Real-world-robustness of tree-based classifiers

Anonymous authors
Paper under double-blind review

Abstract

The concept of trustworthy AI has gained widespread attention lately. One of the aspects relevant to trustworthy AI is robustness of ML models. In this study, we show how to compute the recently introduced measure of real-world-robustness - a measure for robustness against naturally occurring distortions of input data - for tree-based classifiers. The original method for computing real-world-robustness works for all black box classifiers, but is only an approximation. Here we show how real-world-robustness, under the assumption that the natural distortions are given by multivariate normal distributions, can be exactly computed for tree-based classifiers.

1 Introduction

Robustness of machine learning models is a recently widely investigated topic. One extensively studied topic is adversarial robustness (Szegedy et al., 2014), which deals with small particular manipulations of the input to cause misclassifications. These systematic manipulations are called adversarial attacks, and several algorithms have been developed (Chen et al., 2019b; Pawelczyk et al., 2020; Sharma et al., 2020) to find the nearest counterfactual (Kment, 2006; Wachter et al., 2018; Pawelczyk et al., 2021) (closest point to the input according to a distance metric that leads to misclassification) of data samples in various scenarios. Especially the area of adversarial attacks on images is highly researched since it can cause a variety of safety concerns, e.g., in medical image processing and classification (Ma et al., 2021; Kaviani et al., 2022).

A definition of robustness that is different than adversarial robustness was introduced in Scher & Trügler (2022) and termed *real-world-robustness*. It describes the robustness of the predictions of a trained machine learning model against natural distortions, e.g., data-processing errors, noise or measurement errors in the input data, opposed to systematic manipulations as in adversarial attacks. The real-world-robustness $\mathbf{R}_\mu$ of the prediction f of an N-dimensional data sample μ with a distortion $p_\mu(\vec{x})$ given as a probability density function (PDF) is defined via a binary function f',

$$f'(\mu, p_\mu(\vec{x})) = \begin{cases} 0, & f(\mu + p_\mu(\vec{x})) = f(\mu) \\ 1, & f(\mu + p_\mu(\vec{x})) \neq f(\mu) \end{cases} \tag{1}$$

and an integral that determines the probability P for a different prediction compared to the data sample μ,

$$P(f(\mu + p_\mu(\vec{x})) \neq f(\mu)) = \int_{\mathbb{R}^N} f'(\mu, p_\mu(\vec{x}))\, dp_\mu(\vec{x}). \tag{2}$$

The real-world-robustness $\mathbf{R}_\mu$ of the data sample μ with distortion $p_\mu(\vec{x})$ is then computed by

$$\mathbf{R}_\mu = 1 - P.$$

In words, real-world-robustness is the probability that the prediction (classification) of an input sample does not change under the given uncertainty of the input sample. Scher & Trügler (2022) showed how real-world-robustness can approximately be computed for any black-box classifier with a Monte-Carlo based method, under the constraint that the input feature space is not too high dimensional. They additionally provide

a detailed discussion for the justification of this definition of real-world-robustness, and how it differs from adversarial robustness.

Another research direction deals with robust adversarial training of machine learning models. Qian et al. (2022) present a comprehensive survey on robust adversarial training by introducing the fundamentals and a general theoretical framework, and by summarising different training methodologies against various attack scenarios. Tan et al. (2022) introduce a training framework by adding an adversarial sample detection network to improve the classifier. In tree-based models, a training framework to learn robust trees against adversarial attacks has been developed by Chen et al. (2019a), and Ghosh et al. (2017) investigate the robustness of Decision Trees with symmetric label noise in the training data. Chen et al. (2019b) propose a robustness verification algorithm for tree-based models to find the minimal distortion in the input that leads to a misclassification.

In this paper, we show how to precisely compute real-world-robustness for tree-based classifiers (Decision Trees, Random Forests and XGBoosted classifiers), under the assumption that the uncertainty of the input test samples can be described by certain statistical distributions. This is possible because the decision boundaries of tree-based classifiers are explicitly given (in contrast to, e.g., neural network classifiers). The idea is to extract the decision rule of each decision node of a tree-based classifier to separate the input feature space into non-overlapping regions. We then determine the probability that a random data sample wrt. the given uncertainty, which is modelled as a probability distribution, around a test sample lies in a region that has the same label as the prediction of the test sample itself.

The paper is structured as follows. First, we describe how to compute real-world-robustness for single Decision Trees in Section 2. Then the approach is extended to Random Forest classifiers and XGBoosted classifiers in Section 3. In Section 4, we present experimental results and Section 5 concludes the paper.

2 Robustness of Decision Trees

At first we show how to compute real-world-robustness for a trained Decision Tree (DT) classifier (Quinlan, 1986) with a categorical target variable. We extract the decision rule from each decision node of a trained DT to split the input feature space into non-overlapping regions. For two-dimensional inputs, the regions are rectangles and for higher dimensions, the regions are hyperrectangles. For ease of description, we call the regions *boxes* (Chen et al., 2019b). To determine the robustness of the prediction of a data sample with uncertainty (e.g., noise or measurement errors), we classify the data sample with the trained DT and compute the probability that a random sample wrt. the given uncertainty is in a box that has the same label as the data sample. Taking the sum over the computed probabilities returns the robustness of the DT classification for that particular data sample.

2.1 Segmentation of the feature space

We have a trained DT without prior knowledge about the input features X_i. Each decision node in a DT is a decision rule of the form $X_i \leq \tau_{ij}$, where τ_{ij} marks the j^{th} decision rule of feature X_i. Note that τ_{ij} is unique for each j in a DT. We extract the decision rule from each decision node in the tree, add it to the decision rule set τ_i of the associated feature X_i and sort each τ_i in ascending order. The elements of τ_i split one dimension of the input feature space into non-overlapping segments and the individual elements of two sets τ_j and τ_k are orthogonal to each other. Using two successive elements of each decision rule set τ_i to split the input feature space creates one individual box. If a feature X_i is bounded, we expand τ_i to its minimum and/or maximum values, otherwise we expand τ_i to negative and positive infinity to cover the entire input feature space, i.e., if $\tau_i = \{60, 80, 100\}$, the expanded unbounded set is $\tau_i' = \{-\infty, 60, 80, 100, \infty\}$. This results in a total number of boxes n_b given by

$$n_b = \prod_{i}^{N}(|\tau_i'| - 1) = \prod_{i}^{N}(|\tau_i| + 1), \tag{3}$$

where N is the number of input features and $|\tau_i|$ is the number of decision rules for feature X_i.

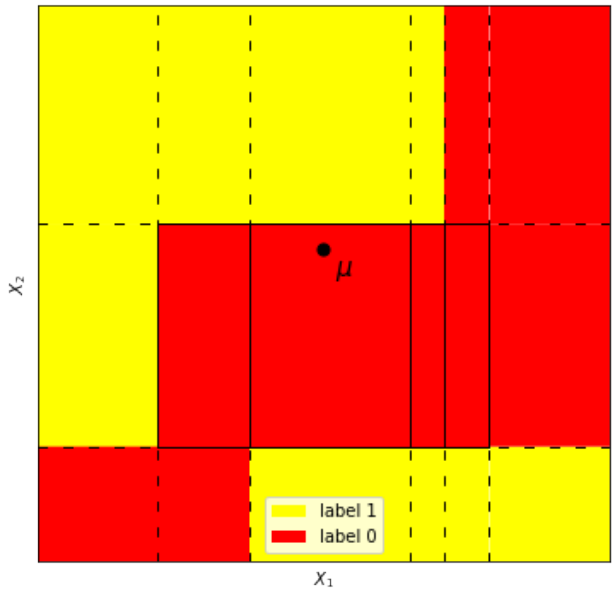


Figure 1: Illustration of boxes of a trained binary Decision Tree with two input features X_1, X_2 and a data sample μ.

2.2 Robustness

We use the created boxes to determine the robustness $\mathbf{R}_\mu$ of the prediction of an input data sample μ with associated uncertainty $p_\mu(\vec{x})$. We determine the predicted label of μ as well as the labels of all boxes by classifying their centre with the DT. To compute the robustness of μ, it suffices to only consider the boxes that have the same label as μ itself, denoted as $\mathbf{B}_\mu$. We determine the probability mass m_B that each box $B \in \mathbf{B}_\mu$ is covering wrt. the given uncertainty p_μ around the data sample μ. Taking the sum over the determined probability masses returns the robustness of the prediction of μ. Figure 1 illustrates the classified boxes (two labels) of a trained DT with two input features X_1, X_2 and a data sample μ.

In case the uncertainty distribution of the data sample μ is analytically tractable (e.g., a multivariate normal distribution with uncertainty Σ), exact solutions can be computed. We determine the probability mass m_B that each box $B \in \mathbf{B}_\mu$ is covering by integrating the probability density function $p_\mu(\vec{x})$ of the underlying uncertainty distribution between the lower and upper boundaries of each box. Taking the sum over all computed probability masses gives the robustness $\mathbf{R}_\mu$ of the prediction of the data sample μ,

$$\mathbf{R}_\mu = \sum_{B \in \mathbf{B}_\mu} \int \cdots \int_{B_{\text{low}}}^{B_{\text{upp}}} p_\mu(\vec{x})\, dx_1 dx_2 \ldots dx_N, \tag{4}$$

where B_{low} denotes the lower boundaries and B_{upp} denotes the upper boundaries of a box.

One-dimensional feature space For one-dimensional data samples μ with uncertainty given as a probability distribution, we can just integrate the PDF of the uncertainty probability distribution between the lower and upper boundary of each box $B \in \mathbf{B}_\mu$ if it is analytically tractable and take their sum to get the robustness of μ.

Input data with multivariate normal uncertainty For an N-dimensional data sample μ with multivariate normal uncertainty Σ, we integrate the multivariate normal probability density function

$$p_\mu(\vec{x}) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left(-\frac{1}{2}(\vec{x}-\mu)^T \Sigma^{-1} (\vec{x}-\mu)\right) \tag{5}$$

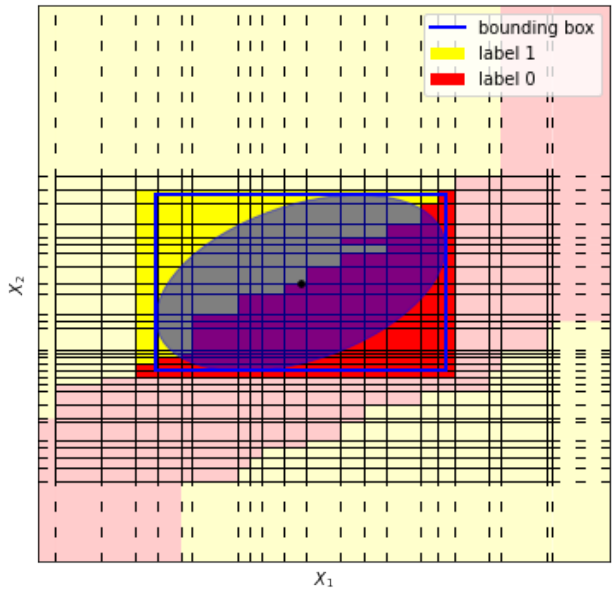


Figure 2: Boxes inside of or intersecting with the bounding box of the 99% confidence ellipse around a data sample with multivariate normal uncertainty, and more transparent boxes outside of the bounding box.

with (Genz, 1992) between the lower and upper boundaries of each box $B \in \mathbf{B}_\mu$ and take the sum over the probability masses, which returns the robustness of μ.

Input data with mixed uncertainty distributions For N-dimensional data samples with correlated features and the uncertainty in different dimensions given by different distributions, e.g., normal, exponential or lognormal, computing the robustness via analytical integration might not be applicable, since integrating the joint PDF might not be analytically tractable. In these cases, other integration techniques (e.g., numerical integration) need to be applied to approximate the robustness of the prediction of the data sample.

For an N-dimensional data sample with independent features, the robustness can be computed via analytical integration, if the PDF of the uncertainty distribution in each dimension is analytically tractable. We determine the covered probability mass of each box $B \in \mathbf{B}_\mu$ by taking the product of the covered probability masses in each dimension and then take the sum over the probability masses m_B per box.

2.2.1 Runtime Improvement

Computing the robustness of low-dimensional inputs and shallow trees is fast since the number of boxes is small. For high-dimensional inputs and deep trees, the number of boxes (see Equation 3) and simultaneously the runtime increases with each input dimension. One approach to decrease the runtime for the robustness computation of the prediction of a data sample μ with multivariate normal uncertainty Σ is to only consider the boxes that are inside of or intersect with the bounding box of, e.g., the 99% confidence hyperellipsoid around a data sample μ. Experiments have shown that the resulting robustness computations are much faster since the probability mass for less boxes needs to be computed, while the difference in the results is negligible. There is also a theoretical upper bound for the error of the resulting robustness (at max 1 percentage point). Another approach to speed up computations, which has not been tested, would be to compute the probability mass m_B of each box with the same label as μ in parallel.

Figure 2 represents a two-dimensional data sample μ with multivariate normal uncertainty, where the boxes that are outside of the 99% confidence ellipse are more transparent. Only using the boxes that are inside of or intersect with the 99% confidence ellipse to determine the robustness of μ speeds up computations, while the difference in the results is negligible (see Section 4).

3 Robustness of other tree-based methods

We now extend the approach to determine the robustness of a DT to more advanced tree-based methods. We look at Random Forests, which consist of multiple trees, and at XGboosted trees.

3.1 Random Forest

A Random Forest (RF) (Breiman, 2001) extends a Decision Tree model and consists of multiple DTs. Each tree is trained on a bootstrapped dataset with the same size as the original training dataset, i.e., a sample can be part of the training set for one tree multiple times, whereas other samples are not part of that training set. We extract the decision rule of each decision node in all individual trees, merge them per input feature X_i to form the decision rule set τ_i and create boxes to compute the robustness of the prediction of a data sample.

Analogous to the DT setup, we have a trained RF without prior knowledge about the input features X_i or the individual trees. For each tree in the RF, we extract the decision rule $X_i \leq \tau_{ij}$ of each decision node, add it to the decision rule set τ_i of the associated feature X_i and sort each τ_i in ascending order, as was done for DTs. Since a RF consists of multiple DTs, decision nodes in different trees can have the same decision rule $X_i \leq \tau_{ij}$, leading to duplicate entries in τ_i. We eliminate all but one of the duplicate entries τ_{ij}, such that each τ_{ij} is unique. To create boxes for robustness computations, we use the method described in Section 2.1 for DTs.

After creating the boxes, we compute the robustness of the prediction of a data sample μ in a RF, analogous to the approach for DTs described in Section 2.2. First we determine the label of each box by classifying its centre with the RF, not with the single trees. Then we classify μ with the RF to determine its label and take the sum over all probability masses m_B of the boxes $B \in \mathbf{B}_\mu$ (boxes with the same label as μ).

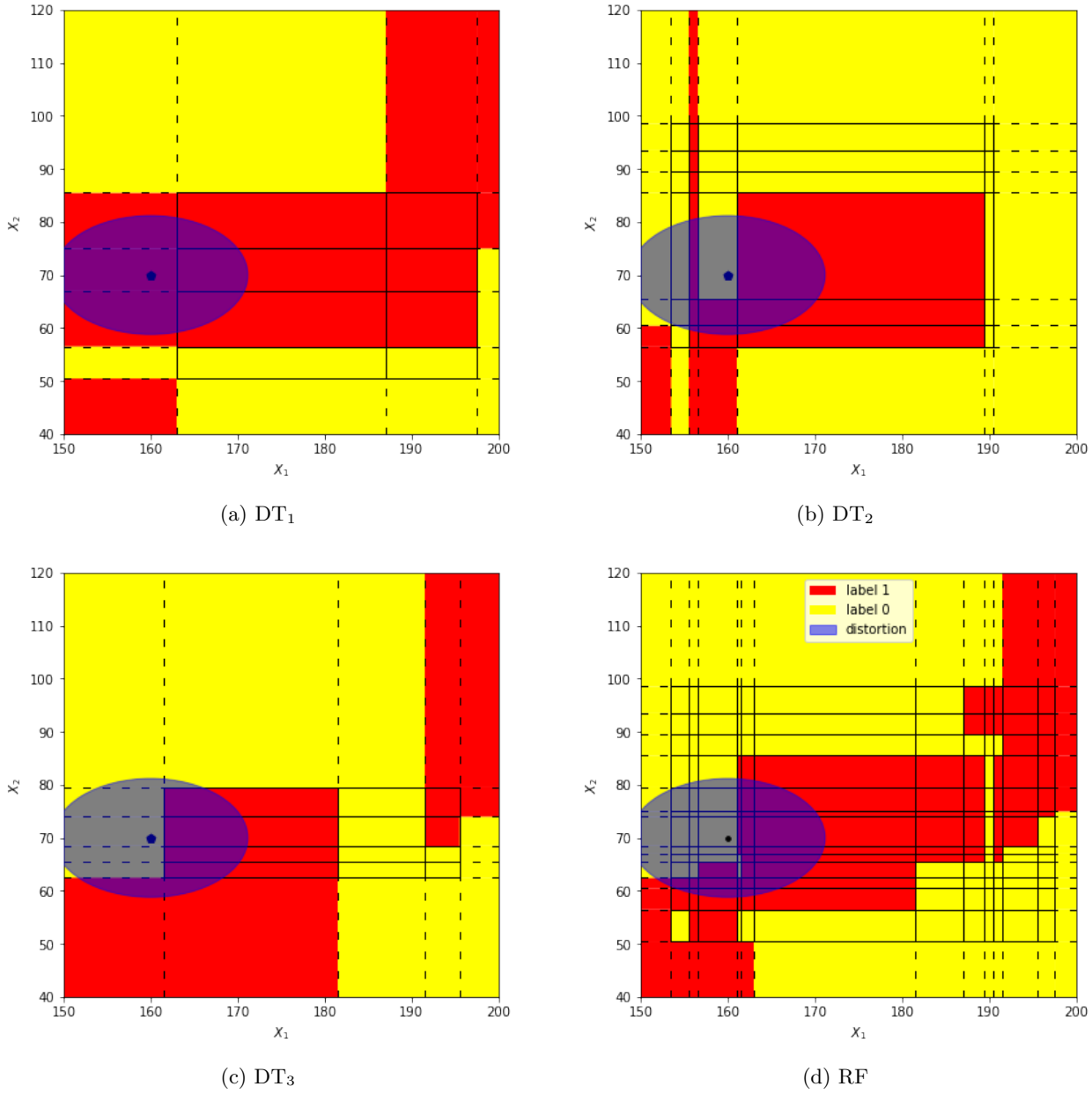
Computing the robustness in a RF is computationally more expensive than in a DT, since a RF consists of multiple trees, leading to more decision rules and a higher number of boxes (see Equation 3). In Section 2.2.1, we described an approach to approximate the robustness of the prediction of a data sample with multivariate normal uncertainty in a DT by only considering the boxes that are inside of or intersect with the bounding box of the 99% confidence hyperellipsoid. Experiments with various sizes of RFs (number of trees and depth of trees) have shown that computing the robustness of the prediction of a data sample is much faster when only considering these boxes, while the robustness results only differ marginally (see Secion 4).

Figure 3 contains the boxes of a trained RF (Figure 3d) with two input features X_1, X_2, two labels and three associated DTs (Figures 3a, 3b, 3c), as well as a data sample with multivariate normal uncertainty. We see that each individual DT is represented by different decision boundaries and different boxes. Overlaying the individual Figures of the DTs gives the representation for the trained RF (Figure 3d). We also observe that the data sample is classified differently in DT_1 (red label) compared to the other two DTs and the RF (yellow label).

At first glance, it would actually seem simpler to compute the real-world-robustness of a RF by first computing it individually for each DT of the forest, and then combining the results. This is, however, not possible, as averaging the robustness of the individual trees cannot account for interdependencies (e.g., even if tree A and tree B have the same robustness, the contributions from different parts of the feature space might be different, and averaging them would lead to false results).

3.2 XGBoosted Decision Tree model

Similar to a Random Forest, an XGBoosted Decision Tree model (Chen & Guestrin, 2016) also consists of multiple trees. Instead of training the trees from bootstrapped datasets, the individual trees build on each other. We again extract the decision rules of each decision node in all trees and combine them per feature X_i to form the associated decision rule set τ_i. We create boxes which are used to compute the robustness of the prediction of a data sample. Since trees in an XGBoosted Decision Tree model build on each other, there are less different decision rules than in a RF, when the number of trees and depth of the trees is the same. This leads to a smaller number of boxes and thus computations are faster.



(a) DT_1 (b) DT_2

(c) DT_3 (d) RF

Figure 3: Boxes of 3 Decision Trees with two-dimensional input (a) - (c) and the resulting combined boxes (d) of the associated binary Random Forest.

Table 1: Comparison of runtimes using all boxes and only the boxes that are within or intersect with the 99% confidence hyperellipsoid around a test sample for robustness computation

	All boxes	99% boxes
Test sample	Boxes - Runtime	Boxes - Runtime
1	19,118 - 488s	296 - 8s
2	24,616 - 589s	3,848 - 99s
3	32,564 - 502s	1,824 - 46s
4	39,636 - 697s	52 - 1s
5	13,356 - 274s	1,512 - 39s
6	32,564 - 518s	1,824 - 45s
7	13,356 - 241s	48 - 2s
8	48,128 - 927s	380 - 10s
9	32,564 - 468s	120 - 3s
10	5,736 - 153s	24 - 1s

4 Experimental results

In this Section we summarise experimental results for the computation of real-world-robustness for tree-based classifiers. Experiments were carried out on 1 core of an Intel(R) Xeon(R) 6248 CPU @ 2.50GHz processor with 256GB RAM. The repository with codes will be made available with the camera ready version.

4.1 Results with 99% confidence hyperellipsoid

In Section 2.2.1 we described a method to decrease the runtime of the robustness computation while only having negligible differences in the results. We trained a Decision Tree with a maximum depth of 4 on the Iris flower dataset with a train/test split of 90/10. The uncertainty of the 15 test samples is given as a multivariate normal distribution with Ones on the main diagonal and Zeros on the off-diagonals of the covariance matrix. We compare the robustness of the prediction of the test samples computed with all boxes to the results, where we only look at the boxes that are within or intersect with the 99% confidence hyperellipsoid around the test samples. In the experiments, we achieved R^2-scores exceeding 0.9999 in each test run, showing that it suffices to only consider the boxes that are within or intersect with the 99% confidence hyperellipsoid to compute the robustness of the prediction of the test samples.

4.2 Runtime Analysis

Considering only the boxes that are within or intersect with the 99% confidence hyperellipsoid around a test sample not only returns good results for the computation of the robustness of the prediction of a test sample, but is also much faster. With the Iris flower dataset, the runtime difference for the test samples was negligible since it only contains 4 input features. We therefore conducted experiments with the MNIST dataset (Deng, 2012). We resized the images from 28×28 to 5×5 pixels, normalised them, flattened them into a vector and used the 25-dimensional vector as input to train a Random Forest with 5 trees and a maximum depth of 3 per tree on the training set (60,000 training samples). To evaluate the runtime of the robustness computation, we determined the robustness of the first 10 images of the test set (10,000 test samples). The uncertainty is given as a multivariate normal distribution with 0.001 on the main diagonal elements and Zeros on the off-diagonal elements of the covariance matrix. The resulting runtimes and number of boxes for the robustness computation of the 10 test samples are listed in Table 1. We see that the number of boxes that are within or intersect with the 99% confidence hyperellipsoid is much smaller and that the runtimes are much shorter, compared to using all boxes. The difference in the results is negligible as we achieved R^2-scores exceeding 0.9999.

4.3 Comparison to approximate robustness computation

Scher & Trügler (2022) introduced the term real-world-robustness, but their method is based on random sampling and therefore only returns approximate results. With this comes the limitation that it only works

when the feature dimensionality is not too high. We compare the two methods (Random Sampling against 99% confidence hyperellipsoid around a test sample) by computing the robustness of the prediction of data samples on trained Decision Trees. The DTs have a depth between 3 and 7 and were trained on the MNIST dataset. We perform the same data preprocessing steps as for the runtime analysis, but resize the images to various sizes, ranging from 3×3 to 10×10 such that the DTs have input feature dimensions ranging from 9 to 100, and train them on the training set. The uncertainty is given as a multivariate normal distribution with 0.0001 on the main diagonal elements and Zeros on the off-diagonal elements of the covariance matrix. We compared the results of each setting for the first 10 test samples and observed that both methods return similar results for the robustness of the prediction of data samples, with negligible differences starting in the third decimal place. This indicates that the method of Scher & Trügler (2022) to compute the robustness still works well with an input feature dimension of 100, and more input features would be necessary to see a difference in the results. We also computed the robustness of data samples with higher values in the main diagonal of the multivariate uncertainty distribution (0.001, 0.01, 0.1 and 0.5) to cover more parts of the input feature space and compared the results, but again only observed minor differences.

5 Conclusion

In this paper, we presented a method to precisely determine the real-world-robustness (robustness against natural distortions in the input) of tree-based classifiers. We extract the decision rules from a trained classifier to separate the input feature space into non-overlapping regions, called boxes, and integrate the underlying probability distribution that models the uncertainty of a test sample between the lower and upper boundaries of each box to determine their covered probability mass. Taking the probability sum over the boxes that have the same label as the test sample returns the real-world-robustness of the prediction of the test sample. We presented this approach for Decision Trees in detail and discussed the extension to Random Forests and XGBoosted trees. The method gives a precise measure of the real-world-robustness of the prediction of individual data samples with tree-based classifiers.

One limitation of our approach is that the uncertainty distribution needs to be modelled as a probability distribution and that the PDF of the distribution must be analytically tractable to compute exact solutions. This is not always possible, especially when features are correlated and the uncertainty in different dimensions is best modelled by different distributions. In case the uncertainty distribution is more complex (e.g., because it is not given as a probability distribution, but by a stochastic function that models some process), we can in principle use numerical integration techniques to integrate the probability mass over the boxes for classifiers with explicit decision boundaries (such as tree-based models) and find approximate solutions. We carried out some initial experiments in that direction, solving Equation 4 with the *quadpy* package (Schlömer et al., 2021) for numerical integration. We observed however that the results with numerical integration techniques are often unstable and unreliable, especially when a data sample μ is in a large box, i.e., there is a big gap between the lower and upper boundaries in at least one dimension of the box, compared to cases that are actually analytically solvable. While it could be possible that with more carefully choosing types and parameters of the numerical routines the results could be better, this shows that simply using off-the-shelf integration routines is not a feasible option.

A second limitation of our approach is the high amount of data storage space needed. Equation 3 shows the number of boxes that are being created for a trained classifier. Classifiers with high input feature dimension and especially Random Forests quickly exceed the available storage space, such that the robustness cannot be computed anymore. With a more powerful machine, experiments on tree-based classifiers with high input feature dimension could be carried out and still return exact solutions.

The method presented in this paper is applicable because tree-based classifiers have the convenient property of having explicitly described decision boundaries that form hyperrectangles. Future research should be dedicated to extending the presented approach to more advanced classifiers with complicated decision boundaries, such as (nonlinear) Support Vector Machines and Neural Networks, and to find solutions for computing the *real-world-robustness* of high dimensional classifiers.

References

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL http://dx.doi.org/10.1023/A%3A1010933404324.

Hongge Chen, Huan Zhang, Duane Boning, and Cho-Jui Hsieh. Robust decision trees against adversarial examples. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1122–1131. PMLR, 09–15 Jun 2019a. URL https://proceedings.mlr.press/v97/chen19m.html.

Hongge Chen, Huan Zhang, Si Si, Yang Li, Duane Boning, and Cho-Jui Hsieh. Robustness verification of tree-based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019b. URL https://proceedings.neurips.cc/paper/2019/file/cd9508fdaa5c1390e9cc329001cf1459-Paper.pdf.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *Signal Processing Magazine, IEEE*, 29:141–142, 11 2012. doi: 10.1109/MSP.2012.2211477.

Alan Genz. Numerical computation of multivariate normal probabilities. *Journal of computational and graphical statistics*, 1(2):141–149, 1992.

Aritra Ghosh, Naresh Manwani, and P. S. Sastry. On the robustness of decision tree learning under label noise. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 685–697, Cham, 2017. Springer International Publishing. ISBN 978-3-319-57454-7.

Sara Kaviani, Ki Jin Han, and Insoo Sohn. Adversarial attacks and defenses on ai in medical imaging informatics: A survey. *Expert Systems with Applications*, 198:116815, 2022. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2022.116815. URL https://www.sciencedirect.com/science/article/pii/S095741742200272X.

Boris Kment. Counterfactuals and Explanation. *Mind*, 115(458):261–310, 04 2006. ISSN 0026-4423. doi: 10.1093/mind/fzl261. URL https://doi.org/10.1093/mind/fzl261.

Xingjun Ma, Yuhao Niu, Lin Gu, Yisen Wang, Yitian Zhao, James Bailey, and Feng Lu. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, 110:107332, 2021. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107332. URL https://www.sciencedirect.com/science/article/pii/S0031320320301357.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. *Proceedings of The Web Conference 2020*, 2020.

Martin Pawelczyk, Sascha Bielawski, Jan van den Heuvel, Tobias Richter, and Gjergji Kasneci. Carla: A python library to benchmark algorithmic recourse and counterfactual explanation algorithms. *ArXiv*, abs/2108.00783, 2021.

Zhuang Qian, Kaizhu Huang, Qiu-Feng Wang, and Xu-Yao Zhang. A survey of robust adversarial training in pattern recognition: Fundamental, theory, and methodologies. *Pattern Recognition*, 131:108889, 2022. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2022.108889. URL https://www.sciencedirect.com/science/article/pii/S0031320322003703.

J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, mar 1986. ISSN 0885-6125. doi: 10.1023/A:1022643204877. URL https://doi.org/10.1023/A:1022643204877.

Sebastian Scher and Andreas Trügler. Robustness of machine learning models beyond adversarial attacks, 2022. URL https://arxiv.org/abs/2204.10046.

Nico Schlömer, Nick Papior, Darius Arnold, Jan Blechta, and Rasmus Zetter. nschloe/quadpy: None, August 2021. URL https://doi.org/10.5281/zenodo.5218525.

Shubham Sharma, Jette Henderson, and Joydeep Ghosh. *CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-Box Models*, pp. 166–172. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450371100. URL https://doi.org/10.1145/3375627.3375812.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, D. Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.

Xiao Tan, Jingbo Gao, and Ruolin Li. A simple structure for building a robust model, 2022. URL https://arxiv.org/abs/2204.11596.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard journal of law and technology*, 31:841–887, 04 2018. doi: 10.2139/ssrn.3063289.