The Effect of Bootstrapping Deep Ensembles on Explained Predictive Uncertainty

–

Bachelor Thesis

–

Institute of Computer Science, Freie Universität Berlin

Jan-Niklas Klein

[redacted]

Supervisor and First Reviewer: Prof. Dr. Grégoire Montavon
Second Reviewer: Prof. Dr. Wojciech Samek (TU Berlin)

September 5, 2024

Abstract

This bachelor thesis investigates the impact of bootstrapping on deep ensembles in the context of explained predictive uncertainty in machine learning models. As the need for not just accurate but also interpretable and reliable predictions grows, understanding and quantifying predictive uncertainty becomes essential. Deep ensembles are well-known to deliver both accurate predictions and reliable uncertainty estimates. However, they do not account for some sources of uncertainty due to the finite nature of the dataset samples. This work explores how introducing sampling variability through bootstrapping affects uncertainty estimates in deep ensembles and especially the explanations of predictive uncertainty. In this, it focuses on trustworthiness of the estimates and the reliability of uncertainty explanations, particularly by looking at second-order effects. Through an experimental analysis, this thesis tests hypotheses regarding the influence of naïve and parametric bootstraps on model diversity, calibration, and the sensitivity to spurious correlations in experiments on explained predictive uncertainty. This thesis will show the potential usefulness of bootstrapping deep ensembles in the context of explaining predictive uncertainty. It will highlight that bootstrapping deep ensembles increases epistemic uncertainty estimates and improves metrics such as negative log-likelihood and calibration. Furthermore, this thesis will demonstrate that in certain scenarios, bootstrapping is able to detect relevant noisy features in second-order explanations better than non-bootstrapped deep ensembles.

Contents

1 Introduction

As machine learning is applied in nearly all possible domains and applications, the focus has shifted away from models solely making the most accurate predictions. The assessment of a good prediction depends on several different factors and is more complexly faceted. It has been becoming increasingly more important to be able to understand *why* a model predicts as it does, and to unlock valuable insights into model behaviors and the underlying datasets. Particularly in more critical applications, often epitomized as scenarios such as medical diagnosis or decision-making in autonomous systems, it is of vital importance to understand the reliability of the predictions [1]. Thus, one needs a good measure of predictive uncertainty. In the event of an uncertain prediction beyond a specific threshold, one could then redirect a sample to a human supervisor, e.g. a doctor in case of medical image classification. Ideally, the system should then be able to point out the dubious areas in the picture, or ambiguous features in the data point.

It is therefore not only important to be able to detect and quantify uncertainty, but also to be able to explain uncertainty in a precise and easily interpretable way – particularly because often it might not be caused by a single feature, but a correlation of two or multiple at the same time. The field of explainable AI has been developed to be able to do just that. Neural networks have long been considered "black box" models, due to a lack of methods for tracing back explanations and understanding the reasoning behind their predictions. As a result, explainable AI has seen significant growth in recent years, and various ways of shedding light into the then "black box" models have been developed [2]. In this effort, several different explanation techniques have been introduced, which are especially being used in common classification and regression tasks [3]. In this context, a particularly interesting property to know about a model is its level of confidence in its predictions, which we can also describe as an estimation of its predictive uncertainty. Knowing this helps us to better determine when we can trust a supposedly accurate prediction, and when we can not.

However, obtaining reliable estimates of predictive uncertainty is not straightforward, especially for individual models. Thus, there are several approaches of quantifying uncertainty in practice. Because of their reliability and practicability, deep ensembles – constellations of several independently trained models – are the state-of-the-art-technique for uncertainty estimates in widely used non-Bayesian model architectures. Despite their success, deep ensembles fail to account for some sources of uncertainty as they do not capture variability in the data set sampling. To address this limitation, this thesis investigates *bootstrapping* as a method to introduce such variability by training the ensemble models on slightly different training sets, and its effects on explained predictive uncertainty. While some studies have explored bootstrapping in deep ensembles [4] [5], none have directly examined its qualitative impact on explaining predictive uncertainty.

This work aims to improve the understanding of the impact of bootstrapping deep ensembles by focusing on its effects on uncertainty explanations. In this, it is primarily concentrating on the *naïve bootstrap* (bagging) [6], which introduces variability by training on random subsets (with replacement) for each model in the ensemble. Several hypotheses are proposed in this thesis. The first hypothesis is that bootstrapping increases the estimates for predictive uncertainty, as it introduces a wider model diversity into the ensemble. By doing so, it might be able to reflect the true predictive uncertainty more realistically. Secondly, the more diverse prediction range may lead to improved measures of trustworthiness. These effects could vary on different factors such as the dataset size and complexity. Lastly, a bootstrapped deep ensemble may be more effective in identifying relevant sources or features causing the uncertainty. In this, it may be less prone to random correlations between features and focus more on truly relevant interactions.

These hypotheses will be examined in an experimental analysis. For this analysis, different explanation techniques will be utilized, including a novel method for explaining relevances of feature correlations (*second-order effects*) introduced by Bley et al. (2024) [1]. An exploratory research over different ensemble configurations will be conducted, mainly consisting of two experiments. The first experiment will be a benchmark on several regression datasets, aiming to provide comparable results to other studies on this topic. Then, a second experiment will explore the hypotheses in a specifically designed synthetic regression environment. The results of these experiments will demonstrate the usefulness of naïve bootstrap in deep ensembles in certain scenarios, in line with the proposed hypotheses.

2 Background

2.1 Predictive Uncertainty Framework

First of all, it is vital to understand the nature of predictive uncertainty in depth. In machine learning research, predictive uncertainty and its sources are often categorized into two different general types. These two types are typically being referred to as *aleatoric* and *epistemic* uncertainty [7] [8], as seen in Figure 1.

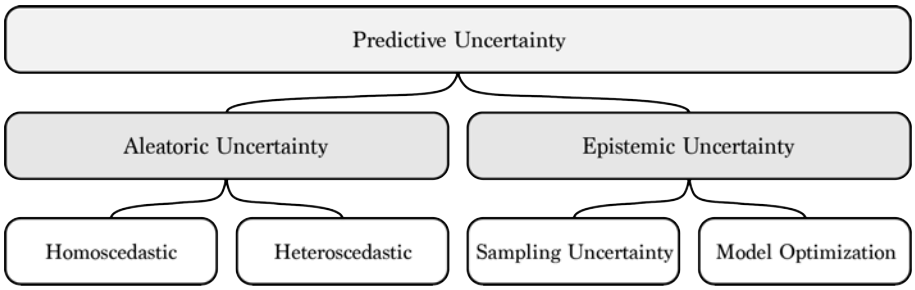

Figure 1: Types of predictive uncertainty.

Aleatoric uncertainty (AU) describes the intrinsic randomness and variability of the system that is being modeled. Thus, it can not be reduced or eliminated by gathering more data points or refining the model optimization procedures [7]. This type of uncertainty can be caused by things like measurement noise in the data collection, stochastic events, or other randomness in the observed processes. It is therefore sometimes also being called "data uncertainty", as it is an inherent property of the data distribution and not of the model itself [9]. Aleatoric uncertainty can also be further refined into the subcategories of *heteroscedastic* and *homoscedastic* noise, which describes whether the variability in the data changes (heteroscedastic) or stays the same (homoscedastic) across different inputs to the model [10]. However, in most scenarios it is reasonable to assume heteroscedastic noise [11].

In contrast, *epistemic uncertainty* (EU) is very much reducible and describes the uncertainty that arises from a lack of knowledge or incomplete modeling of the ground truth[1]. For example, this can be due to the finite number of samples drawn from the unknown true distribution or due to tunable or random parameters in the model optimization procedure [7]. One can therefore further decompose epistemic uncertainty into different sources [5]. We will refer to the *sampling uncertainty*, when we talk about sources related to the finite number of data points, and *model optimization* for sources inherent to the model's structure or parameter space. Both types of epistemic sources can be reduced, as one can gain a better understanding of the overall system by either gathering more data or improving the model. Finally, aleatoric and epistemic uncertainty can be represented as their sum, Equation 1 [9].

$$\textit{Predictive Uncertainty} = \textit{Aleatoric Uncertainty} + \textit{Epistemic Uncertainty} \tag{1}$$

Let us set a couple of assumptions to properly formalize and define predictive uncertainty. First of all, we will assume a regression[2] setting throughout this work, as this simplifies the formalization and the experimental process later on. Let $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a data set that we assume to be *independent* and *identically distributed* (IID). D is consisting of tuples (x_n, y_n), where $x \in \mathbb{R}^d$ (*data points* or *inputs*) and $y \in \mathbb{R}$ (*observations, inputs* or *targets*). In this setting, $f(x)$ is the true, unknown regression function that describes the true relationship between the data points and the targets. We can express this regression relationship now as a combination of a deterministic part, the true function, and a stochastic part, the noise term ϵ_i, as in Equation 2. We generally assume ϵ_i to be normally distributed.[3]

$$y_i = f(x_i) + \epsilon_i \tag{2}$$

When we attempt to model this regression setting with a neural network, we are essentially trying to approximate $f(x)$ with $\hat{f}(x)$. This approximation introduces epistemic uncertainty, as we are

[1] The term *ground truth* refers to the unknown, true relationship of data points x and observations y.

[2] In regression, the goal is to model the relationship between (multiple) input variables and continuous output variables.

[3] This is a general assumption made by most papers on this topic [5].

confronted with a lack of knowledge about the proximity of our model to the true function. This includes the epistemic uncertainty due to the finite number of (randomly sampled) data points from the true distribution and the limitations of the model hypothesis or parameter space.

$$\hat{f}(x_i) = f(x_i) + \epsilon_{epistemic} \tag{3}$$

We can therefore approximate Equation 2 with:

$$\hat{y}_i = \hat{f}(x_i) + \epsilon_i \tag{4}$$

The noise term ϵ_i on the other hand relates to aleatoric uncertainty, which is inherent in the observations and therefore cannot be reduced by collecting more data. If ϵ_i is depending on x_i, we are dealing with heteroscedastic noise, which means the variance of the noise varies with the input:

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2(x_i))$$

Contrary, if the noise is independent of x_i, we call it homoscedastic noise, where the variance stays constant across the inputs:

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

All in all, this gives us a solid theoretical framework for predictive uncertainty, as it is commonly referred to in the literature on this topic [7] [10] [9] [5].

2.2 Predictive Uncertainty Quantification

Naturally, this raises the question of how we can estimate the uncertainty of a prediction. A thought experiment comes in handy – if we consider a prediction as the mean of a probabilistic distribution, we can interpret the *variance* of that same distribution as a measure for predictive uncertainty.[4]

2.2.1 Bayesian Neural Networks

A well-studied approach for obtaining reliable estimations of uncertainty is therefore the use of *Bayesian neural networks* (BNN), which are based on Bayes' theorem[5] [13]. In BNNs, each individual model weight is treated as a probabilistic distribution instead of a single value – effectively allowing BNNs to inherently model uncertainty at the parameter level [14] [10]. However, directly computing these posterior distributions with traditional methods is often intractable due to the complex high-dimensionality [14]. This is why two general types of methods were introduced to solve this problem.

One of them is *Markov Chain Monte Carlo* (MCMC) methods, which sample the exact posterior distribution but are considerably more difficult to implement and computationally slow [14]. The other type is *Variational Inference* (VI) methods, which are more commonly used in Bayesian neural networks, as they typically come with much less computational overhead. In Variational Inference, the aim is to find an approximating distribution as close as possible to the true posterior distribution[6] [10]. Among others, a popular and note-worthy variant of Variational Inference is a BNN interpretation of a common dropout technique, *Monte Carlo* (MC) *dropout*[7], which we will discuss in more detail later in this section [14]. By approximating these probabilistic weight distributions, a BNN incorporates uncertainty about the model parameters directly into its predictions. The aggregation of these distributions allows the BNN to provide an overall uncertainty estimation, reflecting both the uncertainty in the model's parameters (epistemic uncertainty) and potentially in the data (aleatoric uncertainty), if modeled accordingly [14] [10].

However, compared to non-Bayesian deep neural networks, BNNs require various complex modifications in the implementation procedure and are significantly more expensive in their computation [15]. Additionally, applying explanation techniques to a BNN is considerably more difficult, as many popular methods, such as *Layerwise Relevance Propagation* (LRP) or *Integrated Gradients* (IG), need to be adapted first to work with such stochastic models [16].

[4] This interpretation aligns with standard statistical principles where variance is a measure of uncertainty in a probabilistic distribution [12].

[5] Bayes' theorem is a formula for updating the probability of a hypothesis by combining prior knowledge with new evidence. It is key for the so-called *Bayesian perspective* on uncertainty.

[6] This can be achieved by optimizing the parameters of the approximating distribution to minimize the *Kullback-Leibler* (KL) divergence between it and the true posterior distribution.

[7] Not to be confused with the Markov Chain Monte Carlo methods – Monte Carlo dropout is an example of the Variational Inference approach.

2.2.2 Non-Bayesian Neural Networks

Therefore, it appears reasonable to explore predictive uncertainty estimation in commonly used and more easily explainable non-Bayesian neural networks. Several techniques can be employed for this purpose, including the previously mentioned *Monte Carlo dropout* and different ensembling techniques[8] [15]. Dropout techniques like Monte Carlo, for instance, generally switch off randomly selected neurons in the model over multiple forward passes during test time, effectively utilizing this randomness to simulate the process of drawing samples from a probabilistic distribution. Thus, they can be seen as a Bayesian Variational Inference approximation [17]. These individual predictions can then be averaged and their variance can be interpreted as an estimation of predictive uncertainty [17]. As they are practically sampling from a distribution of networks, dropout techniques can also be seen as special cases of ensembling techniques [18]. MC dropout does not add significantly on the computation time, as dropout techniques are often already used as regularization techniques to prevent overfitting[9]. However, one must note that in Monte Carlo dropout, unlike in deep ensembles, the epistemic uncertainty is primarily influenced by the dropout rate and does not cover the entire model parameter space, which might lead to incorrectly sized uncertainty estimates [5].

2.3 Deep Ensembles

Ensembling generally involves creating M independent instances of a neural network model (*ensemble members* or *base learners*), which are randomly initialized in terms of their parameters and then trained on the same dataset or potentially on different subsets of the training data. These so-called *deep ensembles* were first proposed by [15] in 2017. They have been inspired by the concept of *bootstrapping* (or *bagging*), introduced by Breiman (1996) [6], which we will discuss later in this work [19]. While single non-Bayesian neural network models struggle to capture epistemic uncertainty effectively, combining them to deep ensembles does the trick – and can even lead to higher accuracy and calibration as well [20].

The resulting ensemble can capture epistemic uncertainty significantly better, since the diversity among ensemble members reflects different plausible hypotheses about the data, covering a wider parameter space [20]. This is because when making predictions with deep ensembles, we are drawing from a distribution of networks, incorporating the possible parameter states of multiple different models. Wilson and Izmailov (2022) [20] demonstrated that deep ensembles are therefore providing an effective mechanism for practically approximating Bayesian model averaging[10] (BMA), and are actually able to deliver a better approximation to the Bayesian predictive distribution than some standard Bayesian approaches. Their study highlights that deep ensembles are overall very effective in capturing epistemic uncertainty without the need for complex Bayesian computations. Ovadia et al. (2019) [21] coherently observed deep ensembles to generally outperform Bayesian neural networks in most metrics, particularly in being more robust under dataset shifts. Lakshminarayanan et al. (2017) [15] furthermore found that deep ensembles typically also surpass the previously mentioned Monte Carlo dropout in the quality of the uncertainty estimates. This also resonates with the findings of Ashukha et al. (2021) [22] in their comparison of different ensembling techniques.

[8] These are generally applicable for both Bayesian and non-Bayesian neural networks.

[9] Overfitting refers to the phenomenon of models being too well adapted to the training data, leading to models making worse predictions on new, unseen data, as their ability to generalize well declines.

[10] As BMA typically (soft) selects the model it assumes to be the best (and closest to the prior) in the hypothesis class, [15] noted that one should preferably speak of a more powerful *model combination*, rather than Bayesian model averaging, in deep ensembles instead.

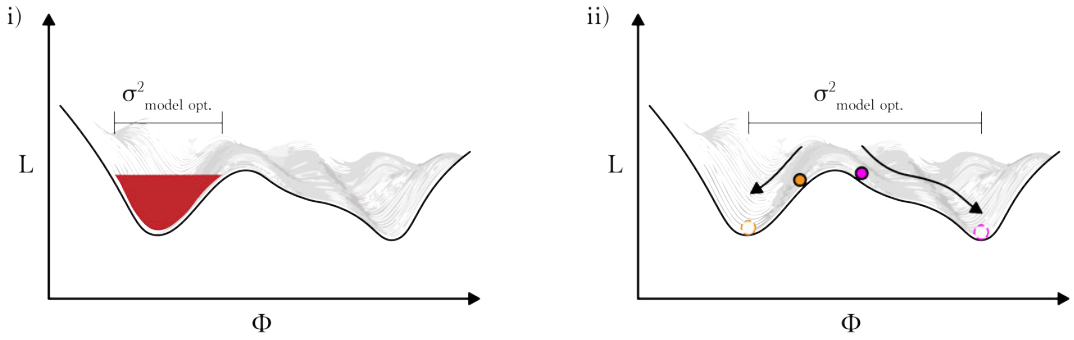

Figure 2: Simplified illustration of the loss-landscape. (i) Variational inference methods only explore a single mode in parameter space, (ii) deep ensembles with random initialisation explore M different modes.

When understanding the model optimization procedure as finding a local minimum on an overall loss landscape, Fort et al. (2020) [19] show that different (random) model initializations generally lead to different (equally good) local minima.[11] This explains why the M ensemble member predictions typically result in M different solutions. Fort et al. (2020) [19] further state that standard Bayesian methods using Variational Inference might fail to explore different of these modes (locally optimal minima in the loss landscape), even though they effectively capture uncertainty within a single mode, as seen in Figure 2. They conclude that because of this, deep ensembles may be better at generalizing than standard Bayesian methods. This gives yet another reason why deep ensembles can be considered the current state-of-the-art in predictive uncertainty estimation. Ovidia et al. (2019) [21] found that small ensemble sizes, e.g. $M = 5$ may be sufficient, which is coherent to the findings of Nixon et al. (2020) [4], in which most metrics do not significantly improve for ensemble sizes above that very number.

In practice, a collective ensemble prediction is made by passing the same inputs through all M members and averaging their outputs, effectively treating the ensemble as a *uniformly-weighted mixture model* [15]. The observed variance for each of these collective outputs can be interpreted as the associated uncertainty estimate for that prediction.

2.3.1 Non-probabilistic Deep Ensembles

There are multiple ways of constructing such deep ensembles in practice. Common deep neural networks typically use model architectures like *multi-layer perceptrons*[12] (MLP) for regression tasks, in which the models are optimized to minimize the *mean squared error*[13] (MSE) of their predictions as their loss function, as in Equation 5. The squaring operation in MSE is used to penalize larger errors more severely.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5}$$

where:

- n is the number of samples,
- y_i is the true observation for the i-th sample, and
- $\hat{y}_i$ is the predicted observation for the i-th sample.

As this is a common practice and easy to implement, it is deemed reasonable to take a closer look at this *non-probabilistic approach* on deep ensembles. It is very much possible to combine M of these randomly initialized models, which are trained on the same data set, to such an aforementioned ensemble. Neural networks trained to optimize their MSE typically output only one continuous

[11] This phenomenon is due to the non-convex nature of neural network loss landscapes. Diverse local minima can be equally optimal but vary due to different starting points in parameter space.

[12] Multi-layer perceptrons are fully-connected models, where each neuron in one layer is connected to each neuron in both the previous and the subsequent layer.

[13] Mean squared error (MSE) measures the average squared difference between predicted values and actual values.

value as their prediction. We can average a mean of these predictions y_m of all base learners to aggregate a collective ensemble prediction $\bar{y}$ for a given input, as in Equation 6.

$$\bar{y} = \frac{1}{M} \sum_{m=1}^{M} y_m \tag{6}$$

Next, we can extract the uncertainty estimate given by the variance $\bar{\sigma}^2$ of all model predictions in the ensemble for a given input, as in Equation 7 [1].

$$\bar{\sigma}^2 = \frac{1}{M} \sum_{m=1}^{M} (y_m - \bar{y})^2 \tag{7}$$

According to our framework of uncertainty, we can understand this uncertainty estimate of the variance of predictions as predominantly modeling epistemic uncertainty, as we are essentially sampling from a distribution of multiple model hypotheses about the true data-target relationship. However in practice, it might be possible that it also indirectly captures some aleatoric uncertainty, as samples with higher aleatoric noise may coherently result in predictions with higher epistemic uncertainty as well. Interestingly, a non-probabilistic network can actually be viewed as a special case of a probabilistic network that predicts a Gaussian distribution with a fixed, zero variance.

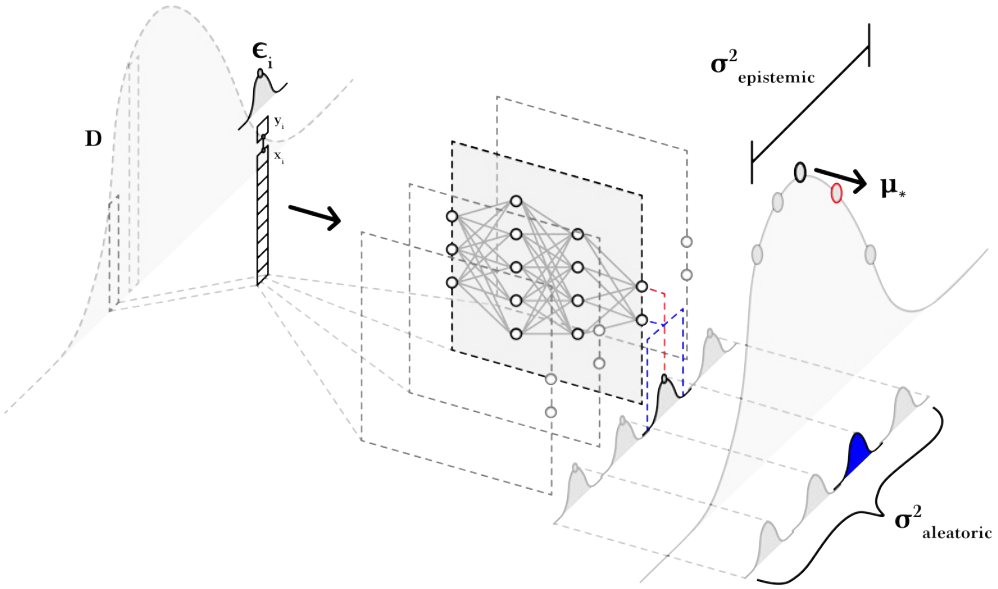

Figure 3: Illustration of decomposed predictive uncertainty estimation for probabilistic deep ensembles. Each base learner outputs a predictive distribution, consisting of mean (red) and variance (blue). Each sample is assumed to have inherent aleatoric noise ϵ_i, estimated by the mean of the variances of all base learners. Additionally, the variance of all mean predictions in the ensemble is the estimate for epistemic uncertainty.

2.3.2 Probabilistic Deep Ensembles

In their initial paper, Lakshminarayanan et al. (2017) [15] take a different approach on deep ensembles by using *probabilistic neural networks* (PNNs) for their ensembles, which do have some disadvantages in regards to computational costs, but by inherently learning distributions may also have potential benefits for the trustworthiness of the uncertainty estimates. Contrary to the previous approach, each model, e.g. a multi-layer perceptron, directly outputs a probability distribution for its predictions, effectively consisting of two values – a mean and a variance. Instead of using MSE[14] as a loss function, Lakshminarayanan et al. [15] utilize *negative log-likelihood* (NLL) for their optimization procedure. NLL is essentially measuring the likelihood of a probabilistic

[14]MSE can actually be seen as a special case of NLL, where the underlying error is assumed to be normally distributed.

model to predict the true distribution, which is why it requires these two probabilistic outputs of mean $\mu(x)$ and variance[15] $\sigma^2(x) > 0$. NLL optimization works by penalizing models that assign low probabilities to the true observations. To quantify this, it utilizes the negative logarithm[16] of the probability that the model assigned to the observed data, as in Equation 8. Therefore, lower values indicate a better fit.

$$\text{NLL} = -\frac{1}{n}\sum_{i=1}^{n}\log \mathbb{P}(y_i \mid x_i, \theta) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(y_i-\mu_i)^2}{2\sigma_i^2} + \frac{1}{2}\log(\sigma_i^2) + \frac{1}{2}\log(2\pi)\right) \tag{8}$$

where:

- n is the number of samples,
- y_i is the observed value for the i-th sample,
- x_i is the input features for the i-th sample,
- θ represents the parameters of the model,
- μ_i is the predicted mean for the i-th sample,
- σ_i^2 is the predicted variance for the i-th sample, and
- $\mathbb{P}(y_i \mid x_i, \theta)$ is the predicted probability of y_i given x_i and model parameters θ.

Similar to the previously described non-probabilistic approach, we can aggregate a collective ensemble prediction μ_* for the individually trained M models by averaging the predicted means μ_m of each model in Equation 9.

$$\mu_* = \frac{1}{M}\sum_{m=1}^{M}\mu_m \tag{9}$$

where μ_* corresponds to $\bar{y}$ in MSE-based approaches, and $\mu_m = \mu_\theta(x_i)$. As we are treating the ensemble of probabilistic models as a mixture of distributions, according to the law of total variance, [23] we can compute the collective variance σ_*^2 as in Equation 10. This directly implies our overall predictive uncertainty estimate [15].

$$\sigma_*^2 = \frac{1}{M}\sum_{m=1}^{M}\left(\sigma_m^2 + \mu_m^2\right) - \mu_*^2 \tag{10}$$

where $\mu_m = \mu_\theta(x_i)$ predicted mean, and $\sigma_m^2 = \sigma_\theta^2(x_i)$ predicted variance of one ensemble member. It is important to note that in line with the law of total variance, σ_*^2 is essentially a composition of the mean of the individual model variances, Equation 11, and the variance of the individual model means, Equation 12. As the variance of a probabilistic model reflects the noise in the data, Equation 11 directly accounts for the aleatoric uncertainty of the ensemble. [5] Secondly, the variance of the individual predictive means captures the epistemic uncertainty, because we are again sampling these predictions from a distribution of various hypotheses about the true data-target relationship. Furthermore, we can see that the aleatoric uncertainty term is depending on the input x, which effectively means that the probabilistic approach is capable of capturing and quantifying heteroscedastic aleatoric noise. Again, as described before, it might be possible that the epistemic term captures some aleatoric noise through indirect effects as well.

$$\text{AU}_* = \frac{1}{M}\sum_{m=1}^{M}\sigma_m^2 \tag{11}$$

$$\text{EU}_* = \frac{1}{M}\sum_{m=1}^{M}\mu_m^2 - \mu_*^2 = \frac{1}{M}\sum_{m=1}^{M}(\mu_m - \mu_*)^2 \tag{12}$$

Lakshminarayanan et al. (2017) [15] argue that utilizing the probabilistic approach offers notable advantages compared to the non-probabilistic approach, because it leads to better-calibrated[17]

[15] Lakshminarayanan et al. [15] ensure the variance's positivity constraint by passing the second output through $\text{softplus}(x) = \log(1 + e^x)$, and adding a minimum variance for numerical stability (e.g., 10^{-6}).

[16] The logarithm is used because it improves the optimization process by penalizing low-probability events more heavily. Additionally, for optimization problems, it makes conceptually more sense to aim for a minimization rather than maximizing the likelihood. This is why the negative logarithm is applied.

[17] Well-calibrated predictions means that the predicted probabilities accurately reflect the true likelihood of the outcomes.

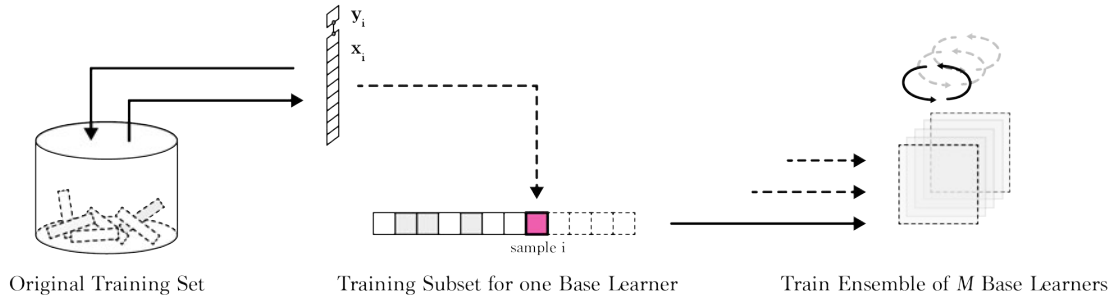

Figure 4: Illustration of the naïve bootstrap principle.

predictions. They claim that MSE-based approaches would consistently underestimate the true uncertainty. From this, they conclude that the trustworthiness of the uncertainty estimates is higher for deep ensembles with probabilistic neural networks. However, they only show this for one dataset as an example, and do not mention the easy access to well-known explanation methods for non-probabilistic approaches – in fact, they do not mention explanation techniques at all. Lakshminarayanan et al. [15] do highlight the benefits of optional *adversarial training*, in which added particular augmentations to the training set can increase the robustness of an ensemble. We will ignore this supplementary procedure for now, as we would like to keep the ensembles easy to compare and isolate them from additional effects on explained predictive uncertainty.

2.4 Bootstrapping Deep Ensembles

As we have discussed up to this point, both the non-probabilistic approach and the probabilistic approach can be considered sufficiently capable of capturing epistemic uncertainty due to the underlying idea of sampling from a distribution of multiple hypotheses, as each ensemble member holds a different parameter space caused by random initialization. Yet, they are typically trained on the very same training data. For this reason, they appear to ignore a potentially significant additional source of epistemic uncertainty, which is due to the finite number of data points sampled from the unknown true distribution [5]. We call this type of epistemic uncertainty the sampling uncertainty, and we can actually attempt to decompose epistemic uncertainty $\epsilon_{epistemic}$ into the two sources of sampling uncertainty and model optimization [5]:[18]

$$\epsilon_{epistemic} = \sigma^2_{sampling} + \frac{\sigma^2_{modeloptim.}}{M} = \sigma^2_{sampling} + \frac{1}{M}\sum_{m=1}^{M}(\mu_m - \mu_*)^2 \tag{13}$$

2.4.1 Naïve Bootstrap

To account for this sampling uncertainty, a certain extent of variation in the training sets needs to be introduced. The well-known procedure of *bootstrapping* (or *bagging*), introduced by L. Breiman [6] in 1996, has been discussed by several researchers as a possible way of doing just this, as it is a widely used approach for introducing randomness into a dataset [15] [4] [5]. When using bootstrap, the M ensemble members or *base learners* generally do not train on the same training set.[19] Instead, M subsets of the same length are created from the original training data, as illustrated in Figure 4. It is important to note that these subsets are created with replacement, which introduces the chance of some data points being represented in the set multiple times, while others might not be included at all. Each subset is then used to train a different ensemble member. Intuitively, by increasing the variability in the data, we should be able to get a more realistic estimate for epistemic uncertainty, and ideally also achieve a higher robustness for the models.

However, Lakshminarayanan et al. (2017) [15] found that using this *naïve bootstrap* approach on deep ensembles often does not lead to better model performance metrics (such as RMSE[20]), in

[18] The sampling uncertainty does not get divided by M as it is the same for all base learners [5].

[19] Training the members with the very same training set can de facto be viewed as a special case scenario in bootstrapping.

[20] Root mean squared error – the square root of the MSE.

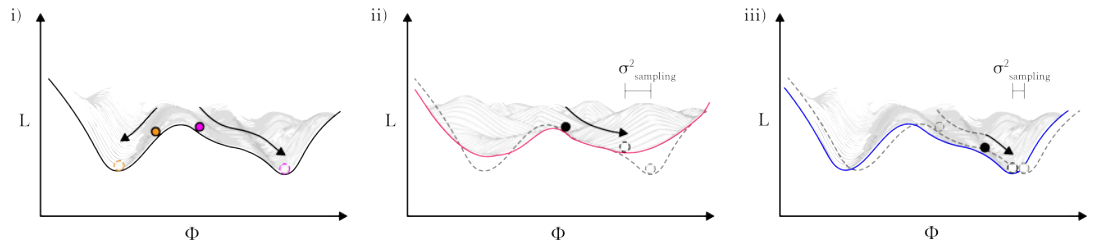

Figure 5: Simplified illustration of the loss-landscape. (i) Non-bootstrapped deep ensembles explore M different minima on same loss landscape, (ii) naïve bootstrap leads to M different loss landscapes with different minima, (iii) parametric bootstrap leads to M similar loss landscapes with similar minima.

scenarios where deep ensembles were trained to optimize accuracy. They assume this to be due to the possibility of bootstrapped base learners not being able to find similarly adequate optima, as the loss-landscape looks different due to missing a portion of the unique data points. Figure 5 illustrates this. They further state state that in standard bootstrapping, the number of unique samples in each bootstrap subset is on average 63,2%. Nixon et al. (2020) [4] therefore question the benefit of bootstrap in deep learning in general, as they wonder if it even introduces a valuable form of randomness, in contrast to random initialization, hyperparameter and architecture spaces. They further argue that not the total quantity of data points is relevant for a model's ability to learn a distribution well, but the percentage of unique data points in the training set.

$$\mathbb{E}[\textit{unique data points}] = n \times \left(1 - \left(1 - \frac{1}{n}\right)^n\right) \Rightarrow \lim_{n\to\infty} \left(1 - \left(1 - \frac{1}{n}\right)^n\right) = 1 - \frac{1}{e} \Rightarrow 0.632 \times n$$

2.4.2 Parametric Bootstrap

This leads to the question if this issue can be addressed by improving the concept of the bootstrap. Sluijterman et al. (2023) [5] therefore proposed an alternative strategy for deep ensembles, adapting an approach by B. Efron (1982) [24] called the *parametric bootstrap*. The basic idea is to utilize new targets generated by already trained models. In this approach, one first trains M base learners (randomly initialized) with the same training set. While training, the states of the models are saved at a specific epoch $\lfloor N_{\text{epoch}} \times (1 - r) \rceil$, constituted by the retraining fraction r [5]. The base learners continue the training on the original training set to then make predictions for the original inputs x. These generated targets are then used as a new training set for each previously saved base learner to finalize the training in a second iteration. When taking the loss landscape perspective, this approach preserves the general areas of the local minima from the first training iteration, with added slight local deviations by the first iterations predictions to induce the variability. By doing this, Sluijterman et al. [5] showed to be able to estimate the *sampling uncertainty* as in Equation 14. They implicitly assume the ensembles to have a random but fixed initialization towards each other, which means that the random initialization needs to be the same for two corresponding base learners in the two ensembles.[21]

$$\sigma^2_{sampling} = \frac{1}{M} \sum_{m=1}^{M} (\hat{f}_m(x) - \hat{\hat{f}}_m(x))^2 \tag{14}$$

where $\hat{f}_m(x)$ is the first iteration model, and $\hat{\hat{f}}_m(x)$ the second iteration model.

2.5 Trustworthiness of Uncertainty Estimates

To obtain reliable uncertainty estimates, it is crucial to understand their trustworthiness. However, assessing the quality of uncertainty estimates is not straightforward. Many papers on this topic typically approach it from two different perspectives: the *out-of-domain* (OOD) perspective and the *in-domain* perspective [22].

[21] Another assumption made here is that the difference of the first model and the true relationship, and of the first and the second model, are similar [5].

2.5.1 Out-of-Distribution Samples

The OOD perspective aims to quantify the ability of a model or ensemble to correctly identify *out-of-distribution* samples, which are samples that significantly differ from the training data. These differences can either be due to augmentation with a considerable amount of noise, or because they are taken from an entirely different distribution [15]. Trustworthy uncertainty estimates should assign much higher predictive uncertainty to these samples than to the original non-augmented, in-distribution data points. A common approach of doing this is by creating two different test sets, one from the original distribution and one as an augmented copy with significant noise on the input features, $X_{OOD} = X_{test} + \alpha \times \mathcal{N}(0, \sigma^2)$, to simulate a shift in the data distribution. Then, the trained ensemble assigns uncertainty scores to each sample of the two sets. It is then possible to combine the sets and compute the *Area Under the Receiver Operating Characteristic curve* (AUROC), which quantifies the ability of the ensemble to correctly rank (or identify) positive cases (OOD samples) higher than negative cases (original samples). A higher value indicates a better distinction of the OOD samples.[22]

2.5.2 Calibration

On the other hand, the *in-domain* perspective focuses on how a model or ensemble performs in scenarios where it is expected to produce reliable predictions. A key aspect of this is understanding the *calibration* of a model. Calibration is a measure that evaluates the statistical consistency of a model's predictions to match the actual values, indicating if a model is generally overconfident or underconfident about its predictions [25]. However, there are various strategies of measuring the calibration of a model in practice, and the discussion on which is best to use is still active [22] [26] [15] [27]. For regression settings, calibration can be defined in at least two different ways, using quantiles[23] and using prediction intervals [28] [26]. The quantile approach usually requires some modifications in the model architecture and is linked to so-called quantile regression.

Thus, prediction intervals are more commonly used, as they are more intuitive, generalizable and easy to interpret. Prediction intervals are ranges with a specified probability or confidence level, within which a future observation is expected to fall. To get the prediction intervals, we first define the corresponding confidence levels z (e.g. $z = 90\%, 80\%, \ldots$). We can then obtain the lower and upper bounds for the prediction intervals by computing $Bounds = Mean \pm z \times \sqrt{Variance}$ on the ensemble predictions for each confidence level. Once we have the bounds, we can measure the observed fractions, or *prediction interval coverage probability* (PICP), of true values that fall within these prediction intervals. A common way of assessing the calibration curve is to then plot the PICP with one axis being expected fractions and one being observed fractions. The ideal line for a well-calibrated model is the diagonal – if the curve falls below the diagonal the predictions are overconfident, if it is above it indicates underconfidence. The area between the curve and diagonal can be quantified as the *miscalibration area* to obtain an overall calibration estimate, as well.

Other noteworthy measures include the *expected calibration error* (ECE), which is more commonly applied in classification tasks [29]. That said, the ECE has some flaws on its own, as it might lead to different results than other calibration quantification methods [4] [22] [26]. This is why some researchers also rely on proper scoring rules [30] for their calibration analysis, such as the negative log-likelihood or the *Brier score*[24] [31]. However, a good performance in NLL does not necessarily translate to well-calibrated predictions [26]. Sluijterman et al. (2024) [26] therefore suggest exploring simulation-based point-wise coverage. For this work though, the common approach of prediction intervals appears to be a reasonable state-of-the-art method to obtain comparable calibration measures.

In terms of calibration, Lakshminarayanan et al. (2017) [15] found probabilistic deep ensembles trained on minimizing NLL to outperform non-probabilistic deep ensembles (trained on minimizing MSE), as they found them to produce less overconfident predictions. Sluijterman et al. (2023) [5] further found ensembles which utilize the parametric bootstrap to outperform both non-bootstrap and naïve bootstrap ensembles on some calibration measures such as the Brier score. In summary, assessing the reliability of uncertainty estimates, and thus of their explanations, should generally

[22] AUROC values are in a range of 0 to 1, with 1 being the optimal distinction between both classes and 0.5 being as good as random guessing.

[23] Subgroups of a certain percentage of the data points, dependent on the distribution – e.g. 25% of all students scored below a certain value.

[24] The Brier Score is also more commonly used in classification settings and is closely related to the MSE.

include an extensive analysis on both the out-of-domain perspective and the in-domain perspective on trustworthiness.

2.6 Explanation Techniques

In recent years, several explanation techniques for neural networks have been introduced. The term explanation technique refers to finding reliable methods to explain a model's output with respect to the input. These techniques can be further categorized into *global* and *local* explanation techniques, where local techniques focus on the attribution of explanations within individual samples, which makes them particularly interesting for this work [16]. Most of these local attribution methods assign some sort of relevance scores to areas or features in the input. Ideally, this should be done in a form that is as intuitive as possible for humans to understand. Typically, this includes visual representations such as highlighting relevant features, areas or pixels, for example in heatmaps. Samek et al. (2021) [3] provide a good overview of different techniques, though in this work we will solely focus on two.

2.6.1 Gradient × Input

A popular explanation technique due to its simplicity is *Gradients* × *Input* (GI), which can give useful explanations in regression settings for simple models [32] [1]. When using Gradient × Input on an ensemble, the general idea is as follows: For each model in the ensemble, we first compute the gradient of the output with respect to the input. We then simply multiply the gradients by the input values (element-wise) to get the Gradients × Input explanations. These explanations can now be averaged across the ensemble members to get the ensemble explanations for the input dataset. It is important to note that these explanations can sometimes be unreliable due to the phenomenon of *shattered* or noisy gradients, particularly in more complex models [33] [34].

$$G \times I_*(x) = \frac{1}{M} \sum_{m=1}^{M} (x \odot \nabla_x \hat{f}_m(x)) \tag{15}$$

where $\nabla_x \hat{f}_m(x)$ denotes the gradients for an input x in a base learner $\hat{f}_m(x)$.

2.6.2 Layerwise Relevance Propagation

A more reliable and well-scalable explanation technique is *Layerwise Relevance Propagation* (LRP), as introduced by Bach et al. (2015) [35]. LRP allows the predictions of a neural network to be attributed back to the input features by propagating the output backwards through all layers of the network.[25] Essentially, this redistribution is taking into account the weights and activations of each neuron to determine its contributions to the final output as computed in the forward-pass. The technique effectively redistributes the output recursively and results in distributed relevance scores across all neurons in the model. LRP is leveraging a *conservation* property, which means that a neuron has to redistribute everything it received to the lower layer [33]. Finally, this also assigns relevance scores across the initial input features, which practically explain the decision process of the neural network for an input sample. To utilize this technique correctly, specific LRP-rules need to be applied according to the type of the layer in the network. The most basic of these is the LRP-0 rule (Equation 16), which is essential for understanding the concept. A comprehensive overview for the different rules and its usages can be found in Montavon et al. (2019) [33], to be extended also by the generalized LRP-γ rule as described by Keyl et al (2022) [36] , Equation 17.

LRP-0:

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{j'} a_{j'} w_{j'k}} R_k \tag{16}$$

Generalized LRP-γ:

$$\begin{aligned} R_j = &\sum_k \left[\frac{a_j^+ \left(w_{jk} + \gamma w_{jk}^+\right) + a_j^- \left(w_{jk} + \gamma w_{jk}^-\right)}{\epsilon + \sum_{j'} a_{j'}^+ \left(w_{j'k} + \gamma w_{j'k}^+\right) + a_{j'}^- \left(w_{j'k} + \gamma w_{j'k}^-\right)} \cdot \mathbf{1}_{a_k>0} R_k \right] \\ &+ \sum_k \left[\frac{a_j^+ \left(w_{jk} + \gamma w_{jk}^-\right) + a_j^- \left(w_{jk} + \gamma w_{jk}^+\right)}{-\epsilon + \sum_{j'} a_{j'}^+ \left(w_{j'k} + \gamma w_{j'k}^-\right) + a_{j'}^- \left(w_{j'k} + \gamma w_{j'k}^+\right)} \cdot \mathbf{1}_{a_k<0} R_k \right] \end{aligned} \tag{17}$$

[25] Thus, it is a model-aware attribution technique [16].

where:

- R_j is the relevance of neuron j,
- R_k is Relevance of neuron k in the next layer,
- a_j is the activation of neuron j,
- a_j^+ and a_j^- are positive and negative parts of the activation a_j, where $a_j^+ = \max(a_j, 0)$ and $a_j^- = \min(a_j, 0)$.
- w_{jk} is the weight connecting neuron j to neuron k,
- w_{jk}^+ and w_{jk}^- are positive and negative parts of the weight, respectively,
- γ is a scaling parameter,
- ϵ is small constant for numerical stability,
- $\mathbf{1}_{a_k>0}$ is an indicator function that equals 1 if $a_k > 0$ and 0 otherwise,
- and $\mathbf{1}_{a_k<0}$ is an indicator function that equals 1 if $a_k < 0$ and 0 otherwise.

2.6.3 Second-Order Explanations with Covariance Matrices

As with uncertainty in everydays life, we can assume that epistemic uncertainty in an ensemble prediction may very well be caused by several features in an input at once, possibly even in correlation to each other. To expose these second-order effects on predictive uncertainty, Bley et al. (2024) [1] recently introduced a novel method. They effectively show that using a simple *covariance*[26] computation over the individual feature explanations given by standard attribution techniques (such as LRP or GI), the role of the interaction between pairs of features can be investigated in detail. The resulting $d \times d$ matrix reveals the joined contributions of pairs of features, as they are contributing to the overall uncertainty in correlation. Bley et al. [1] show the effectiveness of their explanation approach by testing it with the *area under the flipping curve* (AUFC) method, in which single features for high uncertainty samples are "flipped" in order of their explanation's relevance scores. Then, the uncertainty estimate for this flipped sample is computed again. If the explanations are reliable, the curve should drop steeper with the most relevant features. The area under that curve can then be used for a comparison or validation of the quality of different explanation techniques.

$$\mathcal{E}(\sigma_*^2, xx^T) = \mathrm{Cov}_m(\mathcal{E}(y_m, x)) \tag{18}$$

For better comparability and interpretability, Bley et al. [1] further isolate the matrix elements into *Cov Diag* (only the elements of the diagonal, representing the individual feature explanations) and *Cov Marg* (the column-wise sum over each column of the matrix, incorporating also the second-order explanations). They conclusively show with the AUFC comparison, that using *Cov Diag* with LRP explanations typically results in a better quality of explanations compared to plain LRP.

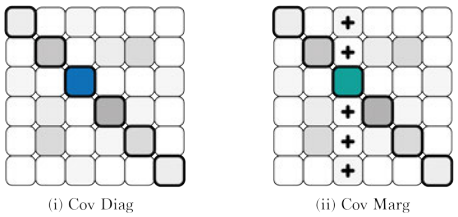

Figure 6: Illustration of *Cov Diag* (i) and *Cov Marg* (ii).

All in all – with a comprehensive understanding of the nature of predictive uncertainty, various methods for estimating uncertainty using deep ensembles, several approaches to quantifying their reliability, and advanced explanation techniques, we are now well-prepared to analyze the impacts of bootstrapping deep ensembles on explained predictive uncertainty.

[26] *Covariance* is a statistical measure that indicates the extent to which two variables change together. Positive covariance means that the correlation is in the same direction (when one variable is high, the other is high), negative covariance means the opposite effect (one is high, the other is low).

3 Method and Hypotheses

As this work aims to explore the effect of bootstrapping deep ensembles on predictive uncertainty explanations, three hypotheses are proposed to be assessed in the following experimental analysis. These hypotheses will be tested in two experiments: The first is a real-world regression dataset benchmark, and the second is in a specifically designed environment of a synthetic toy dataset. The primary focus of this analysis will be on the effect of the *naïve bootstrap* (NB), though in these experiments, the *parametric bootstrap* (PB) will be examined in comparison.

3.1 Research Hypotheses

To recall, the motivation for bootstrapping is to account for the previously ignored sampling uncertainty by introducing an additional variability into the training process of deep ensembles. With bootstrapping, every ensemble member or base learner has different samples of the true unknown distribution to train on. Thus, it appears reasonable to expect a greater estimation for the epistemic uncertainty, as the ensemble will likely have a greater variance in its predictions due to the exploration of different loss landscapes. As a result, the uncertainty estimates should be in closer proximity to the true uncertainty inherent to the ensemble prediction, particularly in regions of the dataset where the data is sparse or highly variable. This may also be reflected in an improved negative log-likelihood (NLL).

Hypothesis 1: *Bootstrapping deep ensembles increases epistemic uncertainty estimates by introducing sampling variability, which enhances model diversity. These potentially improved uncertainty estimates may lead to better NLL.*

Subsequently, this raises the question on how great this effect will be, and how it will affect performance and trustworthiness metrics (such as calibration and OOD detection). As trustworthiness is a key desirable property for predictions and uncertainty estimates alike, the effect on calibration and out-of-distribution samples will be of particular interest. For the calibration, the introduced variability and thus more diverse prediction range may lead to an improved ensemble calibration. Intuitively, one would expect the same for the ability to detect out-of-distribution samples, as the diversity among the models might also reduce the likelihood of overfitting to specific patterns in the training data, and thus might lead to more generalized and robust uncertainty estimates. These effects could vary with the size and complexity of the datasets, as we can expect the effect of sampling variability to be proportionally stronger with smaller datasets. The effect could furthermore vary depending on the complexity of the models.

Hypothesis 2: *Bootstrapping deep ensembles increases the model calibration. The effects may differ on dataset size and model complexity, and may be more pronounced on small datasets.*

Finally, this will also lead to the question of how it will impact the quality and reliability of the uncertainty explanations. Adding on a potentially better calibration, it appears likely to expect bootstrapped deep ensembles to be less prone to learning *spurious correlations*[27], especially in small datasets or those with strong but irrelevant feature relationships. This effect might be particularly visible in the naïve bootstrap's explanations. Bootstrapping may amplify the relevance of aleatoric noise, as different base learners on each of their limited datasets may learn different noise patterns. Conversely, averaging explanations across bootstrapped models may enhance interpretability by smoothing out arbitrary relevance scores.

Hypothesis 3: *Bootstrapping may be more effective in detecting relevant noisy features that lead to uncertainty. As a result, explained predictive uncertainty in naïve bootstrapped deep ensembles may be less prone to spurious feature correlations.*

[27] Correlations between features that do not reflect any true underlying relationship but rather arise due to randomness.

4 Experimental Analysis

4.1 Experiment 1: Benchmark

This first experiment benchmarks the different ensemble configurations on ten widely used real-world regression datasets of different sizes. It aims to test the first and the second hypothesis by obtaining comparative metrics – such as RMSE, NLL, calibration, OOD and uncertainty estimates. To gain initial insights on the effect of bootstrapping deep ensembles on explained predictive uncertainty, the experiment will consider both non-probabilistic and probabilistic deep ensembles to begin with, before the scope will be narrowed down.

Dataset	# Samples	# Features	Description	Target Range
Yacht Hydrodynamics	308	6	Predicts the hydrodynamic performance of sailing yachts	0 to 4.14
Diabetes	442	10	Predicts if a patient has diabetes	25 to 346
Energy Efficiency	768	8	Predicts the heating load of buildings	6.01 to 43.1
Concrete Compression Strength	1,030	8	Predicts the compressive strength of concrete	2.33 to 82.6
Wine Quality Red	1,599	11	Predicts the quality of wine based on physicochemical tests	0 to 10
Kin8nm	8,192	8	Predicts the forward dynamics of an 8-link robot arm	-0.1 to 0.1
Combined Cycle Power Plant	9,568	4	Predicts the electrical power output of a power plant	420.26 to 495.76
Naval Propulsion	11,934	16	Predicts the propulsion behaviour of naval vessels	0 to 1 (normalized)
Protein Structure	45,730	9	Predicts the distance between residues in protein sequences	2 to 20
Year Prediction MSD	515,345	90	Predicts the year of release of a song from its audio features	1922 to 2011

Table 1: Overview on the benchmark datasets, sorted by the number of samples.

4.1.1 Experimental Setup

Maintaining a consistent comparability, each ensemble has the same $M = 5$ fully connected multi-layer perceptrons (MLPs) as its base learners. Each of these models is composed of four layers, with the only difference between the two ensemble types being the output layer. The three hidden linear layers have 900, 600, and 300 neurons, respectively. All hidden layers are followed by ReLU activation functions. The layers are implemented with a special modification for the generalized LRP-γ rule (as in [36]), to be able to perform relevance propagation for the LRP explanations.[28] However, in their general modus operandi they remain standard PyTorch linear layers. For initialization, *Kaiming Uniform* is used, as it is the standard for linear layers in PyTorch. For the final top layer, the non-probabilistic ensembles only have one output, while the probabilistic ensembles have two outputs: mean and variance. The correct variance is ensured with a softplus positivity constraint, as in the original implementation by Lakshminarayanan et al. (2017) [15].

Across all ensembles, each base learner is then set up with a consistent random initialisation. This effectively means that each M-th base learner in every ensemble is initialized with the identical random seed, which is important for maintaining comparability. All ensembles are trained using PyTorch, with mean squared error (MSE) or negative log-likelihood (NLL) as loss functions, matching the ensemble type. Each ensemble is trained for 100 epochs, with a batch size of 100 and an Adam optimizer with learning rate of 0.01. The loss is computed for every processed batch of data, then the gradients are computed and the model's weights are updated. After each epoch, the model's performance is evaluated on a separate validation set (10% of the data), and the best model state is saved for final use. For the non-bootstrap ensembles, after 70% of epochs, the model state is additionally saved to be used for the parametric bootstrap later on. This whole experiment pipeline is then repeated a total of 20 times and all measures are averaged accordingly. This repetition and averaging is done to smooth out the inherent randomness for every experiment and obtain a better comparability for the ensembles.

On every iteration of the training pipeline, before training the ensembles, the data set is shuffled and split into 75% training set, 15% test set, and 10% validation set. Each input set is then standardized to have a mean of zero and a standard deviation of one, using *StandardScaler* from the *scikit*-learn library (fitted on the training data). For the naïve bootstrap ensembles, the training set is then used to create an individual random subset of the same length (with replacement) for each base learner to train on. For the parametric bootstrap ensembles on the other hand, the previously trained non-bootstrap ensembles are used to make predictions on the training set, so that these predictions can then be used as synthetic training set for the saved model states (which were saved at 70% of epochs, as mentioned before), to finalize their training and form the parametric bootstrap ensemble.

[28] With a γ set to 0.2, as proposed by Bley et al. (2024) [1].

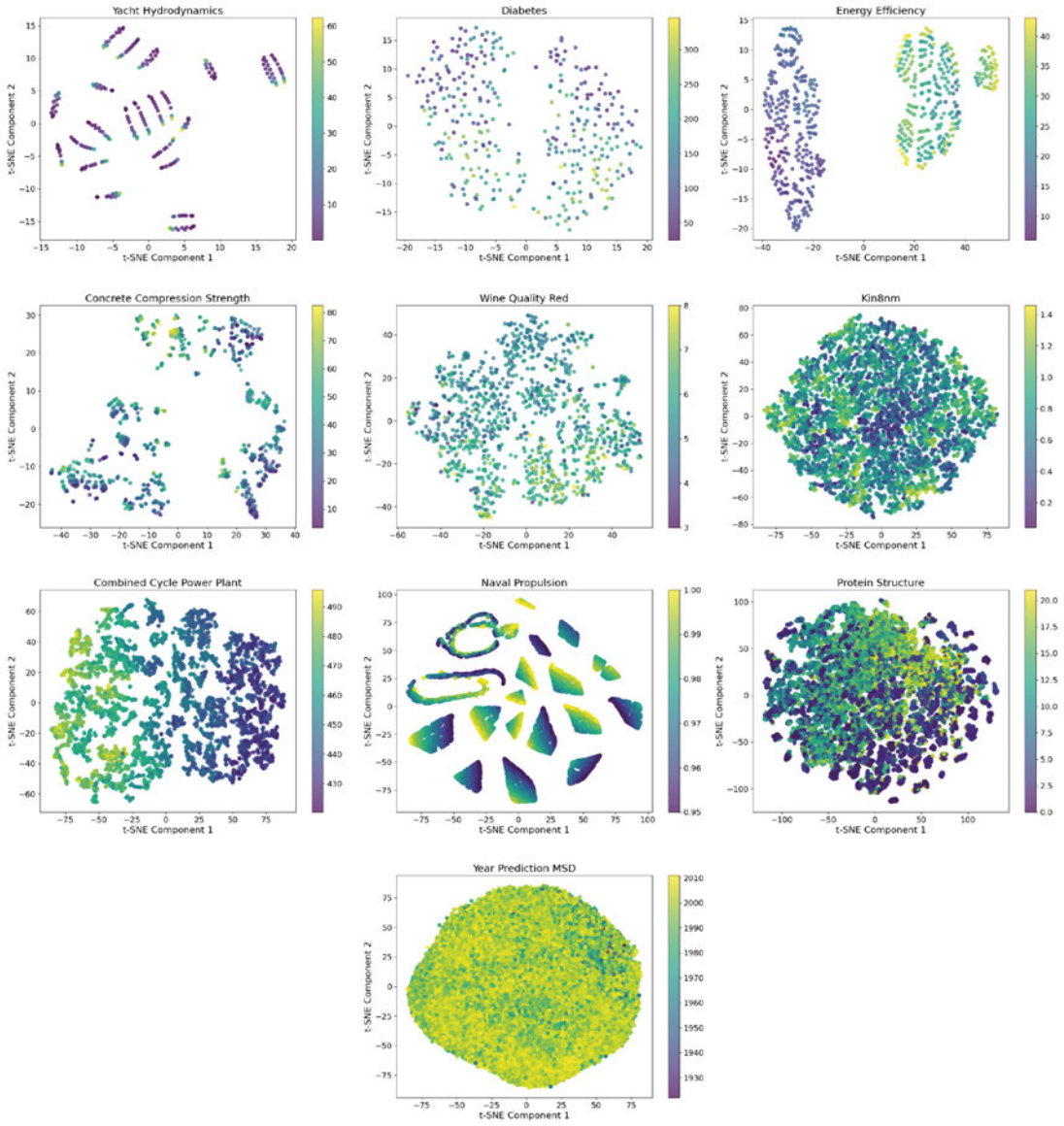

Figure 7: t-SNE plots for all benchmark datasets.

The choice of datasets is oriented on a benchmark by Hernández-Lobato and Adams (2015) [37], containing ten real world datasets, as seen in Table 1. This benchmark became a standard for predictive uncertainty in regression tasks, as it has been used in many papers on this topic ([17] [15] [26], [38] and more). However, contrary to the initial benchmark, this experiment will not use the Boston Housing dataset, as it generally should not be used any longer due to ethical considerations about the inherent racist structure [39]. Instead, the similarly sized diabetes dataset will be used, to still be able to assess the effect of bootstrapping deep ensembles in relatively small dataset scenarios. To get a better understanding for the diversity of the benchmark data, two-dimensional t-SNE plots[29] can be seen in Figure 7 for the datasets. Furthermore, the experiments conducted in this work include some changes in the model architecture and training process by using more than one hidden layer, as in [1] and [5], and a higher number of epochs (100 compared to 40), to generally adjust the models better to their tasks. For the number of iterations of the experiment pipeline, a total of 20 times will be used, with an exception for the *Year Prediction MSD* dataset, which is only run twice[30] due to the size and computational time.

[29] A t-SNE plot is a visualization technique that reduces high-dimensional data to three, or in this case two dimensions, while maintaining local relationships between data points. This helps in exploring the structure of the data, as in clusters or other patterns, and finding anomalies or outliers.

[30] Twice for the non-probabilistic ensembles and one time for the probabilistic ensembles.

RMSE

	Non-Probabilistic Ensembles			Probabilistic Ensembles		
Dataset	Deep Ensemble	Naïve Bootstrap	Param. Bootstrap	Prob.-DE	Prob.-NB	Prob.-PB
Yacht	1.14 ± 0.57	1.43 ± 0.61	**1.13 ± 0.58**	15.38 ± 1.81	**15.28** ± 1.90	15.31 ± 1.88
Diabetes	65.84 ± 4.67	**61.62 ± 6.37**	66.81 ± 4.89	**82.30 ± 5.56**	82.76 ± 6.38	82.54 ± 5.49
Energy	**0.56 ± 0.09**	0.71 ± 0.19	0.58 ± 0.09	10.05 ± 0.46	**10.02 ± 0.44**	10.11 ± 0.47
Concrete	**4.85 ± 0.54**	5.08 ± 0.48	4.92 ± 0.52	16.70 ± 0.63	16.68 ± 0.65	**16.60 ± 0.68**
Wine	**0.64 ± 0.04**	0.65 ± 0.04	0.65 ± 0.04	3.29 ± 8.17	**1.00 ± 0.48**	4.03 ± 7.88
Kin8nm	0.072 ± 0.006	**0.071 ± 0.002**	0.072 ± 0.006	**0.33 ± 0.22**	0.56 ± 1.06	0.37 ± 0.26
CCPP	**4.36 ± 0.43**	4.38 ± 0.57	4.46 ± 0.39	36.45 ± 64.40	**18.14 ± 2.03**	39.19 ± 65.61
Naval	0.0031 ± 0.0014	**0.0024 ± 0.0011**	0.0034 ± 0.0015	**0.35 ± 1.01**	1.51 ± 6.45	0.51 ± 1.19
Protein	**3.73 ± 0.05**	3.74 ± 0.05	3.74 ± 0.05	6.12 ± 0.04	6.12 ± 0.04	**6.12 ± 0.04**
Year MSD	10.32 ± 0.19	**10.22 ± 0.31**	10.27 ± 0.76	**30.96 ± n/a**	55.85 ± n/a	33.57 ± n/a

NLL

	Non-Probabilistic Ensembles			Probabilistic Ensembles		
Dataset	DE	NB	PB	Prob.-DE	Prob.-NB	Prob.-PB
Yacht	0.83 ± 0.90	**0.60 ± 0.47**	0.75 ± 0.64	4.12 ± 0.10	4.11 ± 0.10	**4.11 ± 0.12**
Diabetes	18.86 ± 6.35	**9.81 ± 2.97**	15.28 ± 3.65	**5.87 ± 0.10**	5.87 ± 0.10	5.89 ± 0.11
Energy	2.18 ± 1.50	2.35 ± 3.49	**1.53 ± 0.84**	3.73 ± 0.04	**3.73 ± 0.04**	3.74 ± 0.05
Concrete	7.82 ± 3.29	**5.15 ± 1.10**	5.89 ± 1.43	4.23 ± 0.04	4.23 ± 0.04	**4.22 ± 0.05**
Wine	6.88 ± 2.71	**3.82 ± 1.36**	4.62 ± 1.31	2.12 ± 1.28	**1.90 ± 1.23**	2.11 ± 1.24
Kin8nm	0.31 ± 0.48	-0.06 ± 0.33	**-0.14 ± 0.35**	0.58 ± 0.85	0.72 ± 1.03	**0.57 ± 0.85**
CCPP	3.11 ± 0.24	3.21 ± 0.57	**3.09 ± 0.20**	4.84 ± 0.96	**4.69 ± 0.69**	4.86 ± 0.97
Naval	-4.26 ± 0.40	**-4.47 ± 0.41**	-4.06 ± 0.51	-1.67 ± 2.35	**-2.34 ± 1.35**	-1.67 ± 2.35
Protein	13.21 ± 2.00	**10.00 ± 1.63**	11.89 ± 1.76	3.23 ± 0.01	3.23 ± 0.01	**3.23 ± 0.01**
Year MSD	4.36 ± 0.53	4.20 ± 0.43	**4.29 ± n/a**	4.22 ± 0.43	**3.83 ± n/a**	4.25 ± n/a

Table 2: Averaged RMSE and NLL results for the different ensemble configurations across the dataset benchmark. Ensembles are denoted as non-bootstrapped deep ensemble (DE), naïve bootstrap (NB), parametric bootstrap (PB), and the prefix (Prob.) for the probabilistic ensemble variants. The best performing ensemble configuration for each dataset and for each ensemble type is highlighted.

4.1.2 Experimental Results

After training, general performance metrics such as RMSE and NLL are evaluated on the test set. The resulting metrics, averaged across all runs, are reported in Table 2. We can observe a balanced picture for the RMSE values of non-probabilistic ensembles, in which both the non-bootstrap and the naïve bootstrap appear to perform slightly better than the parametric ensemble, as can be expected due to the artificial data points introduced by the parametric bootstrap. Noteworthily however, the non-probabilistic approach performs better in the RMSE metric than the probabilistic approach. This may be due to the fact that the non-probabilistic ensembles are optimized for the mean squared error, while the others are primarily optimized for NLL. Additionally, when comparing these results to the reports of Lakshminarayanan et al. (2017) [15], it is worth mentioning that in this experiment the gap between the non-probabilistic and the probabilistic variants appears to be larger for RMSE. Furthermore, in comparison to the benchmark results in [15], the non-probabilistic ensembles perform considerably better in NLL.[31] This may be due to the use of different model architectures, as this experiment differs to their implementation in the number of hidden layers and neurons, and the learning rate. However, it is remarkable that both the naïve bootstrap and the parametric bootstrap consistently improve the performance on NLL, compared to a non-bootstrap ensemble, suggesting that bootstrapped ensembles are more effective at assigning higher probabilities to the correct outcomes. In conclusion, the RMSE and NLL results indicate that both non-probabilistic and probabilistic approaches achieve generally comparable performance metrics, with the non-probabilistic approach exhibiting better RMSE values. This demonstrates that with the appropriate model architecture, non-probabilistic models can effectively perform on par with probabilistic ones in terms of these standard metrics.

To assess the confidence of each ensemble into its predictions, the calibration curves for all ensemble configurations are computed for the benchmark datasets. First, the prediction intervals are created using 1% steps as confidence levels z on $\mathit{Bounds} = \mathit{Mean} \pm z \times \sqrt{\mathit{Variance}}$. Then, the observed fractions are measured, and the prediction interval coverage probability (PICP) can be obtained. After gathering these values, the calibration curves can be plotted, as can be seen exemplary in Figure 8 for the *Wine Quality Red* dataset. The area between the curve and the ideal diagonal line is called the *miscalibration area* and delivers a suitable quantification for the calibration. Averaged on all runs, the miscalibration areas for all ensemble configuration across the benchmark datasets are reported in Table 3. It is important to underline that the probabilistic models appear significantly better calibrated and at times even underconfident, while the non-probabilistic models

[31] The non-probabilistic approach performs significantly better in the NLL metric than in the study of Lakshminarayanan et al. (2017) [15].

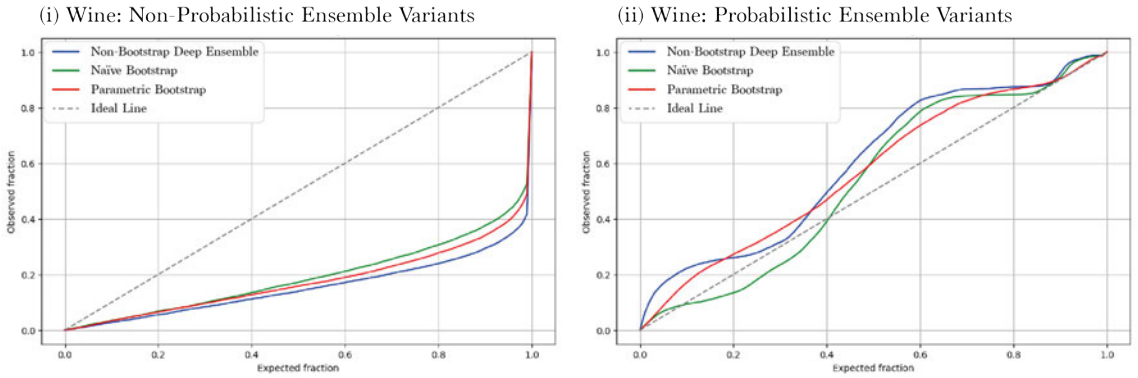

Figure 8: Calibration curves for the *Wine Quality Red* dataset, for non-probabilistic ensemble variants (i) and probabilistic ensemble variants (ii).

remain eminently overconfident, with an exception on the by far largest dataset, *Year Prediction MSD*, where the non-probabilistic ensembles appear to be well-calibrated. This supposedly better calibration of probabilistic ensembles aligns well with the findings of Lakshminarayanan et al. (2017) [15]. As the objective of this work however is to determine the effects of bootstrapping in this context, a valuable trend can be observed. Noteworthily, both in the non-probabilistic and probabilistic variants, the naïve bootstrap and the parametric bootstrap consistently achieve improvements on minimizing the miscalibration area, thus coherently improving the calibration across both ensemble types.

Calibration (Miscalibration Area)						
	Non-Probabilistic Ensembles			Probabilistic Ensembles		
Dataset	DE	NB	PB	Prob.-DE	Prob.-NB	Prob.-PB
Yacht	0.4538	0.4506	**0.4416**	*0.0896*	*0.0898*	***0.0758***
Diabetes	0.3979	**0.3440**	0.3841	0.0988	**0.0972**	0.1150
Energy	0.4678	**0.4586**	0.4650	0.1053	**0.1047**	0.1088
Concrete	0.4444	**0.4321**	0.4355	*0.0130*	***0.0124***	0.0153
Wine	0.3451	**0.3075**	0.3232	*0.0993*	***0.0680***	*0.0717*
Kin8nm	0.4263	0.4244	**0.4154**	*0.0863*	*0.1506*	***0.0795***
CCPP	0.3848	0.3823	**0.3754**	*0.0892*	***0.0441***	*0.0961*
Naval	0.3454	0.3779	**0.3334**	0.0468	**0.0323**	0.0440
Protein	0.4045	**0.3886**	0.3974	0.1070	0.1069	**0.1065**
Year MSD	**0.0984**	0.1504	0.1153	n/a	n/a	n/a

Table 3: Average miscalibration area results for different ensemble configurations across the benchmark datasets, averaged across all runs. The calibration computations used 1% intervals. Lower values indicate better calibration, the best performing configuration for each ensemble type on each dataset is highlighted. *Italic* values indicate predominant underconfidence, non-italic values predominant overconfidence.

For the further evaluation, the uncertainty estimates for all predictions on the test set can be computed by Equation 7 for the non-probabilistic deep ensembles, and Equation 10 for the probabilistic deep ensembles, respectively. With these uncertainty estimates at hand, it is now possible to examine the ability to identify out-of-distribution samples. Subsequently, a second augmented test set is created from the first. Using a noise level $\alpha = 0.8$ for $X_{OOD} = X_{test} + \alpha \times \mathcal{N}(0, \sigma^2)$, a shift in the data distribution is simulated. A 3D t-SNE visualization of test set and OOD set for the *Naval Propulsion* dataset can be seen in Figure 9. Each ensemble configuration then makes predictions on this OOD set as well, accompanied by computing the uncertainty estimates for each augmented sample. Following this, the sets are combined and sorted (or ranked) according to the uncertainty estimates, while keeping track of which sample originates from which test set class. As this is essentially a quantification of the quality of a binary classification, the *Area Under the Receiver Operating Characteristic Curve* (AUROC) can be computed.

OOD (AUROC)						
	Non-Probabilistic Ensembles			Probabilistic Ensembles		
Dataset	DE	NB	PB	Prob.-DE	Prob.-NB	Prob.-PB
Yacht	**0.6495**	0.6084	0.6162	**0.8323**	0.8282	0.7767
Diabetes	0.7101	0.7128	**0.7134**	0.7602	**0.7609**	0.7604
Energy	**0.9424**	0.8868	0.9352	0.8713	0.8711	**0.8783**
Concrete	**0.6915**	0.6858	0.6523	**0.8431**	0.8408	0.8236
Wine	**0.8063**	0.7931	0.7860	0.7342	**0.7451**	0.6897
Kin8nm	0.6751	**0.6868**	0.6760	0.6450	**0.6554**	0.6325
CCPP	0.6206	0.6409	**0.6507**	0.6389	0.6486	**0.6574**
Naval	0.9993	**0.9998**	0.9987	0.5744	**0.6065**	0.5658
Protein	**0.8968**	0.8935	0.8929	0.7533	**0.7938**	0.7446
Year MSD	0.6638	**0.7494**	0.7097	n/a	n/a	n/a

Table 4: Average AUROC results for OOD sample identification for different ensemble configurations across the benchmark datasets, averaged across all runs. Higher values indicate better OOD detection, the best performing configuration for each ensemble type on each dataset is highlighted.

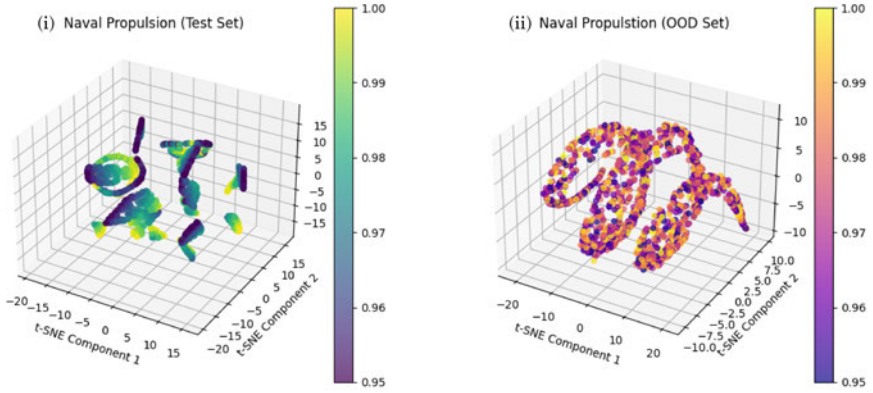

Figure 9: 3D t-SNE plot for the *Naval Propulsion* test set (i) and the equally sized augmented out-of-distribution set (ii).

The average resulting areas for all ensemble configurations are reported in Table 4. In these results, only slight variations in the AUROC scores between the ensemble configurations for each of the two ensemble architecture types can be noticed. This mixed picture indicates that the effect of bootstrapping on the detection of OOD samples could vary with factors such as dataset complexity and inherent noise in the datasets. Nonetheless, in the probabilistic approach, the naïve bootstrap and the parametric bootstrap both seem to handle the identification of OOD samples slightly better, compared to the non-bootstrap ensemble.

By now it makes sense to narrow down the scope of this work to concentrate further on the effects of bootstrapping on explained predictive uncertainty. Although Equation 11 and 12 make it possible to decompose the estimated uncertainty for probabilistic ensembles into an aleatoric and an epistemic component, it is yet unclear whether explanation techniques like LRP or GI can be applied to probabilistic deep ensembles in a straightforward way. Since they are outputting two values, the mean and the variance of a predictive distribution, and since the aforementioned explanation techniques in regression tasks typically only aim to explain the relevance of features leading to a single predicted output, it is unclear how to combine the mean and the variance in an appropriate manner, to allow for a full picture on their uncertainty explanations. Thus, we will focus more on non-probabilistic ensembling approaches for the subsequent part of this work, and the effect of the introduced sampling variability to them.

To obtain an estimate for the sampling uncertainty, the parametric bootstrap method further offers a way to directly extract this part from the epistemic component by using Equation 14. Table 5 reports these epistemic uncertainty estimates for non-probabilistic ensemble variants. The results clearly show consistently higher estimates for both the naïve bootstrap and the parametric bootstrap – indicating that the sampling variability can indeed be a substantial factor for predictive uncertainty estimation. The additionally computed sampling uncertainty component for the parametric bootstraps supports this as well, suggesting that the proportion of relevance of the sampling

Predictive Uncertainty Estimates (Average)				
	Epistemic Uncertainty			Sampling Uncertainty Component
Dataset	DE	NB	PB	PB
Yacht	2.45	2.94	2.51	0.129
Diabetes	337.79	902.64	436.75	121.91
Energy	0.27	0.64	0.31	0.081
Concrete	6.11	7.80	7.67	1.93
Wine	0.0692	0.1198	0.0982	0.0359
Kin8nm	0.0026	0.0023	0.0032	0.00091
CCPP	20.12	21.13	23.94	4.04
Naval	0.000036	0.000023	0.000041	0.000010
Protein	1.79	2.67	2.06	0.46
Year MSD	80.05	79.03	80.24	13.76

Table 5: Average epistemic uncertainty estimates for different ensemble configurations (non-probabilistic approach) across the benchmark datasets, including the sampling uncertainty component for the parametric bootstrap deep ensemble (PB).

uncertainty can be substantial, but depends on the individual sampling situation of each dataset.

4.2 Experiment 2: Synthetic Dataset

To further investigate these preliminary insights of the real-world regression benchmark experiment, it appears consequential to isolate the effects of bootstrapping deep ensembles in a more controlled environment. Thus, this second experiment applies the same experimental setup from the previous experiment to a simplified synthetic toy dataset, aiming to verify the third hypothesis. This dataset is specifically designed to isolate the effects on explained predictive uncertainty, with known insights about the true relationship.

4.2.1 Experimental Setup

The experiment utilizes the same ensemble architectures and training procedures as the previously described benchmark experiment. However, for the dataset it creates a controlled scenario. This toy dataset has $n = 5000$ samples with $d = 10$ features and associated targets y. All input features for each sample are generated from a standard Gaussian normal distribution $x_i \sim \mathcal{N}(0, 1)$. The initial target y is defined as $y = X \cdot \beta + \epsilon$, where β is a coefficient vector of ones (i.e., $\beta = \mathbf{1}_d$), and $\epsilon \sim \mathcal{N}(0, 0.1^2)$ represents Gaussian noise. The target y is then further adjusted by adding heteroscedastic noise, dependent on selected features. More precisely, a single feature noise ϵ_{single} and a correlated feature noise $\epsilon_{\text{correlated}}$ are added, where $\epsilon_{\text{single}} \sim \mathcal{N}(0, 0.8 \times |\text{feature}_1|)$ and $\epsilon_{\text{correlated}} \sim \mathcal{N}(0, 0.8 \times |\text{feature}_3| \times |\text{feature}_8|)$. Taken together, this leads to a composed target y, defined as:

$$y = X \cdot \mathbf{1}_d + \epsilon + \epsilon_{\text{single}} + \epsilon_{\text{correlated}}$$

The motivation for this is to assess how well the ensemble configurations are able to detect these noisy features as the causes for uncertain predictions. Subsequently, the explanations for the top $k = 5\%$ most uncertain predictions are most vital. The method of Bley et al. (2024) [1] allows for a detailed analysis and visual interpretation of single feature relevances and correlated explanations. They recommend using their method with LRP explanations (*CovLRP*).[32] For this, the feature relevances for each sample and within each base learner are obtained utilizing relevance propagation by the generalized LRP-γ rule, before they are averaged across the ensemble [1] [36]. Then, the covariance matrix for each sample is computed. The matrices for the top 5% are then averaged across the 20 runs to ascertain a mean matrix for the most uncertain samples, to be able to analyze the feature contributions to these uncertain predictions in detail. The diagonal elements of the matrix represent the contributions of each feature, while the off-diagonal elements reports the extend of the correlation between two. A positive covariance indicates a correlated relationship, a value closer to 0 indicates a smaller relationship, and a negative covariance indicates an inverse[33] relationship.

4.2.2 Experimental Results

As seen in Figure 10, the resulting covariance matrices can be nicely visualized as heatmaps, where red elements indicate positive and blue elements indicate negative covariance. The only ensemble

[32] Different explanation methods can also be utilized in this step. The use of Gradients × Inputs leads to similar results in this experiment, as can be seen in Appendix A.2.1. However, LRP typically results in more trustworthy explanations, as it is not influenced by the *shattered gradients* phenomenon.

[33] The correlation occurs when one feature has high values and the other has low values.

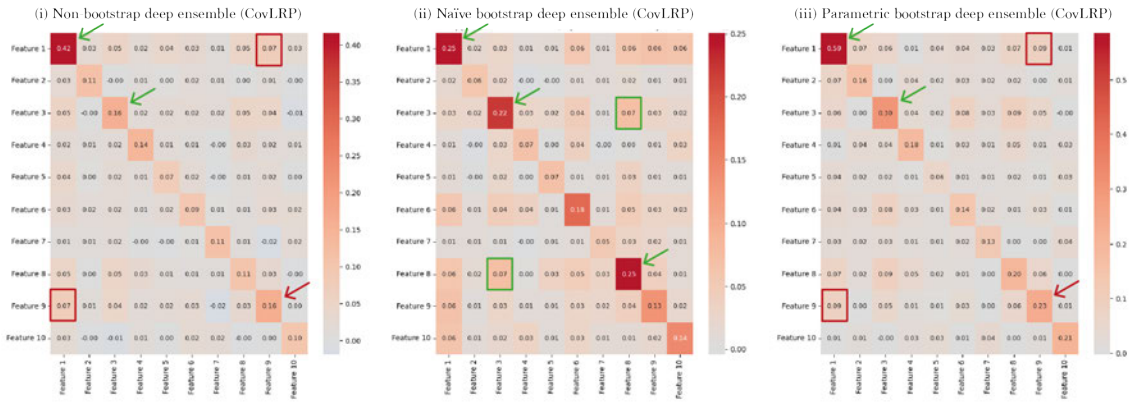

Figure 10: Covariance Matrices for the top 5% most uncertain samples in the synthetic dataset experiment ($n = 5000$) using *Cov LRP* explanations. The top three assigned relevance scores are marked with arrows, the highest off-diagonal element is highlighted with a square. Green color indicates correct, red color indicates wrong assignment. The only configuration to identify all noisy features and the feature correlation correctly is (ii) the naïve bootstrap deep ensemble.

configuration that was able to identify the three noisy features correctly as the top most relevant single features for the 5% most uncertain samples was the naïve bootstrap deep ensemble. All other configurations incorrectly assigned at least one higher relevance score to spurious features. Furthermore, only the naïve bootstrap deep ensemble was able to accurately assign the highest correlation relevance to the correlation of the correct feature_3 and feature_8 However, it is important to stress that these results represent the effect on a simplified toy dataset and might not be directly applicable to more complex real-world scenarios.

Further interesting insights can be obtained when we take a closer look on the qualitative characteristics of the matrices in Figure 10. If we analyze the ratio of the sum of the absolute values of the diagonal elements to the sum of the absolute values of all elements, we can get an understanding whether the ensemble configuration assigned proportionally more relevance to single feature contributions than to inter-feature codependency, or vice-versa. This could be particularly interesting for determining if there are differences in the focus of relevance among the ensemble configurations. The ratio can be calculated as in Equation 19, where a_{ij} denotes the i-th covariance matrix element in the j-th row:

$$\text{Ratio}_{diag} = \frac{\sum_{i=1}^{n} |a_{ii}|}{\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|} \tag{19}$$

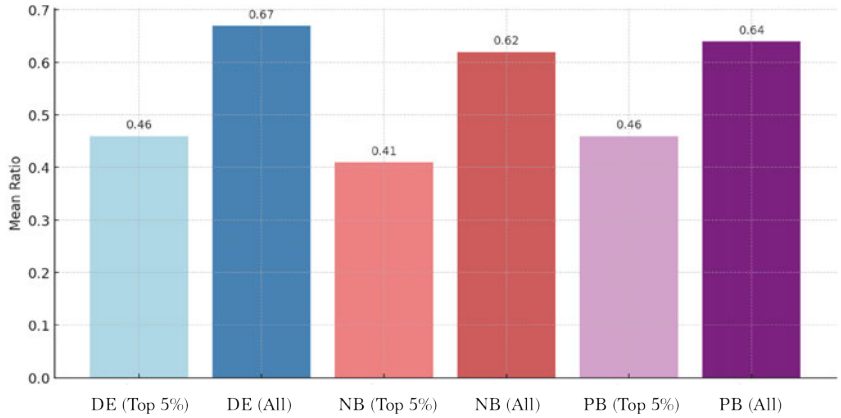

Figure 11: Ratios on the CovLRP matrices for different ensemble configurations (non-probabilistic approach) on all samples of the test set and the top 5% most uncertain samples only. Denotations are non-bootstrap deep ensemble (DE), naïve bootstrap deep ensemble (NB), and parametric bootstrap deep ensemble (PB)

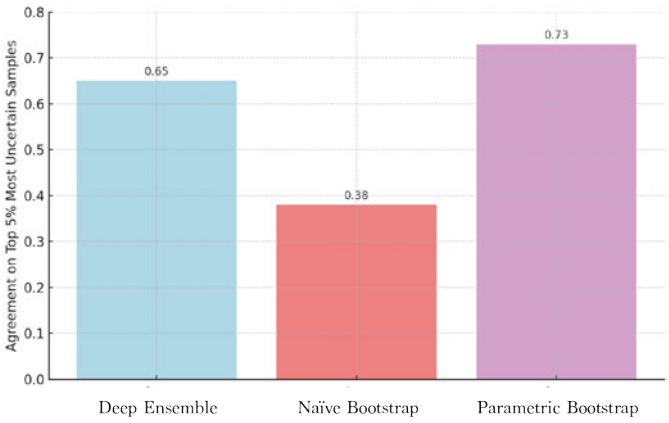

Figure 12: Agreement of the different ensemble configurations (non-probabilistic approach) with each other on the top 5% most uncertain samples.

Figure 11 reports these ratios for this toy dataset experiment. Two insights can be extracted from these results. Firstly, the proportion of the diagonal elements in the matrices for the top 5% most uncertain samples is consistently lower than for the matrices of the whole test set, underlining the role of second-order effects for uncertain predictions in neural networks. Secondly, as the ratio values for naïve bootstrap deep ensembles are lower than for the other configurations, it appears that the naïve bootstrap results in less proportional relevance of the diagonal elements, and thus emphasizes the role of inter-feature correlation. Furthermore, a reason for the different quality of explanation results may be traced back to the different choice of top uncertain samples of the ensemble configurations.

Figure 12 confirms that the naïve bootstrap assigns the highest uncertainty estimates to different samples than the other ensemble configurations, and agrees on less than half of the samples that the others choose. Additionally, Figure 13 also visualizes this in exemplary t-SNE plots on the assignment of uncertainty estimates among the test set samples. As the naïve bootstrap was the only configuration to correctly identify the relevant sources of uncertainty in this toy example, this disagreement on the top samples may suggest a more accurate predictive uncertainty estimation, leading to the useful explanation results in this experiment.

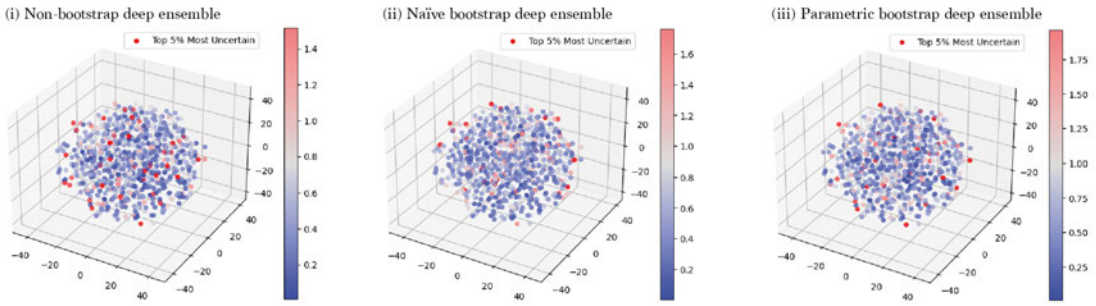

Figure 13: Exemplary 3D t-SNE plots for the assigned uncertainty estimates on the test set samples, among the ensemble configurations (non-probabilistic approach): Non-bootstrap deep ensemble (i), naïve bootstrap (ii) and parametric bootstrap (iii). The top 5% most uncertain samples for each configuration are highlighted in red.

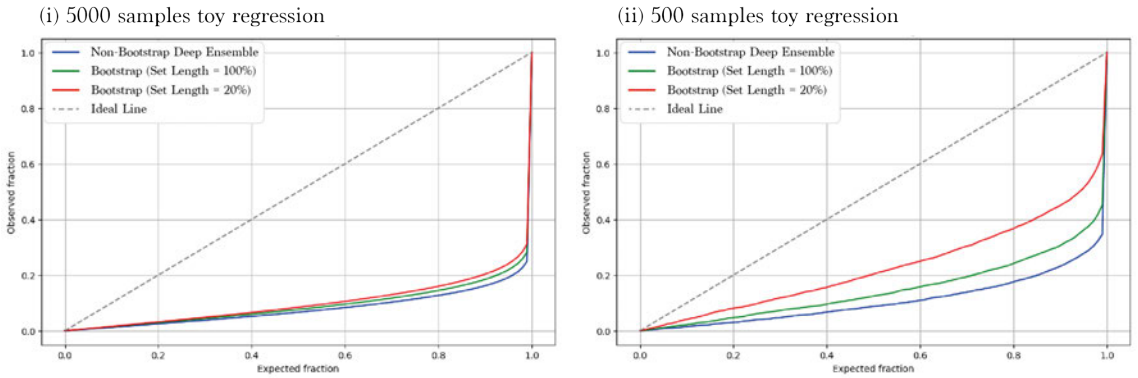

Figure 14: Covariance curves for non-probabilistic configurations without bootstrap (blue), with naïve bootstrap on the full subset length (green), and with naïve bootstrap on 20% subset length (red); on a (i) 5000 sample toy regression dataset and a (ii) 500 sample toy regression dataset.

Interestingly, another phenomenon can be observed, when the initial synthetic toy dataset experiment is repeated for a smaller, other than that equally set up toy dataset consisting of $n = 500$ samples instead of $n = 5000$. This additional study aims to investigate the effect of bootstrapping on ensemble calibration, with a particular focus on dataset size, to provide further insights into the second hypothesis. In Figure 14, the resulting calibration curves can be seen. The used configurations here are a non-bootstrap ensemble, straightforward naïve bootstrap, and a naïve bootstrap using only a subset size of 20% of the original training set length, thus increasing the effect of bootstrap variability drastically. In comparing the curves of the smaller ($n = 500$) and the larger ($n = 5000$) dataset, it can be observed that the effect of bootstrapping appears much more pronounced in a smaller dataset scenario. However, it is worth mentioning that the stronger effect on the calibration of the second 20% bootstrap do not translate to qualitatively better explanations, indicating an exceeded threshold in the trade-off of the introduced variability by bootstrapping. Further results for this can be found in Appendix A.2.2.

5 Discussion

As this experimental analysis demonstrates, the introduced variability by bootstrapping deep ensembles has various notable effects on the ensembles and does affect explained predictive uncertainty in potentially significant ways. The set of two experiments provided insightful perspectives on the research hypotheses to be discussed.

In the first experiment on the regression benchmark datasets, both the naïve bootstrap and the parametric bootstrap yielded consistently higher estimates for epistemic uncertainty as a result of the variability in the training sets and subsequently the increased model diversity. These estimates indicate that depending on the dataset, the sampling uncertainty due to the finite number of samples in the dataset can indeed play a substantial role in the overall uncertainty of ensemble predictions. The results therefore support the first research hypothesis, as measures on negative log-likelihood in the benchmark experiment are also generally improved by bootstrapping. As shown in the second experiment, particularly for the naïve bootstrap this seems to manifest in the identification of different samples as the most uncertain predictions in a test set.[34] Thus, the increased model diversity might help a bootstrapped ensemble in better identifying regions of the input space where the model's predictions are highly uncertain. This might in fact lead to practical use cases, as the insights about such areas could then be utilized to improve a dataset accordingly, e.g. by obtaining more samples in that specific region. Hence, one could reduce sources of epistemic uncertainty when harnessing the effect of bootstrapping on deep ensembles.

Additionally, both the results in the benchmark experiment and the results in the toy dataset experiment suggest that bootstrapping does generally lead to improved results on trustworthiness metrics like calibration, confirming the second hypothesis. With a change in the dataset size from 5000 to 500 samples, the second experiment was able to demonstrate more pronounced effects on a smaller dataset. However, these effect may depend on several additional factors, such as the complexity of the dataset, its inherent structure, various tunable hyperparameters, and the initially chosen underlying model architecture. To narrow down the scope of this work, only some of these factors have been examined thoroughly, and particularly the effect of different model architectures could be an interesting area of further investigation.

Furthermore, this exploratory research analyzed the qualitative effects on explained predictive uncertainty by bootstrapping deep ensembles in the non-probabilistic approach. The interplay between different sources of uncertainty remains an interesting aspect. By extending explanation techniques like LRP to probabilistic neural networks, one could widen the scope of these experiments and obtain a better decomposition of predictive uncertainty across input and target ranges, which could lead to more nuanced results and further practical use cases. However, the second experiment demonstrated that the epistemic uncertainty estimates in non-probabilistic ensembles are in fact influenced by features with strong aleatoric noise, and thus capture the aleatoric uncertainty indirectly as well.

The second experiment was also able to deliver further qualitative insights into the effect of bootstrapped deep ensembles on the explanations for predictive uncertainty. By training on multiple resampled datasets, the second experiment showed that bootstrapped ensembles can mitigate the influence of spurious correlations that might appear in individual training subsets and focus on the relevant noisy features, thus affirming the third hypothesis. In non-bootstrapped ensembles, certain spurious features might consistently receive high relevance if they coincidentally correlate with the output in the full training set. Bootstrapping introduces a form of randomness that can help to average out the effect of these spurious features across different models, potentially leading to the more robust explanations in the toy dataset experiment.

This exploratory study primarily concentrated on the effect of the naïve bootstrap on explained predictive uncertainty. Nonetheless, as could be seen by the experimental results, the parametric bootstrap also yields promising results, and has the benefits of being able to directly quantify the sampling uncertainty estimates. Thus, it constitutes a solid approach to combining advantages from both non-bootstrap and naïve bootstrap ensembles. This raises the question if there are other variations or mechanisms of bootstrapping that could introduce sampling variability into the ensemble members and lead to comparable or improved qualitative results. An interesting idea to investigate further could be the division of training samples into clusters, or intervals based on the

[34] Appendix A.1.1 shows this exemplary for a real-world dataset (*Naval Propulsion*) as well.

input or target ranges, which could lead to individual base learners being experts in certain areas of the data space. Particularly for more practical use cases, another concept to explore could be hybrid approaches, where some base learners are bound to some bootstrapping mechanism and others are not, potentially further incorporating the benefits of both approaches.

6 Conclusion

This thesis explored the effects of bootstrapping deep ensembles on predictive uncertainty and its explanations by an analysis over a set of two experiments. The experimental results demonstrated the potential of introducing sampling variability by bootstrapping into the training process of deep ensembles. This enhanced diversity in the models allows bootstrapped deep ensembles to obtain potentially better estimates of predictive uncertainty, enhances trustworthiness metrics such as calibration and is able to deliver more insightful explanations in certain scenarios.

Overall, the results supported the three proposed research hypotheses. However, it can be noted that the findings highlighted several factors – such as dataset size, complexity, and model architecture – that all influence the impact of bootstrapping on deep ensembles and its quality of explanations. Further research could therefore help in comprehending these effects and obtaining more meaningful generalizations.

Ultimately, the utilization of bootstrapping deep ensembles greatly depends on the use case. As discussed, bootstrapping can be instrumental in identifying noisy features and areas of uncertainty within a dataset. It can potentially provide better and more reliable estimates for predictive uncertainty, as well as more insightful explanations of it. Thus, it can help in determining areas where further or better training data might be needed. However, the effects of bootstrapping appear to be contingent on factors such as the dataset size or complexity, hyperparameters and model architecture. Therefore, bootstrapping should be applied with careful consideration of these factors in mind.

7 References

[1] F. Bley, S. Lapuschkin, W. Samek, and G. Montavon, *Explaining Predictive Uncertainty by Exposing Second-Order Effects*, en, arXiv:2401.17441 [cs, stat], Jan. 2024. [Online]. Available: http://arxiv.org/abs/2401.17441 (visited on 07/27/2024).

[2] R. Goebel, A. Chander, K. Holzinger, *et al.*, "Explainable AI: The New 42?" en, in *Machine Learning and Knowledge Extraction*, A. Holzinger, P. Kieseberg, A. M. Tjoa, and E. Weippl, Eds., vol. 11015, Series Title: Lecture Notes in Computer Science, Cham: Springer International Publishing, 2018, pp. 295–303, ISBN: 978-3-319-99739-1 978-3-319-99740-7. DOI: 10.1007/978-3-319-99740-7_21. [Online]. Available: https://link.springer.com/10.1007/978-3-319-99740-7_21 (visited on 08/02/2024).

[3] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications," en, *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, Mar. 2021, arXiv:2003.07631 [cs, stat], ISSN: 0018-9219, 1558-2256. DOI: 10.1109/JPROC.2021.3060483. [Online]. Available: http://arxiv.org/abs/2003.07631 (visited on 07/28/2024).

[4] J. Nixon, D. Tran, and B. Lakshminarayanan, "Why Aren't Bootstrapped Neural Networks Better?" en, 2020.

[5] L. Sluijterman, E. Cator, and T. Heskes, *Confident Neural Network Regression with Bootstrapped Deep Ensembles*, en, arXiv:2202.10903 [cs, stat], Aug. 2023. [Online]. Available: http://arxiv.org/abs/2202.10903 (visited on 07/27/2024).

[6] L. Breiman, "Bagging predictors," en, *Machine Learning*, vol. 24, no. 2, pp. 123–140, Aug. 1996, ISSN: 0885-6125, 1573-0565. DOI: 10.1007/BF00058655. [Online]. Available: http://link.springer.com/10.1007/BF00058655 (visited on 07/29/2024).

[7] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods," en, *Machine Learning*, vol. 110, no. 3, pp. 457–506, Mar. 2021, ISSN: 0885-6125, 1573-0565. DOI: 10.1007/s10994-021-05946-3. [Online]. Available: https://link.springer.com/10.1007/s10994-021-05946-3 (visited on 07/27/2024).

[8] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? Does it matter?" en, *Structural Safety*, 2009.

[9] M. Abdar, F. Pourpanah, S. Hussain, *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," en, *Information Fusion*, vol. 76, pp. 243–297, Dec. 2021, ISSN: 15662535. DOI: 10.1016/j.inffus.2021.05.008. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1566253521001081 (visited on 07/27/2024).

[10] A. Kendall and Y. Gal, *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* en, arXiv:1703.04977 [cs], Oct. 2017. [Online]. Available: http://arxiv.org/abs/1703.04977 (visited on 07/27/2024).

[11] J. Davis, J. Zhu, J. Oldfather, S. MacDonald, M. Trzaskowski, and M. Kelsen, "AWS Prescriptive Guidance - Quantifying uncertainty in deep learning systems," en, 2020.

[12] Eric Smith, *Uncertainty Analysis, in Encyclopedia of Environmetrics*, en, A. H. El-Shaarawi and W. W. Piegorsch, Eds. Chichester ; New York: Wiley, 2002, ISBN: 978-0-471-89997-6.

[13] R. M. Neal, *Bayesian Learning for Neural Networks* (Lecture Notes in Statistics), en, P. Bickel, P. Diggle, S. Fienberg, *et al.*, Eds. New York, NY: Springer New York, 1996, vol. 118, ISBN: 978-0-387-94724-2 978-1-4612-0745-0. DOI: 10.1007/978-1-4612-0745-0. [Online]. Available: http://link.springer.com/10.1007/978-1-4612-0745-0 (visited on 07/28/2024).

[14] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, "Hands-on Bayesian Neural Networks – a Tutorial for Deep Learning Users," en, *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 29–48, May 2022, arXiv:2007.06823 [cs, stat], ISSN: 1556-603X, 1556-6048. DOI: 10.1109/MCI.2022.3155327. [Online]. Available: http://arxiv.org/abs/2007.06823 (visited on 07/28/2024).

[15] B. Lakshminarayanan, A. Pritzel, and C. Blundell, *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*, en, arXiv:1612.01474 [cs, stat], Nov. 2017. [Online]. Available: http://arxiv.org/abs/1612.01474 (visited on 07/27/2024).

[16] K. Bykov, M. M.-C. Höhne, A. Creosteanu, *et al.*, *Explaining Bayesian Neural Networks*, en, arXiv:2108.10346 [cs, stat], Aug. 2021. [Online]. Available: http://arxiv.org/abs/2108.10346 (visited on 07/28/2024).

[17] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," en, *ICML*, 2016.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," en, 2014.

[19] S. Fort, H. Hu, and B. Lakshminarayanan, *Deep Ensembles: A Loss Landscape Perspective*, en, arXiv:1912.02757 [cs, stat], Jun. 2020. [Online]. Available: http://arxiv.org/abs/1912.02757 (visited on 07/27/2024).

[20] A. G. Wilson and P. Izmailov, *Bayesian Deep Learning and a Probabilistic Perspective of Generalization*, en, arXiv:2002.08791 [cs, stat], Mar. 2022. [Online]. Available: http://arxiv.org/abs/2002.08791 (visited on 08/02/2024).

[21] Y. Ovadia, E. Fertig, J. Ren, *et al.*, *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift*, en, arXiv:1906.02530 [cs, stat], Dec. 2019. [Online]. Available: http://arxiv.org/abs/1906.02530 (visited on 07/29/2024).

[22] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, *Pitfalls of In-Domain Uncertainty Estimation and Ensembling in Deep Learning*, en, arXiv:2002.06470 [cs, stat], Jul. 2021. [Online]. Available: http://arxiv.org/abs/2002.06470 (visited on 08/04/2024).

[23] R. Heijungs, *Probability, statistics and life cycle assessment: guidance for dealing with uncertainty and sensitivity*, en. Cham, Switzerland: Springer, 2024, OCLC: 1435755015, ISBN: 978-3-031-49317-1.

[24] B. Efron, "The Jackknife, the Bootstrap and other Resampling Plans," *SIAM*, 1982.

[25] T. Gneiting, F. Balabdaoui, and A. E. Raftery, "Probabilistic forecasts, calibration and sharpness," en, 2007.

[26] L. Sluijterman, E. Cator, and T. Heskes, "How to Evaluate Uncertainty Estimates in Machine Learning for Regression?" en, *Neural Networks*, vol. 173, p. 106 203, May 2024, arXiv:2106.03395 [cs, stat], ISSN: 08936080. DOI: 10.1016/j.neunet.2024.106203. [Online]. Available: http://arxiv.org/abs/2106.03395 (visited on 08/04/2024).

[27] Y. Wen, G. Jerfel, R. Muller, *et al.*, *Combining Ensembles and Data Augmentation can Harm your Calibration*, en, arXiv:2010.09875 [cs, stat], Mar. 2021. [Online]. Available: http://arxiv.org/abs/2010.09875 (visited on 08/17/2024).

[28] Y. Chung, I. Char, H. Guo, J. Schneider, and W. Neiswanger, *Uncertainty Toolbox: An Open-Source Library for Assessing, Visualizing, and Improving Uncertainty Quantification*, en, arXiv:2109.10254 [cs, stat], Sep. 2021. [Online]. Available: http://arxiv.org/abs/2109.10254 (visited on 08/16/2024).

[29] M. Pakdaman Naeini, G. Cooper, and M. Hauskrecht, "Obtaining Well Calibrated Probabilities Using Bayesian Binning," en, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, Feb. 2015, ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v29i1.9602. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/9602 (visited on 08/20/2024).

[30] T. Gneiting and A. E. Raftery, "Strictly Proper Scoring Rules, Prediction, and Estimation," en, *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, Mar. 2007, ISSN: 0162-1459, 1537-274X. DOI: 10.1198/016214506000001437. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1198/016214506000001437 (visited on 08/20/2024).

[31] G. W. Brier, "Verification of Forecasts Expressed in Terms of Probability," en, *Monthly Weather Review*, no. Volume 78, 1, 1950.

[32] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, and K. Hansen, "How to Explain Individual Classification Decisions," en, 2010.

[33] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-Wise Relevance Propagation: An Overview," en, in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, Eds., vol. 11700, Series Title: Lecture Notes in Computer Science, Cham: Springer International Publishing, 2019, pp. 193–209, ISBN: 978-3-030-28953-9 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_10. [Online]. Available: http://link.springer.com/10.1007/978-3-030-28954-6_10 (visited on 08/13/2024).

[34] D. Balduzzi, M. Frean, L. Leary, J. P. Lewis, K. W.-D. Ma, and B. McWilliams, *The Shattered Gradients Problem: If resnets are the answer, then what is the question?* en, arXiv:1702.08591 [cs, stat], Jun. 2018. [Online]. Available: http://arxiv.org/abs/1702.08591 (visited on 08/25/2024).

[35] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation," en, *PLOS ONE*, vol. 10, no. 7, O. D. Suarez, Ed., e0130140, Jul. 2015, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0130140. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0130140 (visited on 07/28/2024).

[36] P. Keyl, M. Bockmayr, D. Heim, *et al.*, "Patient-level proteomic network prediction by explainable artificial intelligence," en, *npj Precision Oncology*, vol. 6, no. 1, p. 35, Jun. 2022, ISSN: 2397-768X. DOI: 10.1038/s41698-022-00278-4. [Online]. Available: https://www.nature.com/articles/s41698-022-00278-4 (visited on 08/14/2024).

[37] J. M. Hernández-Lobato and R. P. Adams, *Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks*, en, arXiv:1502.05336 [stat], Jul. 2015. [Online]. Available: http://arxiv.org/abs/1502.05336 (visited on 08/13/2024).

[38] Q. Liu and D. Wang, *Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm*, en, arXiv:1608.04471 [cs, stat], Sep. 2019. [Online]. Available: http://arxiv.org/abs/1608.04471 (visited on 08/21/2024).

[39] D. E. Lee, *The Gross Racism in The Boston Housing Dataset and the Bias Behind 'B' and 'LSTAT' Features*, en, Jan. 2024. [Online]. Available: https://drlee.io/the-gross-racism-in-the-boston-housing-dataset-and-the-bias-behind-b-and-lstat-features-a9bf1a184904 (visited on 08/21/2024).

A Appendix

A.1 Additional Results for Experiment 1: Benchmark

A.1.1 Exemplary t-SNE Plots for the *Naval Propulsion* Test Set

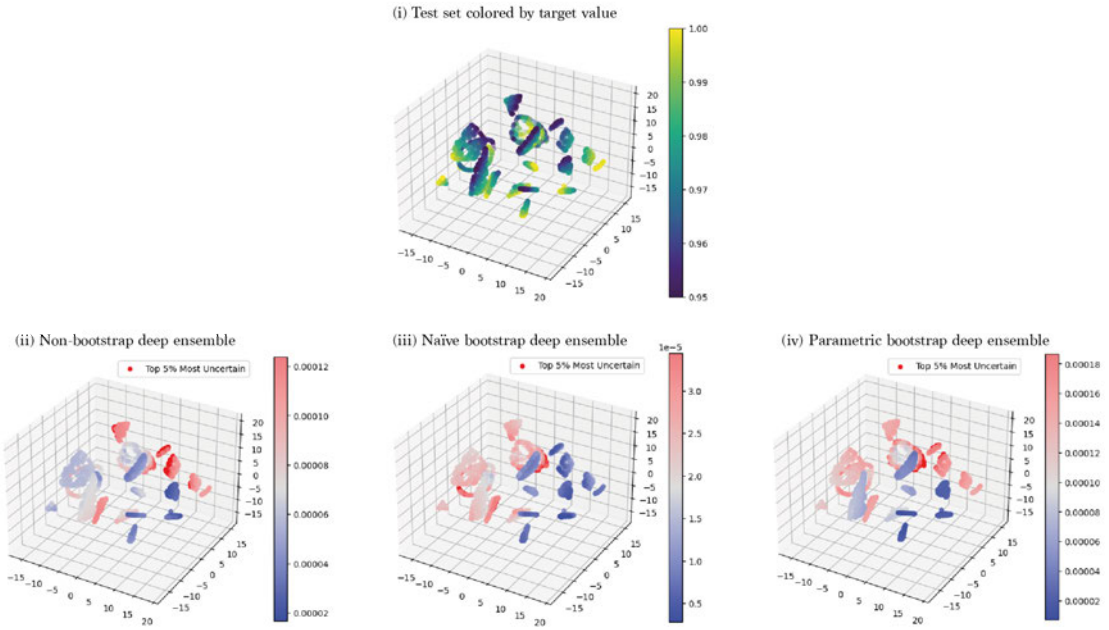

Figure 15: 3D t-SNE plots for the *Naval Propulsion* test set. (i) shows the test set colored by y values. (ii)-(iv) show the assigned uncertainty estimates for the different ensemble configurations (non-parametric approach) on the test set samples. The top 5% most uncertain samples for each configuration are highlighted in red.

These additional plots demonstrate that the ensemble configurations assign higher uncertainty estimates to different areas and clusters in the *Naval Propulsion* test set. This underlines that the increased model diversity in bootstrapped ensembles leads to different selections of top uncertain samples and a different overall understanding of uncertainty in the test set. In this example, the parametric bootstrap appears to combine highlighted uncertain areas from the non-bootstrap and the naïve bootstrap deep ensemble in its selection.

A.2 Additional Results for Experiment 2: Synthetic Dataset

A.2.1 Covariance Matrices for Gradients × Input explanations (CovGI)

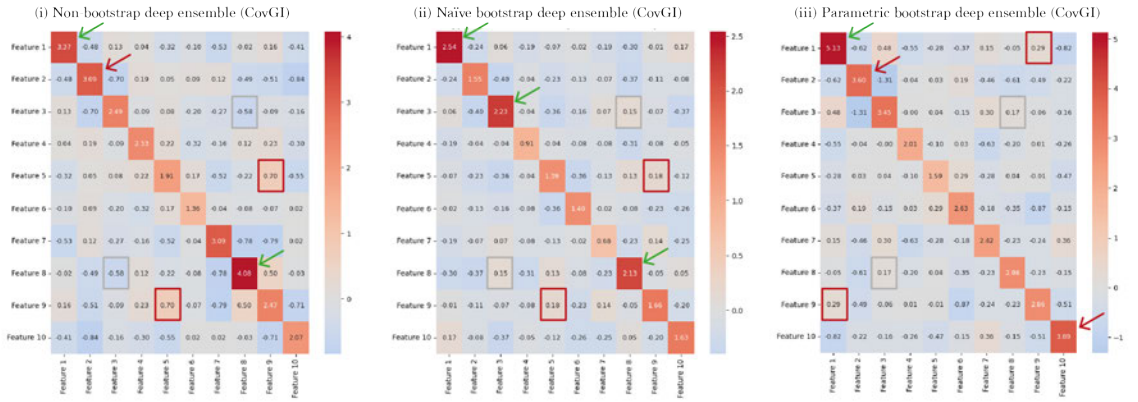

Figure 16: Covariance matrices for *Experiment 2* using Gradient × Input explanations (*CovGI*) for the top 5% most uncertain samples for each configuration on the 5000 sample toy dataset, for non-bootstrap deep ensemble (i), naïve bootstrap deep ensemble (ii) and parametric bootstrap deep ensemble (iii).

These additional results (Figure 16) demonstrate the use of Gradient × Input explanations instead of LRP for the second experiment. It can be seen that the naïve bootstrap deep ensemble (non-probabilistic approach) correctly identifies all features with heteroscedastic noise and are the only configuration to do so. In this case, none of the ensembles identified the designed correlation correctly. However, Gradient × Input can be affected by the *shattered gradients* phenomenon, which is why the LRP results are more trustworthy.

A.2.2 Covariance Matrices for Naïve Bootstrap, Subset Size 20%

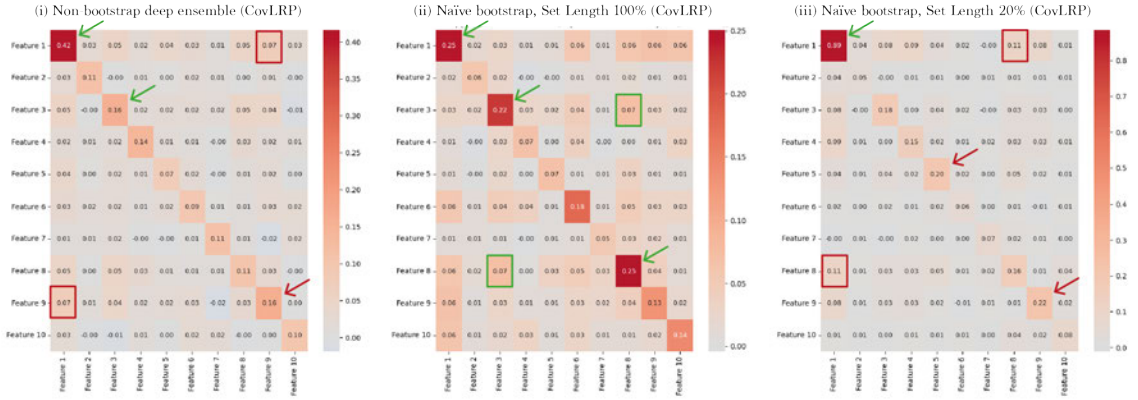

Figure 17: Covariance matrices for *Experiment 2* using LRP explanations (*CovLRP*) for the top 5% most uncertain samples for each configuration on the 5000 sample toy dataset, for non-bootstrap deep ensemble (i), naïve bootstrap deep ensemble on 100% subset size (ii) and naïve bootstrap deep ensemble on 20% subset size (iii).

The results in Figure 17 show the covariance matrix for a non-probabilistic deep ensemble trained with naïve bootstrap on 20% subset size. The resulting relevances indicate that the variability introduced by this extended bootstrap leads to poorer quality explanations, compared to the naïve bootstrap on 100% subset size, as the ensemble only correctly recognizes one of the three features with additional heteroscedastic noise, and does not accurately identify the designed inter-feature correlation.