maps from Ω to $[0, 1]$ that sum up to one.

Definition 1 *Consider a mass function $p \in \Sigma_\Omega$ on a finite set of outcomes Ω. A* **perturbation** *of p is a compact set $\mathcal{P} \subseteq \Sigma_\Omega$ of mass functions on Ω such that $p \in \mathcal{P}$.*

This is a general and deliberately vague definition since it does not specify how the perturbation is constructed. In practice, we will usually construct such a perturbation as a neighborhood around p, with $p \in \Sigma_{\mathcal{C}\times\boldsymbol{\mathcal{F}}}$ a mass function learned by a probabilistic classifier.

Given a perturbation $\mathcal{P}$ of p and set of features $\boldsymbol{f}$, we can now construct the set of possible predictions $\hat{\mathcal{C}}_{\mathcal{P},\boldsymbol{f}} := \bigcup_{p' \in \mathcal{P}} \hat{\mathcal{C}}_{p',\boldsymbol{f}}$ that are compatible with at least one of the perturbed models $p' \in \mathcal{P}$. The prediction $\hat{c}$ is said to be robust w.r.t. the perturbation $\mathcal{P}$ if the only such compatible prediction is $\hat{c}$ itself.

Definition 2 *Let h_p be a generative classifier corresponding to a mass function p. Let $\hat{c}$ be the prediction according to h_p for the set of features $\boldsymbol{f}$, let $\mathcal{P}$ be a perturbation of p, and let $\hat{\mathcal{C}}_{\mathcal{P},\boldsymbol{f}}$ be the set of possible predictions w.r.t. the perturbation $\mathcal{P}$. Then $\hat{c}$ is* **robust** *w.r.t. the perturbation $\mathcal{P}$ if $\hat{\mathcal{C}}_{\mathcal{P},\boldsymbol{f}} = \{\hat{c}\}$.*

For readers that are familiar with imprecise probabilities and credal classification, these ideas should be familiar. In that context, our perturbations would typically be called credal sets and $\hat{\mathcal{C}}_{\mathcal{P},\boldsymbol{f}}$ would be an example of a corresponding credal classifier: a classification procedure that outputs a set of classes.[2] Robustness of an instance then corresponds to this credal classifier being determinate, in the sense that its output contains only a single class. An important observation in credal classification, which inspired the development of RQ (De Bock et al., 2014), is that traditional classifiers tend to perform worse on instances where their credal counterparts remain indeterminate (Corani and Zaffalon, 2008).

The ideas behind RQ are similar, but it has different aims. First, RQ is not interested in determining $\hat{\mathcal{C}}_{\mathcal{P},\boldsymbol{f}}$, but focusses solely on robustness (determinacy). Second, instead of

2. Many different imprecise decision criteria can be used to associate such a set of classes with a credal set Troffaes (2007). Our approach, which simply gathers the predictions of the different distributions in the credal set, essentially corresponds to the use of E-admissibility; it has the advantage of having an intuitive sensitivity-analysis interpretation. Credal classification instead typically uses another decision criterion, called maximality, for which the corresponding set of classes can more easily be determined using optimization techniques. For the purposes of determining robustness, however, this choice does not matter. For readers familiar with maximality, this should follow easily from Theorem 2 further on.

determining whether a prediction is robust w.r.t. a given fixed perturbation, RQ instead wants to quantify *how* robust it is. RQ does this by controlling the size of the perturbation in a parametrized manner and increasing this size until the prediction of the model is no longer robust, or thus until at least one distribution in the neighborhood predicts a different class. This smallest perturbation size for which the prediction is no longer robust is then used as a numeric measure of robustness. To formalize this, we consider *parametrized perturbations*.

Definition 3 *Consider a mass function $p \in \Sigma_\Omega$ on a finite set of outcomes Ω. Let $\mathcal{P}^\delta$ be a perturbation of p for all $\delta \in \Delta \subseteq \mathbb{R}_{\geq 0}$, where Δ is a set of possible values for δ that includes zero. Then the family $\mathcal{P}^\Delta := \left(\mathcal{P}^\delta\right)_{\delta\in\Delta}$ is called a* **parametrized perturbation** *of p if the following conditions hold: (1) if $\delta = 0$, then $\mathcal{P}^0 = \{p\}$; and (2) if $\delta_1 \leq \delta_2$, then $\mathcal{P}^{\delta_1} \subseteq \mathcal{P}^{\delta_2}$.*

Clearly, the bigger the value of δ, the bigger the perturbation, in the sense that it contains more distributions. Concrete examples of such parametrized perturbations are given in Section 6. Next, we formalize the numeric measure of robustness.

Definition 4 *Let h_p be a generative classifier corresponding to a mass function p, and let $\hat{c}$ be the prediction according to h_p for the set of features $\boldsymbol{f}$. Let $\mathcal{P}^\Delta$ be a parametrized perturbation of p, with $\Delta \subseteq \mathbb{R}_{\geq 0}$ the possible values for the parameter δ. Then the* **robustness (value)** *$r_{\mathcal{P}^\Delta}(\boldsymbol{f})$ w.r.t. $\mathcal{P}^\Delta$ of an instance with features $\boldsymbol{f}$ is the infimum $\delta \in \Delta$ for which the prediction $\hat{c}$ is no longer robust w.r.t. the perturbation $\mathcal{P}^\delta$. If there is no such δ, then we set $r_{\mathcal{P}^\Delta}(\boldsymbol{f}) = +\infty$.*

Henceforth, whenever we use the notation $\mathcal{P}^\delta$ for a perturbation, we implicitly assume that it is part of a parametrized perturbation $\mathcal{P}^\Delta$ for some set Δ of possible values for δ.

This definition faithfully captures the ideas behind RQ, but it isn't the most practical one because, at first sight, checking whether a prediction is robust w.r.t. a perturbation would require us to determine the predictions of all the distributions in the perturbation. To address this issue, we now proceed to provide an equivalent, more practical, characterization of robustness. This characterization is expressed in terms of the lower expectation of a perturbation $\mathcal{P}$, defined for any real-valued function f on Ω by

$$\underline{E}_{\mathcal{P}}(f) := \min_{p'\in\mathcal{P}} E_{p'}(f), \tag{2}$$

where $E_{p'}(f) = \sum_{x\in\Omega} p'(x)f(x)$ is the expectation of f w.r.t. the mass function p'. If $\Omega = \mathcal{C} \times \mathcal{Y}$ for some set $\mathcal{Y}$, then for any $c \in \mathcal{C}$, we use $\mathbb{I}_c$ to denote the function that takes the value 1 if $x \in \{c\} \times \mathcal{Y}$ and 0 otherwise. If $\Omega = \boldsymbol{\mathcal{F}} \times \mathcal{Y}$, $\mathbb{I}_{\boldsymbol{f}}$ is defined similarly.

Theorem 5 *Let h_p be a generative classifier corresponding to a mass function p, let $\hat{c}$ be the prediction according to h_p for the set of features $\boldsymbol{f}$, and let $\mathcal{P}$ be a perturbation of p. Then $\hat{c}$ is* robust *w.r.t. the perturbation $\mathcal{P}$ if and only if*

$$\forall p' \in \mathcal{P}, \forall c \in \mathcal{C}\backslash\{\hat{c}\} : p'(\hat{c}, \boldsymbol{f}) - p'(c, \boldsymbol{f}) > 0, \tag{3}$$

or, equivalently, if

$$\min_{c\in\mathcal{C}\backslash\{\hat{c}\}} \underline{E}_{\mathcal{P}}\left[(\mathbb{I}_{\hat{c}} - \mathbb{I}_c)\mathbb{I}_{\boldsymbol{f}}\right] > 0. \tag{4}$$

Proof By Definition 2 and Equation (1), $\hat{c}$ is robust w.r.t. $\mathcal{P}$ if and only if $\hat{\mathcal{C}}_{p',\boldsymbol{f}} = \{\hat{c}\}$ for all $p' \in \mathcal{P}$. This holds if and only if $\forall p' \in \mathcal{P}, \forall c \in \mathcal{C}\backslash\{\hat{c}\}\colon p'(\hat{c}, \boldsymbol{f}) - p'(c, \boldsymbol{f}) > 0$, which proves the first equivalence of the theorem. Since $\mathcal{P}$ is compact, the previous condition is equivalent to requiring that the minimum of this difference over all $c \in \mathcal{C}\backslash\{\hat{c}\}$ and all $p' \in \mathcal{P}$ is positive: $\min_{c\in\mathcal{C}\backslash\{\hat{c}\}} \min_{p'\in\mathcal{P}} p'(\hat{c}, \boldsymbol{f}) - p'(c, \boldsymbol{f}) > 0$. Since $p'(\hat{c}, \boldsymbol{f}) - p'(c, \boldsymbol{f}) = E_{p'}\left[\mathbb{I}_{\hat{c}}\mathbb{I}_{\boldsymbol{f}} - \mathbb{I}_c\mathbb{I}_{\boldsymbol{f}}\right]$, this is equivalent to $\min_{c\in\mathcal{C}\backslash\{\hat{c}\}} \underline{E}_{\mathcal{P}}\left[(\mathbb{I}_{\hat{c}} - \mathbb{I}_c)\mathbb{I}_{\boldsymbol{f}}\right] > 0$, as claimed. ■

Consequently, checking whether a prediction is robust w.r.t. a perturbation $\mathcal{P}$ only requires us to check Equation (4) of Theorem 5. Furthermore, note that the function $R : \Delta \to \mathbb{R} : \delta \mapsto \min_{c\in\mathcal{C}\backslash\{\hat{c}\}} \underline{E}_{\mathcal{P}^\delta}\left((\mathbb{I}_{\hat{c}} - \mathbb{I}_c)\mathbb{I}_{\boldsymbol{f}}\right)$ is non-increasing, since the bigger the perturbation, the smaller the lower expectation. Finding the robustness value therefore amounts to finding the infimum δ for which $R(\delta) = 0$; that is, we need to find the smallest root of a non-increasing function. In practice, the function R will often be continuous and strictly decreasing, which makes this root-finding problem even easier to solve. Several methods exist for finding roots of such functions efficiently, which make the quantification of robustness feasible in practice.

Now that we have the theoretical tools to quantify the robustness of predictions of classifiers, we will apply these tools to two specific types of generative classifiers.

4. Naive Bayes Classifier

The first type of generative classifier we will be looking at is the Naive Bayes Classifier (NBC). This is a very simple model that is often used as a good baseline for more complex classifiers. The central assumption on which the Naive Bayes Classifier is based is that the features are conditionally independent given the class. This implies that there is a mass function $p_C \in \Sigma_{\mathcal{C}}$ and, for all $c \in \mathcal{C}$ and $i \in \{1, \ldots, N\}$, a mass function $p_{F_i|c} \in \Sigma_{\mathcal{F}_i}$ such that

$$p(c, \boldsymbol{f}) = p_C(c) \prod_{i=1}^{N} p_{F_i|c}(f_i), \quad \text{for all } c \in \mathcal{C} \text{ and } \boldsymbol{f} \in \boldsymbol{\mathcal{F}}. \tag{5}$$

We now consider a specific parametrized perturbation for such NBCs; it is a parametrized version of the model of the Naive Credal Classifier (Zaffalon, 2002)—a generalization of an NBC based on imprecise probabilities. In particular, we consider a local perturbation $\mathcal{P}^\delta_C$ of p_C and local perturbations $\mathcal{P}^\delta_{F_i|c}$ of $p_{F_i|c}$ for all $c \in \mathcal{C}$ and all $i \in \{1, \ldots, N\}$, and let

$$\mathcal{P}^\delta_{\text{NBC}} \coloneqq \left\{p' \in \Sigma_{\mathcal{C}\times\boldsymbol{\mathcal{F}}}\colon p'_C \in \mathcal{P}^\delta_C,\ p'_{F_i|c} \in \mathcal{P}^\delta_{F_i|c},\ p'(c, \boldsymbol{f}) = p'_C(c) \prod_{i=1}^{N} p'_{F_i|c}(f_i)\right\} \tag{6}$$

be the corresponding perturbation of p. This is simply the set of all NBCs whose local models are taken from $\mathcal{P}^\delta_C$ and $\mathcal{P}^\delta_{F_i|c}$. The superscript δ indicates that these perturbations are part of a parametrized perturbation where the parameter δ controls the size of the perturbations, enabling us to associate a robustness value with each instance.

Since the local models of the NBC are independent, and all models in the perturbation $\mathcal{P}^\delta_{\text{NBC}}$ are also NBCs, we can use this structure to simplify the condition for robustness given by Theorem 5 even further.

Theorem 6 *Let h_p be a Naive Bayes classifier with local models p_C and $p_{F_i|c}$ for all $c \in \mathcal{C}$ and all $i \in \{1, \ldots, N\}$ and corresponding joint mass function p defined by Equation (5). Let $\hat{c}$ be the prediction according to h_p for the set of features $\boldsymbol{f}$. Consider local perturbations $\mathcal{P}^{\delta}_{C}$ and $\mathcal{P}^{\delta}_{F_i|c}$ for all $c \in \mathcal{C}$ and all $i \in \{1, \ldots, N\}$. Then $\hat{c}$ is robust w.r.t. the corresponding perturbation $\mathcal{P}^{\delta}_{\text{NBC}}$ of p if and only if*

$$\min_{c \in \mathcal{C} \setminus \{\hat{c}\}} \underline{E}_{\mathcal{P}^{\delta}_{C}} \left(\mathbb{I}_{\hat{c}} \prod_{i=1}^{N} \underline{p}_{F_i|\hat{c}}(f_i) - \mathbb{I}_{c} \prod_{i=1}^{N} \overline{p}_{F_i|c}(f_i) \right) > 0, \tag{7}$$

where $\underline{p}_{F_i|\hat{c}}(f_i) = \min_{p'_{F_i|\hat{c}} \in \mathcal{P}^{\delta}_{F_i|\hat{c}}} p'_{F_i|\hat{c}}(f_i)$ and $\overline{p}_{F_i|c}(f_i) = \max_{p'_{F_i|c} \in \mathcal{P}^{\delta}_{F_i|c}} p'_{F_i|c}(f_i)$.

Proof By Theorem 5 $\hat{c}$ is robust w.r.t. $\mathcal{P}^{\delta}_{\text{NBC}}$ if and only if

$$\min_{c \in \mathcal{C} \setminus \{\hat{c}\}} \min_{p' \in \mathcal{P}^{\delta}_{\text{NBC}}} p'(\hat{c}, \boldsymbol{f}) - p'(c, \boldsymbol{f}) > 0. \tag{8}$$

Due to Equation (5) and the structure of $\mathcal{P}^{\delta}_{\text{NBC}}$ in Equation (6), this is equivalent to

$$\min_{c \in \mathcal{C} \setminus \{\hat{c}\}} \min_{p'_C \in \mathcal{P}^{\delta}_{C}} \min_{\forall c \in \mathcal{C}, \forall i \in \{1, \ldots, N\}:\ p'_{F_i|c} \in \mathcal{P}^{\delta}_{F_i|c}} p'_C(\hat{c}) \prod_{i=1}^{N} p'_{F_i|\hat{c}}(f_i) - p'_C(c) \prod_{i=1}^{N} p'_{F_i|c}(f_i) > 0. \tag{9}$$

To minimize this difference, we want to minimize the first (positive) term and maximize the negation of the second (negative) term. Because of the independence of the local mass functions in the above expression, we can rewrite it as

$$\min_{c \in \mathcal{C} \setminus \{\hat{c}\}} \min_{p'_C \in \mathcal{P}^{\delta}_{C}} \left(p'_C(\hat{c}) \prod_{i=1}^{N} \min_{p'_{F_i|\hat{c}} \in \mathcal{P}^{\delta}_{F_i|\hat{c}}} p'_{F_i|\hat{c}}(f_i|\hat{c}) - p'_C(c) \prod_{i=1}^{N} \max_{p'_{F_i|c} \in \mathcal{P}^{\delta}_{F_i|c}} p'_{F_i|c}(f_i|c) \right) > 0. \tag{10}$$

By using the definitions of the local lower and upper probabilities and local lower expectations, the above condition is equivalent to

$$\min_{c \in \mathcal{C} \setminus \{\hat{c}\}} \underline{E}_{\mathcal{P}^{\delta}_{C}} \left(\mathbb{I}_{\hat{c}} \prod_{i=1}^{N} \underline{p}_{F_i|\hat{c}}(f_i) - \mathbb{I}_{c} \prod_{i=1}^{N} \overline{p}_{F_i|c}(f_i) \right) > 0, \tag{11}$$

as claimed. ■

So, for NBCs the robustness value of an instance can be found by finding the smallest value δ for which the condition of Theorem 6 is not satisfied anymore.

5. Generative Forests

Even in times when deep learning models are dominating the field of machine learning, tree-based models are still very much relevant (Grinsztajn et al., 2022), while being more interpretable and easier to train. For this reason, and because of their easy to work-with structure, we choose to also apply RQ to tree-based models. To be able to apply our approach, we opt for Generative Forests (GeFs), that, in contrast to most tree-based

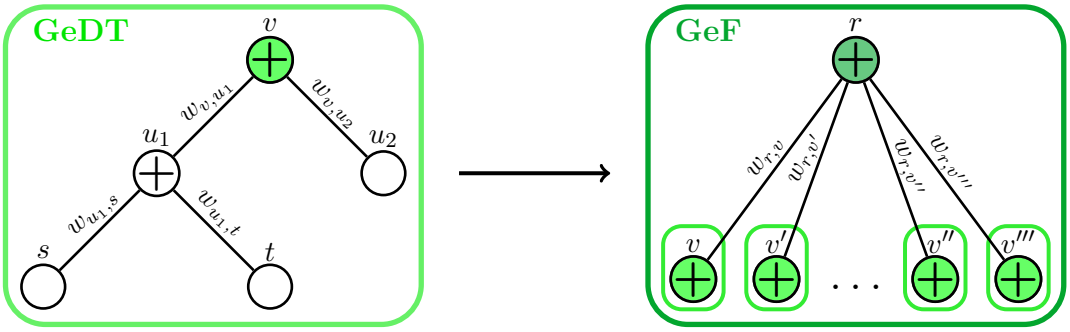


Figure 1: Visualization of the structures of a GeDT and a GeF.

models, are generative classifiers. A Generative Forest is either an ensemble of Generative Decision Trees (GeDTs) (Correia et al., 2020a) or a uniform mixture of GeDTs (Correia et al., 2020b); a GeDT can be seen as generative version of a Decision Tree (DT). Because the mixture variant has a single joint distribution, it is more suitable for our purposes than the ensemble one. We will therefore focus on the mixture variant.

A GeDT can be seen as a very simple Probabilistic Circuit (PC) with, in our case, a binary tree structure containing only leaf and sum nodes. A GeF combines several GeDTs into one PC, where the root node is a sum node of which all the children are the root of a GeDT, as shown in Figure 1. If v is a node in a GeF, then its children are collected in $\mathrm{ch}(v)$, its descendants in $\mathrm{de}(v)$ (excluding v itself), and its sum node descendants (excluding the leaf nodes) in $\mathrm{sde}(v)$. Note that each node in a such a tree is the root of a subtree containing all its descendants and itself. If a node has no children, so $|\mathrm{ch}(v)| = 0$, then it is a leaf node; otherwise it is a sum node. A sum node v has for each of its children $u \in \mathrm{ch}(v)$ a weight $w_{v,u} \geq 0$ such that $\sum_{u \in \mathrm{ch}(v)} w_{v,u} = 1$. For the root node r we set all weights equal, so $w_{r,v} = \frac{1}{|\mathrm{ch}(r)|}$ for all $v \in \mathrm{ch}(r)$. The vector of all weights of a sum node v is denoted by w_v and the collection of all weight vectors of the sum nodes in a GeF rooted at v is denoted by $\overline{w}_v := \{w_v\} \times \bigtimes_{u \in \mathrm{sde}(v)} \{w_u\}$.

A selling point of GeDTs and GeFs is that they can specify complex joint distributions $p \in \Sigma_{\mathcal{C} \times \boldsymbol{\mathcal{F}}}$ in such a way that they can be easily computed. They do so by associating simple joint distributions with each of their leaves and then using the sum nodes to take weighted averages of these simple distributions. Following Correia et al. (2020b), for each leaf node l, its distribution p_l is taken to be factorized over the class and features, meaning that $p_l(c, \boldsymbol{f}) = p_{l,C}(c) \prod_{i=1}^{N} p_{l,F_i}(f_i)$ for all $c \in \mathcal{C}$ and $\boldsymbol{f} \in \boldsymbol{\mathcal{F}}$, where $p_{l,C}$ is a local mass function on $\mathcal{C}$ and p_{l,F_i} a local mass function on $\mathcal{F}_i$ for all $i \in \{1, \ldots, N\}$. In the case of a sum node, its joint distribution p_v is a weighted sum of the joint distributions of its children, given by $p_v(c, \boldsymbol{f}) = \sum_{u \in \mathrm{ch}(v)} w_{v,u} p_u$ for any $c \in \mathcal{C}$ and $\boldsymbol{f} \in \boldsymbol{\mathcal{F}}$. In this way, the collection of all weights $\overline{w}_v$ of a GeDT or GeF rooted at v, together with the local models of the leaves, completely determine its joint distribution p_v. To make this explicit, we denote the joint distribution of the tree rooted at v by $p_{\overline{w}_v}$, leaving the dependency on $p_{l,C}$ and p_{l,F_i} implicit. In particular, the joint distribution of a GeF with root r is denoted by $p_{\overline{w}_r}$.

The main goal of GeFs is to classify each instance based on its set of features $\boldsymbol{f}$. Let r be the root node of a GeF with joint distribution $p_{\overline{w}_r}$, then the predicted class $\hat{c} = h_{p_{\overline{w}_r}}(\boldsymbol{f})$

is chosen from $\hat{\mathcal{C}}_{p_{\overline{w}_r},\boldsymbol{f}}$ as defined in Equation (1). We now want to assess the robustness of this prediction by constructing neighborhoods of the learned distribution $p_{\overline{w}_r}$. As we also did for the NBC, we do this by perturbing their local parameters. Previous work on calculating the robustness of predictions of GeFs did this by perturbing the weights of the sum nodes using ε-contamination (Correia et al., 2020b). We adopt a similar approach but generalize it to allow for arbitrary local perturbations. Note that, in contrast to the NBC, we won't be perturbing any probability distributions over classes or features directly here, but only indirectly by perturbing the weights of the sum nodes.

Given a sum node v in a GeDT or GeF with weights w_v, we consider a perturbation $\mathcal{W}_v^\delta$ of w_v as defined in Definition 1. In this work, we choose not to perturb the weights of the root node r of a GeF to keep the importance of each GeDT equal. This means that we choose $\mathcal{W}_r^\delta = \{w_r\}$ for all δ. When we perturb the weights of all or multiple nodes in a GeF rooted at r, then we are essentially perturbing $\overline{w}_r$. The collection containing all possible combinations of perturbations of the weights $\overline{w}_r$ of the GeF rooted at r is given by $\overline{\mathcal{W}}_r^\delta := \{w_r\} \times \bigtimes_{v \in \mathrm{sde}(r)} \mathcal{W}_v^\delta$. Now, since each $\overline{w}'_r \in \overline{\mathcal{W}}_r^\delta$ has a corresponding joint distribution $p_{\overline{w}'_r}$, each perturbation $\overline{\mathcal{W}}_r^\delta$ corresponds to a perturbation $\mathcal{P}_{\overline{\mathcal{W}}_r^\delta} := \left\{p_{\overline{w}'_r} : \overline{w}'_r \in \overline{\mathcal{W}}_r^\delta\right\}$ of $p_{\overline{w}_r}$. The parameter δ again indicates that this perturbation is part of a parametrized perturbation, enabling us to associate a robustness value with each instance.

By Theorem 2, to calculate the robustness of a GeF, we need to calculate the lower expectation over a given perturbation of its joint probability mass function. We show that this can be done efficiently by using the structure of GeFs. To that end, we first define a (recursive) function of the nodes in a GeF and prove that the value of this function at the root node of a GeF is enough to check the robustness.

Theorem 7 *Let $h_{p_{\overline{w}_r}}$ be a GeF rooted at node r with mass function $p_{\overline{w}_r} \in \Sigma_{\mathcal{C}\times\boldsymbol{\mathcal{F}}}$ corresponding to the weights $\overline{w}_r$. Let $\hat{c}$ be the prediction according to $h_{p_{\overline{w}_r}}$ for the set of features $\boldsymbol{f}$. Consider local perturbations $\mathcal{W}_v^\delta$ of the weights w_v of all sum nodes $v \neq r$ and let $\mathcal{P}_{\overline{\mathcal{W}}_r^\delta}$ be the corresponding perturbation of $p_{\overline{w}_r}$. Then $\hat{c}$ is robust w.r.t. $\mathcal{P}_{\overline{\mathcal{W}}_r^\delta}$ if and only if*

$$\min_{c \in \mathcal{C}\setminus\{\hat{c}\}} \underline{V}(r) > 0, \tag{12}$$

where the function $\underline{V}$ is defined recursively by

$$\underline{V}(v) := \begin{cases} (p_{l,C}(\hat{c}) - p_{l,C}(c)) \prod_{i=1}^N p_{l,F_i}(f_i) & \text{if } v \text{ is a leaf node } l, \\ \frac{1}{|\mathrm{ch}(r)|} \sum_{u \in \mathrm{ch}(r)} \underline{V}(u) & \text{if } v \text{ is the root node } r, \\ \min_{w'_v \in \mathcal{W}_v^\delta} \sum_{u \in \mathrm{ch}(v)} w'_{v,u} \underline{V}(u) & \text{if } v \text{ is any other (sum) node.} \end{cases} \tag{13}$$

Proof Since $\mathcal{W}_r = \{w_r\}$ in our case and the product that appears for the case $v = \ell$ is equal to $E_{p_l}((\mathbb{I}_{\hat{c}} - \mathbb{I}_c)\mathbb{I}_{\boldsymbol{f}})$, it follows from Theorem 8 in Appendix A that $\underline{V}(r) = \underline{E}_{\mathcal{P}_{\overline{\mathcal{W}}_r^\delta}}((\mathbb{I}_{\hat{c}} - \mathbb{I}_c)\mathbb{I}_{\boldsymbol{f}})$. The result now follows directly from Theorem 5. ■

Consequently, checking whether a prediction of a GeF is robust w.r.t. a perturbation can be done by recursively calculating the function $\underline{V}(r)$ at the root node of the GeF for each of the classes in $\mathcal{C}\setminus\{\hat{c}\}$, and finding the robustness value of an instance amounts to finding the smallest value of δ for which this is not the case.

6. Local Perturbations

In this section, we discuss several examples of how a mass function can be perturbed, and give the formulas needed to calculate the robustness of predictions w.r.t. these perturbations. Recall that the local perturbations that we consider in the paper are, in the case of the NBC, $\mathcal{P}^\delta_C$ of p_C and $\mathcal{P}^\delta_{F_i|c}$ of $p_{F_i|c}$ for all $c \in \mathcal{C}$ all $i \in \{1, \ldots, N\}$; and in the case of GeFs, $\mathcal{W}^\delta_v$ of the weight vector w_v for each sum node $v \neq r$. For the sake of ease of notation, we generically denote any of these as a perturbation $\mathcal{Q}^\delta$ of a mass function q on a possibility space Ω. Then for the three types of perturbations that we consider in this work, namely ε-contamination, Total Variation (TV) distance balls and χ^2-divergence balls (Gibbs and Su, 2002), the perturbations are defined as follows:

- **ε-contamination:** $\mathcal{Q}^\delta_\varepsilon = \{(1-\delta)q + \delta q' : q' \in \Sigma_\Omega\}$, with $\delta \in [0,1]$;
- **TV distance:** $\mathcal{Q}^\delta_{\text{TV}} = \{q' \in \Sigma_\Omega : \frac{1}{2}\sum_{\omega\in\Omega} |q(\omega) - q'(\omega)| \leq \delta\}$, with $\delta \in [0,1]$;
- **χ^2-divergence:** $\mathcal{Q}^\delta_{\chi^2} = \{q' \in \Sigma_\Omega : \sum_{\omega\in\Omega} \frac{(q'(\omega)-q(\omega))^2}{q(\omega)} \leq \delta\}$, with $\delta \geq 0$.

For each of the models, different optimization problems need to be solved to calculate the robustness of their predictions. We now list all of these optimization problems and give for each of the three types of perturbations the formulas needed to solve these problems. Even though there has already been work on calculating the robustness of predictions of NBCs and GeFs w.r.t. ε-contamination, we nevertheless give the formulas to make the paper self-contained and to make it easier to compare with the other perturbations. For previous work on the robustness of NBCs w.r.t. ε-contamination, we refer to the work of Detavernier and De Bock (2025) and for GeFs to the work of Correia et al. (2020b). The formulas for the TV distance and χ^2-divergence balls are particular cases of more general formulas in recent work of De Bock (2026), which simplify in our particular context.

Firstly, we know from Theorem 6 that for the NBC we need to calculate the lower and upper probabilities of $p_{F_i|\hat{c}}(f_i)$ w.r.t. the perturbations $\mathcal{P}^\delta_{F_i|c}$, for all $c \in \mathcal{C}$ and $i \in \{1, \ldots, N\}$. These are particular instances of the task of computing the lower and upper probability of an outcome over a perturbation $\mathcal{Q}^\delta$ of a mass function q on Ω, which we denote by $\underline{q}_{\mathcal{Q}^\delta}(\omega) = \min_{q'\in\mathcal{Q}^\delta} q'(\omega)$ and $\overline{q}_{\mathcal{Q}^\delta}(\omega) = \max_{q'\in\mathcal{Q}^\delta} q'(\omega)$ for any $\omega \in \Omega$. For our three perturbations the formulas for these bounds are given in Table 1.

Table 1: The formulas for $\underline{q}_{\mathcal{Q}^\delta}(\omega)$ (left) and $\overline{q}_{\mathcal{Q}^\delta}(\omega)$ (right) for our three perturbations.

ε	$(1-\delta)q(\omega)$	$(1-\delta)q(\omega) + \delta$
TV	$\max\{0, q(\omega) - \delta\}$	$\min\{1, q(\omega) + \delta\}$
χ^2	$\max\{0, q(\omega) - \sqrt{\delta q(\omega)(1-q(\omega))}\}$	$\min\{1, q(\omega) + \sqrt{\delta q(\omega)(1-q(\omega))}\}$

Next, still for the NBCs, by Theorem 6 we also need to calculate the lower expectation of $\mathbb{I}_{\hat{c}}\alpha - \mathbb{I}_c\beta$ w.r.t. the local perturbations $\mathcal{P}^\delta_C$ of p_C, for all $c \in \mathcal{C}\backslash\{\hat{c}\}$, where $\hat{c}$ is the predicted class by the NBC and with $\alpha, \beta \geq 0$. In the equation in Theorem 6, α and β represent products of lower and upper probabilities that do not depend on the perturbation $\mathcal{P}^\delta_C$, which is why we write them as constants here. The formulas for $\underline{E}_{\mathcal{P}^\delta}(\mathbb{I}_{\hat{c}}\alpha - \mathbb{I}_c\beta)$ for

our three perturbations are given in Table 2, where μ and σ^2 are the expected value and variance of $\mathbb{I}_{\hat{c}}\alpha - \mathbb{I}_c\beta$ under p_C, and μ_2 and σ_2^2 are the expected value and variance of the restriction of this function to $\mathcal{C}\backslash\{\hat{c}\}$ under the mass function p_C^* on $\mathcal{C}\backslash\{\hat{c}\}$ defined by $p_C^*(c) = \frac{p_C(c)}{1-p_C(\hat{c})}$ for all $c \in \mathcal{C}\backslash\{\hat{c}\}$.

Table 2: The formulas for $\underline{E}_{\mathcal{P}_C^\delta}(\mathbb{I}_{\hat{c}}\alpha - \mathbb{I}_c\beta)$ for our three perturbations.

ε	$(1-\delta)(\alpha p_C(\hat{c}) - \beta p_C(c)) - \delta\beta$
TV	$\alpha \max\{0, p_C(\hat{c}) - \delta\} - \beta \min\{1, p_C(c) + \delta\}$
χ^2	$\begin{cases} \mu - \sigma\sqrt{\delta} & \text{if } \delta < \frac{\sigma^2}{(\alpha-\mu)^2} \\ -\beta & \text{if } \delta \geq \frac{1}{1-p_C(\hat{c})}\left(\frac{\sigma_2^2}{\mu_2^2} + p_C(\hat{c})\right) \\ \mu_2 - \sigma_2\sqrt{\delta(1-p_C(\hat{c})) - p_C(\hat{c})} & \text{otherwise} \end{cases}$

Lastly, for the GeFs, given a perturbation $\mathcal{W}_v^\delta$ of the weights of a sum node v, we need to compute the minimum weighted average of the values of its children, where the value of a child $u \in \mathrm{ch}(v)$ is given by the function $\underline{V}(u)$. In the GeFs we consider, this average always consists of only two terms, since all sum nodes of which we perturb the weights have only two children. If we let u_1 and u_2 be the two children of v, such that, without loss of generality, $\underline{V}(u_1) \leq \underline{V}(u_2)$, then this minimum reduces to:

$$\min_{w'_v \in \mathcal{W}_v^\delta} w'_{v,u_1}\underline{V}(u_1) + w'_{v,u_2}\underline{V}(u_2) = \underline{V}(u_1) + (\underline{V}(u_2) - \underline{V}(u_1))\underline{w}_{v,u_2}, \tag{14}$$

where $\underline{w}_{v,u_2} = \min_{w'_v \in \mathcal{W}_v^\delta} w'_{v,u_2}$ is the minimum of w_{v,u_2} w.r.t. the perturbation $\mathcal{W}_v^\delta$. Since the weight vector of a node is a mass function, finding this is equivalent to calculating a lower probability. We can thus use the formulas in the left column of Table 1 to calculate this minimum for each of our three perturbations.

7. Experiments

We present some preliminary experiments[3] on 11 datasets to illustrate the potential of the robustness values defined in this paper. In the following, we denote the robustness values for ε-contamination, TV distance balls and χ^2-divergence balls as r_ε, r_{TV} and r_{χ^2} respectively. For details on the experimental setup, training procedure and the datasets we used, we refer to Appendix B. We also compare our three robustness values with another approach that is commonly used to assess the trustworthiness of predictions, namely Uncertainty Quantification (UQ). Details on the six UQ metrics we compare with (u_{marg}, $u_{\max}$, u_{H}, u_{t}, u_{a}, u_{e}) are given in Appendix C.

To evaluate the performance of our approach, we check how the accuracy of the model changes as we continuously reject the predictions with the lowest robustness (or highest uncertainty); the idea being that if robustness is a good indicator of the trustworthiness of a prediction, then the accuracy on the remaining predictions should increase as we reject more and more instances. A curve showing for every percentage of rejected instances the

3. The source code is available at https://github.com/addtaver/RobustnessQuantification

accuracy on the remaining predictions, is called an *accuracy-rejection curve* (ARC); Figure 2 shows an example, where each line represents an average over 10 different runs. On this

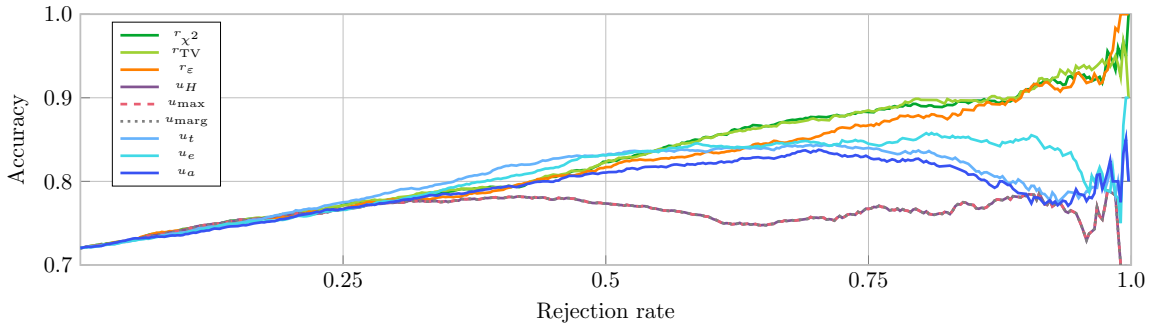


Figure 2: Accuracy-rejection curve for dataset D5 for GeFs.

particular dataset, the three robustness measures seem to perform best overall since they keep increasing until the very end, more so than the other metrics shown. However, if we would focus on specific Rejection Rates (RRs), then other metrics sometimes perform better. Take for example u_{t}, which is clearly the best measure for RRs of around 40%.

Focussing on specific RRs can be very relevant in practice if you only want to let the model make a prediction for the most trustworthy X% of instances. To evaluate the performance of the robustness values in such use cases, we focus on specific RRs in the ARC and check what metric of trustworthiness performs best for each of these RRs. The results are summarized in Table 3 and Table 4, where for each specific RR and trustworthiness metric the mean value of the accuracy at that RR over all datasets is shown, with the number of wins (W) for that metric at that RR next to it. A win is defined as having the highest accuracy at that RR for a given dataset, and ties are solved by giving a win to all metrics that are tied for the highest accuracy. We also included in the last row of these tables the average area under the ARC (AU-ARC) for each metric over all datasets, which instead of focussing on specific RRs, gives an overall indication of the performance of a metric as an indicator of trustworthiness. The AU-ARC can also help us to summarize and compare ARCs in a more quantitative way, since comparing ARCs by eye can be very difficult, especially when there are many of them.

Table 3: Each row, except the last one containing the AU-ARC, shows the mean accuracy on the remaining instances and the number of datasets for which each metric had the highest accuracy at that RR for the NBC.

RR(%)	r_{χ^2}		r_{TV}		r_ε		u_{t}		u_{a}		u_{e}		u_{m}		u_{H}		u_{c}	
	%	W	%	W	%	W	%	W	%	W	%	W	%	W	%	W	%	W
10%	82.22	1	82.00	0	82.13	**4**	**82.51**	3	82.39	2	81.92	1	82.24	0	82.26	0	82.26	1
25%	85.30	1	84.73	2	85.14	**4**	**85.41**	2	85.17	1	84.73	2	85.06	0	84.97	0	85.09	0
75%	**91.58**	4	90.48	1	91.22	**5**	91.28	3	91.00	3	91.03	1	91.10	3	90.91	3	91.11	3
90%	**93.14**	3	91.74	2	92.62	4	92.27	4	91.94	4	92.45	4	91.64	**5**	91.92	**5**	91.42	**5**
AU-ARC	**0.88401**		0.87679		0.88114		0.88249		0.88029		0.87720		0.87901		0.87848		0.87928	

If we focus on the two new robustness values in these tables, we see that r_{χ^2} seems to be the best performing metric for the NBC, especially for higher RRs, while r_{TV} seems to be the best performing metric for the GeF, especially for lower RRs. The robustness value r_ε

Table 4: Each row, except the last one containing the AU-ARC, shows the mean accuracy on the remaining instances and the number of datasets for which each metric had the highest accuracy at that RR for the GeF.

RR(%)	r_{χ^2}		r_{TV}		r_{ε}		u_{t}		u_{a}		u_{e}		u_{m}		u_{H}		u_{c}	
	%	W	%	W	%	W	%	W	%	W	%	W	%	W	%	W	%	W
10%	83.94	1	83.96	**4**	83.83	1	**84.06**	3	83.63	0	83.68	2	83.81	1	83.83	1	83.79	1
25%	86.44	1	**86.54**	3	86.21	1	**86.54**	**6**	85.68	1	85.68	2	85.25	1	85.25	1	85.27	1
75%	89.82	2	89.77	3	89.73	1	88.72	**5**	87.31	2	**89.87**	4	84.21	1	84.23	1	84.20	1
90%	90.36	2	90.52	2	90.23	3	87.82	4	86.77	2	**91.73**	**6**	81.44	1	81.44	1	81.47	1
AU-ARC	0.87952		**0.88003**		0.87810		0.87078		0.86149		0.87736		0.84156		0.84140		0.84160	

has similar AU-ARCs as the others, seems to do very well in terms of the number of wins for the NBC, and is just lacking behind the other two for the GeF. For the UQ measures, we see that u_{e} and u_{t} seem to perform very well, since they tend to have a high number of wins and the highest AU-ARCs for certain RRs. Overall, since some of the differences in performance are rather small, we prefer to refrain from drawing strong conclusions about which is the 'best' metric, especially so since we observe that the relative performance of the different metrics depends on the model, RR and performance metric We can conclude however that all three robustness values are competitive as indicators of trustworthiness. What makes this particularly interesting, is that the metrics u_{t}, u_{a} and u_{e} use an ensemble that requires retraining the classifier on bootstrap samples of the training data. This retraining can be computationally expensive though and is not always possible. This makes our robustness values, which do not require retraining, a promising alternative.

8. Conclusion and Future Work

The main contribution of this paper is that we generalized existing RQ approaches for NBCs and GeFs, which only considered ε-contamination, to a more general approach that allows for any type of perturbation of the local parameters of these models. To illustrate this, we applied our results to two specific types of perturbations, namely TV distance and χ^2-divergence balls, and provide the formulas needed to calculate the robustness of predictions w.r.t. these perturbations (and, for the sake of completeness, also for ε-contamination). Finally, our experiments demonstrate that our new robustness values w.r.t. TV distance and χ^2-divergence balls can be used as indicators for the trustworthiness of predictions.

Possible extensions of this work could be to consider other types of perturbations of the local parameters, or other types of models. Since the behavior of the robustness values seems to be quite different for different contexts and models, it would also be interesting to study what perturbations are best suited for which contexts and models. A start could be to study and compare our new methods in a context with noise or distribution shift, as previous work has shown that RQ is particularly well-suited in such contexts in comparison with UQ (Detavernier and De Bock, 2025, 2026).

Previous work (Detavernier and De Bock, 2026) also shows that RQ and UQ can be complementary, and can be usefully combined to obtain even better assessments of trustworthiness. It would therefore be interesting to employ our new RQ metrics in such combinations as well.

Acknowledgments

We would like to thank the anonymous reviewers for their time, kind words and helpful feedback. The work of both authors was partially supported by Ghent University's Special Research Fund, through Jasper De Bock's Basic Research Funding project entitled "Modelling Uncertainty with Imprecise Probabilities".

References

Thomas Augustin, Frank P.A. Coolen, Gert De Cooman, and Matthias C.M. Troffaes. *Introduction to imprecise probabilities*. John Wiley & Sons, 2014. doi: 10.1002/9781118763117.

Giorgio Corani and Marco Zaffalon. Learning reliable classifiers from small or incomplete data sets: The naive credal classifier 2. *Journal of Machine Learning Research*, 9(4), 2008.

Giorgio Corani, Joaquín Abellán, Andrés Masegosa, Serafin Moral, and Marco Zaffalon. *Classification*, chapter 10, pages 230–257. John Wiley & Sons, Ltd, 2014. ISBN 9781118763117. doi: 10.1002/9781118763117.ch10.

Alvaro Correia, Robert Peharz, and Cassio P. de Campos. Joints in random forests. In *Advances in Neural Information Processing Systems*, volume 33, pages 11404–11415. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/8396b14c5dff55d13eea57487bf8ed26-Paper.pdf.

Alvaro H. C. Correia, Robert Peharz, and Cassio P. de Campos. Towards robust classification with deep generative forests, 2020b.

Jasper De Bock. Computing lower expectations with respect to total variation distance and chi-squared divergence balls, 2026. arXiv:2605.30091.

Jasper De Bock, Cassio P. de Campos, and Alessandro Antonucci. Global sensitivity analysis for MAP inference in graphical models. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/0966289037ad9846c5e994be2a91bafa-Paper.pdf.

Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1184–1193. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/depeweg18a.html.

Denis Deratani Mauá, Diarmaid Conaty, Fabio Gagliardi Cozman, Katja Poppenhaeger, and Cassio Polpo de Campos. Robustifying sum-product networks. *International Journal of Approximate Reasoning*, 101:163–180, 2018. doi: 10.1016/j.ijar.2018.07.003.

Adrián Detavernier and Jasper De Bock. Robustness quantification: a new method for assessing the reliability of the predictions of a classifier. In *Proceedings of the Fourteenth International Symposium on Imprecise Probabilities: Theories and Applications*, volume 290 of *Proceedings of Machine Learning Research*, pages 126–136. PMLR, 15–18 Jul 2025. URL https://proceedings.mlr.press/v290/detavernier25a.html.

Adrián Detavernier and Jasper De Bock. Robustness quantification and uncertainty quantification: Comparing two methods for assessing the reliability of classifier predictions, 2026. arXiv:2603.22988.

Alison L. Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International Statistical Review / Revue Internationale de Statistique*, 70(3):419–435, 2002. URL http://www.jstor.org/stable/1403865.

Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems*, volume 35, pages 507–520. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0378c7692da36807bdec87ab043cdadc-Paper-Datasets_and_Benchmarks.pdf.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, 110(3):457–506, 2021. doi: 10.1007/s10994-021-05946-3.

Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The UCI Machine Learning Repository. URL https://archive.ics.uci.edu.

Aryan Mobiny, Pengyu Yuan, Supratik K. Moulik, Naveen Garg, Carol C. Wu, and Hien Van Nguyen. Dropconnect is effective in modeling uncertainty of bayesian deep networks. *Scientific Reports*, 11(1), March 2021. doi: 10.1038/s41598-021-84854-x.

Mohammad Hossein Shaker and Eyke Hüllermeier. *Aleatoric and Epistemic Uncertainty with Random Forests*, page 444–456. Springer International Publishing, 2020. ISBN 9783030445843. doi: 10.1007/978-3-030-44584-3_35.

Matthias C.M. Troffaes. Decision making under uncertainty using imprecise probabilities. *International Journal of Approximate Reasoning*, 45(1):17–29, May 2007. ISSN 0888-613X. doi: 10.1016/j.ijar.2006.06.001.

Marco Zaffalon. The naive credal classifier. *Journal of Statistical Planning and Inference*, 105(1):5–21, 2002. doi: 10.1016/S0378-3758(01)00201-4.

Appendix A. Computing Lower Expectations for GeFs

In the paper we consider perturbations $\mathcal{W}_v^\delta$ of the weights w_v of a sum node v in a GeF, where we implicitly assume that they are part of parametrized perturbations $\mathcal{W}_v^\Delta$, with δ taking values in Δ. For notational convenience, in the following result we will simply consider any perturbation $\mathcal{W}_v$ of the weights of a sum node v. For the sake of generality, we also consider

a general perturbation $\mathcal{W}_r$ of w_r for the root node r of the GeF, rather than consider the trivial case $\mathcal{W}_r = \{w_r\}$ as in the paper. We collect all possible combinations of perturbations of the weights of the sum nodes in the tree rooted at v in $\overline{\mathcal{W}}_v \coloneqq \mathcal{W}_v \times \bigtimes_{u \in \text{sde}(v)} \mathcal{W}_u$, and we denote the corresponding perturbation of the joint mass function $p_{\overline{w}_v}$ by $\mathcal{P}_{\overline{\mathcal{W}}_v}$.

Theorem 8 *Consider a GeF rooted at a node r with for each sum node v a weight vector w_v and a perturbation $\mathcal{W}_v$ of w_v. For any (not necessarily sum) node v, let $\mathcal{P}_{\overline{\mathcal{W}}_v}$ be the corresponding perturbation of $p_{\overline{w}_v}$. Then for any function $g : (\mathcal{C}, \boldsymbol{\mathcal{F}}) \to \mathbb{R}$:*

$$\underline{V}(v) = \underline{E}_{\mathcal{P}_{\overline{\mathcal{W}}_v}}(g), \tag{15}$$

where, the function $\underline{V}$ node is defined recursively by

$$\underline{V}(v) \coloneqq \begin{cases} E_{p_v}(g) & \text{if } v \text{ is a leaf node,} \\ \min_{w'_v \in \mathcal{W}_v} \sum_{u \in \text{ch}(v)} w'_{v,u} \underline{V}(u) & \text{if } v \text{ is a sum node.} \end{cases} \tag{16}$$

For our proof of this result, we took inspiration from the work of Deratani Mauá et al. (2018), who define a recursive function on the nodes of a sum-product network to calculate conditional lower expectations. Their result is on the one hand more general because it applies to sum-product networks that are more general than GeFs, but on the other hand less general because it only applies to specific inferences. Their focus is also on the complexity of computations, whereas our focus is on calculating the lower expectation.

Proof of Theorem 8 First, consider the case where v is a leaf node. In this case, the function $\underline{V}(v)$ is given by $\underline{V}(v) = E_{p_v}(g)$. Since v is a leaf node, there are no sum nodes in the tree rooted at v and thus no weights to perturb, so $\mathcal{P}_{\overline{\mathcal{W}}_v} = \{p_v\}$. Hence, the lower expectation of g over $\mathcal{P}_{\overline{\mathcal{W}}_v}$ is given by $\underline{E}_{\mathcal{P}_{\overline{\mathcal{W}}_v}}(g) = \min_{p \in \mathcal{P}_{\overline{\mathcal{W}}_v}} E_p(g) = E_{p_v}(g)$, as claimed.

For the case where v is a sum node, we prove the claim by induction on the structure of the tree rooted at v. Assuming that the claim holds for all children of v, we will show that it also holds for v. Since $\underline{V}(v) = \min_{w'_v \in \mathcal{W}_v} \sum_{u \in \text{ch}(v)} w'_{v,u} \underline{V}(u)$, it follows from the induction hypothesis, that

$$\underline{V}(v) = \min_{w'_v \in \mathcal{W}_v} \sum_{u \in \text{ch}(v)} w'_{v,u} \underline{E}_{\mathcal{P}_{\overline{\mathcal{W}}_u}}(g) = \min_{w'_v \in \mathcal{W}_v} \sum_{u \in \text{ch}(v)} w'_{v,u} \left(\min_{p \in \mathcal{P}_{\overline{\mathcal{W}}_u}} E_p(g) \right). \tag{17}$$

Since each set of weights $\overline{w}'_u \in \overline{\mathcal{W}}_u$ corresponds to an element $p'_{\overline{w}_u}$ of $\mathcal{P}_{\overline{\mathcal{W}}_u}$, each minimization inside the sum is equivalent to minimizing over the perturbed sets of weights. By applying this, and the definition of the expectation, we get

$$\underline{V}(v) = \min_{w'_v \in \mathcal{W}_v} \sum_{u \in \text{ch}(v)} w'_{v,u} \left(\min_{\overline{w}'_u \in \overline{\mathcal{W}}_u} \sum_{(c, \boldsymbol{f}) \in \mathcal{C} \times F} p_{\overline{w}'_u}(c, \boldsymbol{f}) g(c, \boldsymbol{f}) \right). \tag{18}$$

Since the weights of a sum node are non-negative, we can take for each of the children of v the minimum out of the sum:

$$\underline{V}(v) = \min_{w'_v \in \mathcal{W}_v} \min_{(u \in \mathrm{ch}(v))\overline{w}'_u \in \overline{\mathcal{W}}_u} \sum_{(c,\boldsymbol{f}) \in \mathcal{C} \times F} \left(\sum_{u \in \mathrm{ch}(v)} w'_{v,u} p_{\overline{w}'_v}(c, \boldsymbol{f}) \right) g(c, \boldsymbol{f}). \tag{19}$$

Since all perturbations of the weights are independent of each other, minimizing over all these perturbations separately is equivalent to minimizing over all perturbed sets of weights $\overline{w}'_v \in \overline{\mathcal{W}}_v$ in the tree rooted at v. Hence, we can rewrite the above as

$$\underline{V}(v) = \min_{\overline{w}'_v \in \overline{\mathcal{W}}_v} \sum_{(c,\boldsymbol{f}) \in \mathcal{C} \times F} \left(\sum_{u \in \mathrm{ch}(v)} w'_{v,u} p_{\overline{w}_u}(c, \boldsymbol{f}) \right) g(c, \boldsymbol{f}). \tag{20}$$

Finally, if we use the fact that $p_{w'_v}(c, \boldsymbol{f}) = \sum_{u \in \mathrm{ch}(v)} w'_{v,u} p_{\overline{w}'_u}(c, \boldsymbol{f})$, and reuse similar arguments as before, we can finally rewrite the above as

$$\underline{V}(v) = \min_{\overline{w}'_v \in \overline{\mathcal{W}}_v} \sum_{(c,\boldsymbol{f}) \in \mathcal{C} \times F} p_{\overline{w}'_v}(c, \boldsymbol{f}) g(c, \boldsymbol{f}) = \min_{p \in \mathcal{P}_{\overline{\mathcal{W}}_v}} E_p(g) = \underline{E}_{\mathcal{P}_{\overline{\mathcal{W}}_v}}(g), \tag{21}$$

as claimed. ■

Appendix B. Additional Details on the Experiments

The experiments we presented in Section 7 were conducted on several datasets from the UCI Machine Learning Repository (Kelly et al.). The datasets we used are shown in Table 5. Next we give some details on how we cleaned these datasets and adapted the tasks to fit our setting. First, we removed all continuous features from the datasets, because we focus on discrete features in this work. Then we removed all instances with missing values from the datasets. For some of the datasets, we performed more specific cleaning. For the Solar Flare datasets and the Student Performance datasets, we adapted the task to one that leans more toward standard classification. The former datasets are originally about predicting the number of solar flares that occur, and this for three types of flares; we made this classification task binary, with the aim to predict whether a solar flare occurs (of any type) or not. For the Student Performance datasets, the original task is to predict the grade of a student, but we adapted this to a binary classification task as well, where we predict whether the student passes or fails.

The training procedure for the models is as follows. Unless the dataset comes with a predefined split, we split the datasets into a training and test set, where the test set is 40% of the whole dataset, and the size of the test set is maxed out at 500 instances. The NBC is trained by first optimizing a smoothing parameter with 5-fold cross-validation on the training set and then training the model on the whole training set with this smoothing parameter. For the details on how the GeFs are trained, we refer to the original paper of Correia et al. (2020a), where we choose to combine 25 GeDTs and where the distributions of the leaf nodes are learned using the maximum likelihood principle. The whole training procedure, including the splitting of the datasets, is repeated 10 times to reduce the effect of randomness on the results. The ARCs are then calculated for each of the 10 runs and averaged out.

Table 5: Details of the datasets used in the experiments.

ID	Dataset	Size	$\lvert\mathcal{C}\rvert$	$\lvert\boldsymbol{\mathcal{F}}\rvert$
D1	Adult	3000	2	7
D2	Australian Credit Approval	690	2	8
D3	Bank Marketing	3000	2	9
D4	Breast Cancer Wisconsin	683	2	9
D5	German Credit Data	1000	2	13
D6	National Poll Healthy Aging	714	3	13
D7	Solar Flare (big)	1066	2	10
D8	Solar Flare (small)	323	2	10
D9	SPECT Heart	428	2	22
D10	Student Performance Math	649	2	29
D11	Student Performance Port	649	2	29

Appendix C. Uncertainty Quantification Metrics

Another approach to assess the trustworthiness of a prediction is to try to quantify the uncertainty of the prediction, where (unlike for robustness) a higher uncertainty corresponds to a less trustworthy prediction.

In our experiments, we compare our robustness values with six commonly used uncertainty quantification measures. The three straightforward measures are one minus the (conditional) predicted probability of the predicted class, denoted as u_{m}, the entropy of the predicted (conditional) distribution over the classes, denoted as u_{H}, and the margin of confidence being the difference between the highest and second highest predicted (conditional) probabilities, denoted as u_{c}. Besides these, we also use some more complex measures that are based on the information-theoretic decomposition of uncertainty as is explained in the work of Hüllermeier and Waegeman (2021). To estimate these, an ensemble of 10 models is trained on a bootstrap sample of the training set of the same size as the original training set (Depeweg et al., 2018; Mobiny et al., 2021; Shaker and Hüllermeier, 2020). The so-called the *total uncertainty* u_{t} is estimated as the entropy of the average predicted (conditional) distribution over all models, *aleatoric uncertainty* u_{a} is then estimated as the average entropy of the predicted (conditional) distributions over all models and the *epistemic uncertainty* u_{e} is estimated as the difference between the total and aleatoric uncertainty.

For these uncertainty measures, the corresponding ARCs in our experiments are obtained by first rejecting instances with the highest uncertainty.