Proceedings of the 6th Symposium on Advances in Approximate Bayesian Inference, 2024 1–24

Implicitly Bayesian Prediction Rules in Deep Learning

Anonymous Authors
Anonymous Institution

Abstract

The Bayesian approach leads to coherent updates of predictions under new data, which makes adhering to Bayesian principles appealing in decision-making contexts. Traditionally, integrating Bayesian principles in complex models like deep learning involves setting priors and approximating posteriors, despite the lack of direct parameter interpretation. In this paper, we rethink this approach and consider what *characterises* a Bayesian prediction rule. Algorithms meeting these criteria can be deemed *implicitly* Bayesian — they make the same predictions as *some* Bayesian model, without explicitly manifesting priors and posteriors. We propose how to evaluate a prediction rule's proximity to implicit Bayesianism, introduce results illustrating its benefits, and empirically test it across multiple prediction strategies.

1. Introduction

In the Bayesian framework, model predictions are updated coherently and rationally based on new evidence. These rationality properties are epitomised by various theorems showing that a Bayesian agent states and updates their beliefs in a way that cannot be trivially exploited by an adversary (Pettigrew, 2020; Lane and Sudderth, 1984, §4). This arguably gives Bayesian predictions some level of credibility; e.g. when following the Bayesian approach exactly, one does not have to worry (as much) about whether some evidence is being given more weight than other, or whether some evidence is ignored outright. Even putting computational and approximate inference considerations aside, a Bayesian still has plenty to worry about with regards to modelling choices; nonetheless, at the very least, some fundamental sanity checks are taken care of.

This is in contrast to many other — potentially more "black-box" — approaches to prediction, such as training a deep learning model. For example, when continually updating a deep learning model with new data, we might have to worry about catastrophic forgetting (McCloskey and Cohen, 1989), or that the order the data is presented in might affect the predictions (Ash and Adams, 2020).

When a model makes predictions that will be acted upon in a closed decision-making loop, e.g. when finding the optimum of an unknown function (Bayesian Optimisation) or for exploration in reinforcement learning, it is important that the model's predictions change in a coherent way in light of new observed. In these settings, the uncertainty estimates provided by the model are used to guide the decision-making process, often in a way that has to balance exploration against exploitation – the agent has to decide whether it's worth it to pursue exploring an unknown region of the space in order to improve future model's predictions. However, if the model's current predictions and uncertainty estimates are not related to how the model's predictions will change upon observing a currently unobserved variable, then any down-stream decisions made on the basis of these predictions might be highly suboptimal.

This, among other reasons, is often the reason *Bayesian* methods are being advocated for, and why many attempts have been made to incorporate Bayesian principles into deep learning. However, despite considerable effort by a large community, existing approaches for incorporating approximate Bayesian inference into deep learning often perform poorly in practical settings. Current research typically focuses on one specific model class, which is obtained by placing a prior over the parameters of the neural network, and approximating the posterior.[1] Algorithms relying on this perspective usually encumber scalability. For example, in Markov Chain Monte Carlo (MCMC) schemes, effectively, multiple models need to be trained following a cumbersome procedure for an accurate approximation; variational inference schemes, on the other hand, struggle to fit complex posteriors over the parameters without similarly complex and difficult to train models to approximate that posterior.

In this paper, we advocate for taking a different perspective on how to incorporate Bayesian principles into deep learning, and how to even think about measuring how close we are to achieving this goal. Specifically, we look at what desirable properties predictions made by Bayesian methods posses (including a new Dutch-book argument in Section 3), what properties *characterise* a prediction rule as being Bayesian (Section 2.3), and how to measure how close a prediction rule is to having these properties (Section 4). We also empirically investigate how various design decisions can affect how close an algorithm is to being Bayesian on a small regression task (Section 4.2). We advocate that this might be a more fruitful way to think about incorporating Bayesian principles into deep learning, as it only dictates a minimal set of conditions for how the predictions should behave, rather than dictating how the internals of the prediction algorithm should be structured.

2. Background

Whereas statistics often deals more broadly with inferences about various unobserved quantities, the prediction of future observations is arguably the centerpiece of machine learning. Hence, in this piece, we primarily consider the setting of predicting future observations given the past. Concretely, given a sequence of random observations $X_1, X_2, \dots$ we are interested in predicting the values of $X_{n+1}, X_{n+2}, \dots$ given observations of $X_1, \dots, X_n$ for different n. We'll also look at the case of regression/classification where for a sequence of random variables $X_1, X_2, \dots$ and $Y_1, Y_2, \dots$ we are interested in predicting the value of Y_{n+1} given the observations of $X_1, Y_1, \dots, X_n, Y_n$, and X_{n+1}.

To discuss and compare the properties of various approaches to predicting future observations, it is helpful to introduce the concept of a *prediction rule/strategy*. If the observations take values in some space $\mathcal{X}$, then (informally) a prediction rule is a sequence of functions $(s_0, s_1, s_2, \dots)$ where each s_n maps a sequence $(x_1, \dots, x_n) \in \mathcal{X}^n$ to a probability distribution $s_n(\cdot|x_1, \dots, x_n)$. $s_n(\cdot|x_1, \dots, x_n)$ carries the interpretation of the prediction for the next observation X_{n+1} given the observed outcomes $(x_1, \dots, x_n)$ for the previous observations $X_1, \dots, X_n$.

More precisely, Dubins et al. (2014) formally introduce a prediction rule/strategy on a measurable space $(\mathcal{X}, \mathcal{F})$ as a sequence of functions $s_n : \mathcal{F} \times \mathcal{X}^n \to [0, 1]$ where:

1. Function-space variational inference (Sun et al., 2019) being a notable exception, although in this case a prior is still specified explicitly, just directly in the function space.

1. For every fixed $(x_1, \dots, x_n) \in \mathcal{X}^n$, the function $A \mapsto s_n(A|x_1, \dots, x_n)$ for $A \in \mathcal{F}$ is a probability measure on $(\mathcal{X}, \mathcal{F})$,
2. For every fixed $A \in \mathcal{F}$, the function $x_1, \dots, x_{n-1} \mapsto s_n(A|x_1, \dots, x_{n-1})$ is $\otimes_{i=1}^n \mathcal{F}$-measurable (with $\otimes_{i=1}^n \mathcal{F}$ denoting the product σ-algebra on $\mathcal{X}^n$)

These two conditions equivalently specify that each s_k is a Markov kernel from $(\mathcal{X}^k, \otimes_{i=1}^k \mathcal{F})$ to $(\mathcal{X}, \mathcal{F})$, effectively ensuring that the prediction rules define a joint probability measure on $(\mathcal{X}^n, \otimes_{i=1}^n \mathcal{F})$, i.e. on the sequence of first n observations, for all n. Furthermore, by the *Ionescu-Tulcea Theorem* (Hoffman-Jorgensen, 2017; Berti et al., 2023), a prediction rule uniquelly defines a probability measure over the whole infinite sequence space $(\mathcal{X}^n, \otimes_{i=1}^\infty \mathcal{F})$, hence formally justifying using a prediction rule to make predictions on the whole sequence of observations $(X_1, X_2, \dots)$; it also allows us to define properties of prediction rules in terms of the joint they imply over the sequence space.

In a large proportion of cases, when dealing with a machine learning problem, we're dealing with settings where the observations take values either in a Euclidean or a discrete space, and the distribution over the future observations is derived from a conditional probability density or mass function. To simplify the exposition, we will restrict ourselves to those two settings, and define a prediction rule as a sequence of functions $(s_0, s_1, s_2, \dots)$ where each s_n maps a sequence $(x_1, \dots, x_n) \in \mathcal{X}^n$ to a probability density/mass function $s_n(\cdot|x_1, \dots, x_n)$, overloading the notation above. Of course, such prediction rules can be converted to the more general definition above. Much of the literature in statistics exploring prediction rules studies them in the most general settings, formulating the various properties we'll define in following chapters for general measurable spaces (see e.g. Berti et al. (2023) for a broad overview).

A prediction rule describes how a practitioner makes predictions about the future observations given the past. In the context of deep learning, a prediction rule might encompass the whole procedure for training a neural network on a dataset of past observations, and then using the trained neural network to make predictions about future observations. For example, $s_n(x_{n+1}|x_1, \dots, x_n)$ might be defined as the probability density of x_{n+1} given by a normalising flow trained on a dataset of examples $(x_1, \dots, x_n)$ following, for example, Stochastic Gradient Descent (SGD) with a maximum likelihood objective[2].

A standard assumption in machine learning problems is that the data $(X_1, X_2, \dots)$ is independent and identically distributed (*i.i.d.*) — the random variables $(X_1, X_2, \dots)$ are independent, and they all follow the same law: $X_i \sim P_X$ for all i. The *data generating distribution* P_X is unknown the the practitioner. In this paper, we'll primarily concern ourselves with the case of *i.i.d.* data. We'll describe below how one would go about defining a prediction rule in the Bayesian framework in an *i.i.d.* context, and then look at various properties that such prediction rules might have.

2.1. The Bayesian Inferential Approach

In its most general form, the Bayesian framework for inference is to **1)** specify a joint distribution over all the random variables of interest, and **2)** condition on the observed values

2. In this framework, prediction rules are deterministic; to view training a deep learning model as a prediction rule, all sources of randomness other than the data have to be fixed (e.g. through the seed). Each seed or sequence of seeds then effectively leads to a different prediction rule.

to obtain a posterior distribution over the unobserved variables of interest. In the context of a sequential prediction problem, one might specify a joint distribution over the random variables $(X_1, X_2, \dots)$ and condition on the observed values of $X_1, \dots, X_n$ to obtain a posterior distribution over e.g. X_{n+1}. In the context of *i.i.d.* data, a Bayesian would usually treat the data generating distribution P_X as an unknown, and place a prior distribution over it. To make predictions about the next observation X_{n+1} given observed outcomes $(x_1, \dots, x_n)$ for the previous observations $X_1, \dots, X_n$, one could describe the Bayesian *inferential* approach (Berti et al., 2023) as: **I.** Specify a prior over the generating distribution; **II.** Get a posterior over the generating distribution given the observed data; **III.** Compute the prediction $s_n(\cdot|x_1, \dots, x_n)$ by computing the posterior predictive distribution for X_{n+1}.

For instance, assuming the distributions of interest can be described with a parameter $\theta \in \mathbb{R}^d$ and a density $p_{X|\Theta}(x|\theta)$, and that the prior over Θ also has a density $p_\Theta(\theta)$, the procedure above might look like:

I Specify a prior density $p_\Theta(\theta)$ over the parameters;

II Compute the posterior density $p_{\theta|X_1,\dots,X_n}(\theta|x_1, \dots, x_n) \overset{\text{def}}{\propto} \prod_{i=1}^n p_{X|\theta}(x_i|\theta)p_\theta(\theta)$;

III Calculate the prediction $s_n(\cdot|x_1, \dots, x_n) \overset{\text{def}}{=} \int p_{X|\theta}(\cdot|\theta)p_{\theta|X_1,\dots,X_n}(\theta|x_1, \dots, x_n)d\theta$

2.2. Implicitly Bayesian Prediction Rules

An alternative to the inferential approach would to be to specify a prediction rule directly. This approach is often referred to in the literature as the *predictive approach* (Berti et al., 2023) and has been recently studied quite extensively in the statistics literature (Berti et al., 2013; Fong, 2021; Fong and Lehmann, 2022; Berti et al., 2019, 2021, 1998). It should be clear that this procedure *can* in effect result in the same prediction rule as that from the inferential approach.

For example, the practitioner could specify a linear model on (X_n, Y_n) with a uniform prior density on the covariates X_n (say, in the range $[0, 1]^d$), and a Gaussian prior $\mathcal{N}(\theta; 0, I)$ on the weights θ (assuming a homogeneous, variance σ^2, Gaussian noise). Here $y \mapsto \mathcal{N}(y; \mu, \Sigma)$ denotes a Gaussian density with mean μ and covariance Σ. Given observations $((x_1, y_1), \dots, (x_n, y_n))$, the practitioner would then construct a posterior on the weights (e.g. through Monte-Carlo sampling) and average over the posterior samples to obtain a posterior predictive distribution for X_{n+1}, Y_{n+1}. Alternatively, they could directly compute an equivalent prediction, without directly manifesting the posterior, with the prediction rule given in Appendix B. If a practitioner happened, by a stroke of luck, to specify this as their prediction rule without ever considering the underlying assumptions of a linear model and a prior, they'd still make the same predictions as if they followed the Bayesian framework with some underlying model.

A natural question to ask is then, under what conditions on the prediction rule is it equivalent to the inferential approach for *some* prior and likelihood? In other words, given a prediction rule, can one say whether there exists a prior and likelihood such that the predictions from the prediction rule match those of following the Bayesian framework with that likelihood/prior pair under the *i.i.d.* assumption? If this is the case, we'll say that the prediction rule is *implicitly Bayesian*.

In what follows, we'll look at the properties that the prediction rules defined following the Bayesian framework posses, and the properties that characterise them.

2.3. Characterising Implicitly Bayesian Prediction Rules

In the case of *i.i.d.* data, DeFinetti's theorem gives a simple condition for a prediction rule to be implicitly Bayesian. As mentioned before, a prediction rule implies a unique joint distribution over the sequence of random variables $(X_1, X_2, \dots)$. By a version of the DeFinetti's theorem, under some mild assumptions, a prediction rule is implicitly Bayesian if and only if the joint distribution it implies over $(X_1, X_2, \dots)$ is *exchangeable* (Hewitt and Savage, 1955):

Definition 1 (Exchangeable Sequence of Random Variables) *A finite sequence of n random variables $(X_1, \dots, X_n)$ is said to be exchangeable if for any permutation $\pi : \{1, \dots, n\} \to \{1, \dots, n\}$ the joint distribution of $(X_1, \dots, X_n)$ is the same as the joint distribution of $(X_{\pi(1)}, \dots, X_{\pi(n)})$.*

An infinite sequence of random variables $(X_1, X_2, \dots)$ is said to be exchangeable if for any n, the finite sequence $(X_1, \dots, X_n)$ is exchangeable.

By DeFinetti's theorem, we know that a sequence of random variables $(X_1, X_2, \dots)$ is exchangeable if and only if there exists a (unique) prior probability π on the space of probability measures Θ on $\mathcal{X}$ such that:

$$P\left[X_1 \in A_1, \dots, X_n \in A_n\right] = \int_{\Theta} \prod_{i=1}^{n} P_\theta(A_i) d\pi(P_\theta),$$

in other words, only if the there exists a likelihood/prior construction that defines the same joint distribution as the prediction rule. Hence, exchangeability is the defining characteristic of implicitly Bayesian prediction rules on *i.i.d.* data. This suggests one direct way of checking whether a prediction rule is implicitly Bayesian: check whether the joint distribution over $(X_1, X_2, \dots)$ implied by the prediction rule is exchangeable.

Conditionally Identically Distributed Another desirable coherence property that we might expect of a prediction rule is that the future observations are identically distributed given the past. For example, under the *i.i.d.*assumption, if we were to observe $(x_1, \dots, x_n)$, the prediction for the next observation X_{n+1} *surely* shouldn't be different from the prediction for the observation after that. After all, we know they are identically distributed, we just don't know what the distribution is; it'd be irrational to make different predictions for X_{n+1}, X_{n+2} given the same data.

This property can be formalised as follows:

Definition 2 (Conditionally Identically Distributed) *We say that a sequence of random variables $(X_1, X_2, \dots)$ is conditionally identically distributed (c.i.d.) if for any n:*

$$P\left[X_{n+1} \in \cdot | X_1 = x_1, \dots, X_n = x_n\right] = P\left[X_{n+k} \in \cdot | X_1 = x_1, \dots, X_n = x_n\right] \quad \forall k > n$$

holds almost surely.

A prediction rule is then *c.i.d.* if the joint distribution it implies over $(X_1, X_2, \dots)$ is *c.i.d.*. Conditionally Identically Distributed sequences have been introduced in (Kallenberg, 1988) and studied and applied in a range of works (Berti et al., 2004, 2013; Fong et al., 2021a).

It should be clear that each exchangeable sequence is *c.i.d.*, as exchangeability implies $X_1, \ldots, X_n, X_{n+1} \overset{d}{=} X_1, \ldots, X_n, X_{n+k}$ for any $k > 0$. Hence, following the Bayesian framework will yield *c.i.d.* prediction rules, and any implicitly Bayesian prediction rule will be *c.i.d.*. Not all *c.i.d.* sequences are exchangeable, however. *c.i.d.* can hence be seen as a weakening of the condition of exchangeability.

Stationarity There is one complimentary property that not all *c.i.d.* sequences have that would make them *implicitly Bayesian*. Namely, *stationarity*:

Definition 3 (Stationary sequence) *A sequence of random variables* $(X_1, X_2, \ldots)$ *is said to be stationary if for any* $n, k > 0$*:*

$$X_1, \ldots, X_n \overset{d}{=} X_{1+k}, \ldots, X_{n+k} \tag{1}$$

In the context from the prediction rules, stationarity implies sampling k datapoints $x'_1, \ldots, x'_k$, from the prediction rule itself and prepending them to the observed data $\mathcal{D} = (x_1, \ldots, x_n)$, would yield a prediction rule $s_{n+k}(\cdot | x'_1, \ldots, x'_k, \mathcal{D})$ that's in expectation the same as $s_n(\cdot | x_1, \ldots, x_n)$.

By the result of Kallenberg (1988, Proposition 2.1), exchangeability exactly amounts to stationarity and the *c.i.d.* condition; stationarity and *c.i.d.* properties are another way of characterising implicitly Bayesian prediction rules.

Spreadability Another condition that turns out to also characterise implicitely Bayesian models is *spreadability*:

Definition 4 (Spreadable sequence) *A sequence of random variables* $(X_1, X_2, \ldots)$ *is said to be spreadable if for any* $n > 0$ *and any sequence of indices* $k_n > k_{n-1} > \cdots > k_1 \geq 1$*:*

$$X_1, \ldots, X_n \overset{d}{=} X_{k_1}, \ldots, X_{k_n} \tag{2}$$

Spreadability says that the distribution of the first n observations is the same as the distribution of any n observations from the sequence. In the context of a prediction rule, it implies that, if we were to construct a new dataset by possibly sampling $k_i \geq 0$ datapoints from the prediction rule conditioned on the previous datapoints inbetween "observed" datapoints x_i, x_{i+1}, the final prediction given that dataset would in expectation be the same as the prediction given the original dataset. Kallenberg (1988) has shown that spreadability is equivalent to exchangeability. In other words, spreadability is yet another property that characterises implicitly Bayesian prediction rules.

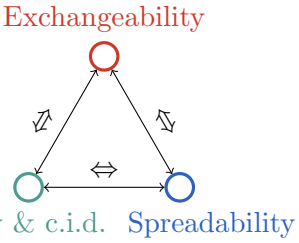

Figure 1: Equivalent characterisation of exchangeable sequences, and hence implicitly Bayesian prediction rules.

In summary, Figure 1 illustrates the different properties that an implicitly Bayesian prediction rule posses and can be characterised by. Some might be easier to establish or approximately enforce then others, and each suggests its own way of constructing an exchangeable prediction rule from a non-exchangeable one. It's an interesting question as to what extent various commonly used machine learning and deep learning methods posses these different properties, and how various design choices might affect them.

Can a prediction rule at a fixed step be extended to an implicitely Bayesian prediction rule? In deep learning, we would often taylor the approach to the amount of data available at hand. If we have fewer than hundreds of datapoints, we might not even consider using a neural network at all. One might therefore wonder, given a specification for a prediction rule at step n only, can it be extended to a prediction rule for all n to satisfy the aforementioned properties? We discuss this in Section C, but, in short, the requirements on s_n turn out to be quite stringent.

3. Non-implicitly Bayesian agents are vulnerable to adversarial bets

Many of the fundamental axioms of probability, and hence probabilistic inference, can be argued for on the basis of various Dutch Book arguments (Pettigrew, 2020). These generally show that, unless one adopts the intuitively sensible-seeming axioms of probability (e.g. probabilities must sum to 1), one is left vulnerable to accepting seemingly appealing, but highly disadvantageous, bets.

In this subsection, along similar lines, we'll show that using a non-exchangeable prediction rule, when the true distribution is exchangeable or *i.i.d.*, leaves one vulnerable to accepting bets that are in expectation disadvantageous. In particular, an agent that assigns probabilities to a sequence of data drawn from an (unknown) exchangeable distribution, and is willing to accept bets on the next sequence of events if their beliefs indicate the bet is in expectation favourable, is vulnerable to an adversary who knows the beliefs of the agent, but doesn't know the true probabilities of the events. We'll show this only in the finite discrete setting, i.e. when the sequence of events takes values in $\mathcal{X}^\infty$ for some finite space $\mathcal{X}$, and the prediction rule $(s_1, s_2, \dots)$ is a sequence of conditional probability mass functions.

Concretely, we will show a result for an adversary that knows the beliefs of the agent, but not the true probabilities of the events. For an agent that is following some prediction rule $(s_1, s_2, \dots)$. For any $n \in \mathbb{N}$, as described before, this implies some joint probability mass function on $\mathcal{X}^n$. Let's denote this probability mass function by $q : \mathcal{X}^n \to [0, 1]$. We will consider an adversary that constructs an "exchangeable-ified" version of the agent's beliefs q as:

$$\bar{q}(x_1, \dots, x_n) = \frac{1}{n!} \sum_{\pi \in \Pi_n} q(x_{\pi(1)}, \dots, x_{\pi(n)}), \tag{3}$$

where Π_n is the collection of all permutations of n elements. It should be clear that $\bar{q}$ is itself an exchangeable probability mass function.

We can represent a bet between the agent an adversary with a function $r : \mathcal{X}^n \to \mathbb{R}$, where, if the observed outcome is $x_1, \dots, x_n$, the change to the agent's wealth after the bet is $r(x_1, \dots, x_n)$; in other words, the agents receives $|r(x_1, \dots, x_n)|$ if $r(x_1, \dots, x_n)$ is positive, and the agent pays $|r(x_1, \dots, x_n)|$ to the adversary if the value of r is negative. The bet r can be said to be *favourable* given a probability mass function P if the expected return is positive, i.e.: $\sum_{x^{(n)} \in \mathcal{X}^n} r(x^{(n)}) P(x^{(n)}) > 0$. The bet reward is anti-symmetric, so from the perspective of the adversary with a probability mass function $\bar{P}$, the expected return is $\sum_{x^{(n)} \in \mathcal{X}^n} (-r(x^{(n)})) \bar{P}(x^{(n)})$. A bet that is favourable for both the agent and the adversary with beliefs $q, \bar{q}$ is said to be *admissible*. It is also helpful to define a notion of a *minimal* bet: a bet r is minimal if it is the smallest possible[3] among all bets that give

the same expected return to both the agent and the adversary. Restricting to minimal bets hence simply ensures that the agents do not arbitrarily make bets that are not justified on the ground of their beliefs.

We can show that the following holds (see Appendix A):

Theorem 5 *Given an agent with beliefs $q : \mathcal{X}^n \to [0, 1]$ on the next sequence of n events, and an adversary with "exchangeable-ified" beliefs $\bar{q}$ as defined in (3), for any exchangeable distribution $p : \mathcal{X}^n \to [0, 1]$ that has common support with q all minimal and admissible bets r have strictly negative expected return for the agent under p:*

$$\sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) r(x^{(n)}) < 0 \tag{4}$$

*Furthermore, either: **1)** q is exchangeable and $\bar{q} = q$, and there are* no *admissible bets; or **2)** q is not exchangeable and there exist admissible (and minimal) bets.*

In particular, the above holds when p is an *i.i.d.* distribution.

The result shows that, in the setting where bets are placed ahead of time, an agent that follows a non-exchangeable (i.e. non-implicitly Bayesian) prediction rule is vulnerable to accepting bets from an adversary that is able to *make* their beliefs exchangeable.

4. Measuring Implicit Bayesianness

Notation As we'll deal with nested expectations and variances in this section, for the sake of clarity we'll adopt the machine learning notation for conditional expectations: for a function f that depends on multiple random variables $A, B, \dots$ we'll write $\mathbb{E}_A[f(A, B, \dots)]$ to denote the *conditional expectation* of f conditioned on all the variables other than A — the subscript indicates the variable that is "marginalised out".[4] The subscript in the variance $\mathrm{Var}_A[f(A, B, \dots)]$ is defined analogously.

In the preceding section, we argued that implicitly Bayesian prediction rules might be desirable, and presented various testable properties that characterise them. In this section, we'll look at how we can go about measuring these properties in practice, present empirical results for both Bayesian and non-Bayesian models including deep learning models, and show that simple design choices can lead to more or less implicitly Bayesian prediction rules. As was shown in the previous section, exchangeability is one defining characteristic of implicitly Bayesian prediction rules. In this paper, we'll focus on measuring exchangeability, although the other criteria in Section 2.3 are potentially equally interesting candidates.

Exchangeability is not easy to verify. Even for a finite sequence, it requires checking that $\prod_{i=1}^{n} s_i(x_i|x_1, \dots, x_{i-1}) = \prod_{i=1}^{n} s_i(x_{\pi(i)}|x_{\pi(1)}, \dots, x_{\pi(i-1)})$ with probability 1 (effectively for almost every sequence $(x_1, \dots, x_n) \in \mathcal{X}^n$) for every permutation π.[5] Even for a single sequence of observations $(x_1, \dots, x_n)$ can be computationally expensive, as it might require checking eq. for all $n!$ permutations of the sequence. Lastly, we rarely expect exact exchangeability to hold in practice.

3. Smallest in L^2 norm, where the L^2 norm of $f : \mathcal{X}^n \to \mathbb{R}$ is taken to be $\sqrt{\sum_{x^{(n)} \in \mathcal{X}^n} f(x)^2}$
4. For example, if A, B, C are random variables, then $\mathbb{E}_A[f(A, B, C)] = \mathbb{E}[f(A, B, C)|B, C]$, and $\mathbb{E}_A[\mathbb{E}_B[f(A, B, C)]] = \mathbb{E}[\mathbb{E}[f(A, B, C)|A, C]|C]$.
5. With respect to the joint measure on $\mathcal{X}^n$ implied by the prediction rule, assuming s_i is a probability mass function/probability density function.

Also, even for implementations of models with a strong Bayesian motivation, the exact equality might not hold due to numerical errors and/or approximate inference. Ideally, we'd like some measure of the *degree* of exchangeability of a prediction rule.

To measure how close to being exchangeable a prediction rule is, in this paper we suggest measuring the *variance* of the log-joint as we randomly sample permutations of the data uniformly at random. Concretely, if Π is a random variable that takes values in the set of all permutations of $\{1, \dots, n\}$ with equal probability, we propose measuring:

$$\mathrm{Var}_\Pi \left[\log \prod_{i=1}^{n} s_i(x_{\Pi(i)} | x_{\Pi(1)}, \dots, x_{\Pi(i-1)}) \right] = \mathrm{Var}_\Pi \left[\sum_{i=1}^{n} \log s_i(x_{\Pi(i)} | x_{\Pi(1)}, \dots, x_{\Pi(i-1)}) \right]$$

for a given sequence of datapoints $(x_1, \dots, x_n)$. The log makes the measure more numerically stable. To get around checking this condition holds for (almost) every possible sequence $(x_1, \dots, x_n)$, we propose sampling sequences of datapoints and reporting the expected variance of the log-joint as a measure of implicit Bayesianness. Since it's nontrivial to evaluate the variance of the log-joint on an infinite sequence, we resort to checking for exchangeability on a finite sequence of datapoints.

Arguably, we might care about 'being implicitly Bayesian' in some regions of the input space more than others. Hence, we evaluate the variance of the log-joint preferentially on sequences of data sampled from a task of interest. Hence, we resort to measuring the *expected* variance given some reference distribution over sequences of n datapoints $\mathcal{D}^n$.[6] Let $X_1, \dots, X_n$ be random variables that follow the law $\mathcal{D}^n$. Then, our measure looks like:

$$m_{\texttt{var}}\left((s_1, s_2, \dots)\right) = \mathbb{E}_{X_1,\dots,X_n} \left[\mathrm{Var}_\Pi \left[\sum_{i=1}^{n} \log s_i(x_{\Pi(i)} | x_{\Pi(1)}, \dots, x_{\Pi(i-1)}) \right] \right] \tag{5}$$

One last caveat remains for the case of non-deterministic prediction rules such as training of a deep learning model. As we mentioned before, in this case each random seed ϵ will yield a different prediction rule $s^\epsilon = (s_1^\epsilon, s_2^\epsilon, \dots)$. Hence, the metric that we actually measure is the average variance of the log-joint over different random seeds $\epsilon \in \mathcal{E}$: $\frac{1}{|\mathcal{E}|} \sum_{\epsilon \in \mathcal{E}} m_{\texttt{var}}\left((s_1^\epsilon, s_2^\epsilon, \dots)\right)$. In practice, we approximate (5) by Monte-Carlo sampling, taking the empirical variance and expectation of the log-joint to arrive at a computable metric.

4.1. Implicit Bayesianness vs. Performance

Although we argue implicit Bayesianness is a desirable property, it is not an end-goal on its own. After all, if a prediction rule has lacklustre performance on the tasks we are interested in, it likely won't be much comfort that it is implicitly Bayesian.[7] Hence, since predictive performance is a key consideration, we'll aim to depict both the aforementioned measure of implicit Bayesianness in (5), as well as a measure of performance on

6. For a distribution $\mathcal{D}^n$ with full support, the prediction rule s is exchangeable (on the first n observations) if and only if the variance of the log-joint is zero, justifying the use of the variance of the log-joint as a measure of exchangeability.

7. It's not difficult to construct trivial implicitly Bayesian prediction rules. For example, one could set $s_i(x_i | x_1, \dots, x_{i-1}) = q(x_i)$ for some fixed probability density/mass function $q : \mathcal{X} \to \mathbb{R}$. This prediction rule is trivially exchangeable, but it's not particularly useful as no learning is taking place — we are predicting the same thing no matter what we observe.

the data-generating distribution $\mathcal{D}^n$ in the experiments that follow. To report an aggregate of the performance on the entire sequence of n observations, we'll report the average negative log-likelihood (NLL) of the prediction rule on the sequence of observations: $\mathbb{E}_{X_1,\ldots,X_n}\left[\sum_{i=1}^{n} \log s_i(x_i|x_1,\ldots,x_{i-1})\right]$. If the true data-generating distribution $\mathcal{D}^n$ is *i.i.d.*, which it will be for all the tasks considered below, then that expectation is the same as:

$$\mathbb{E}_{X_1,\ldots,X_n}\left[\mathbb{E}_{\Pi}\left[\sum_{i=1}^{n} \log s_i(x_{\Pi(i)}|x_{\Pi(1)},\ldots,x_{\Pi(i-1)})\right]\right] \tag{6}$$

allowing as to use the same samples to compute **1)** a measure of implicit Bayesiannes, i.e. the variance of the log-joint in (5) and **2)** a measure of performance, i.e. the expectation of the log-joint in (6). We investigate these metrics empirically in the following section.

4.2. Results

We consider a simple 1D regression task on which we compare deep learning prediction rules against exact Bayesian methods, such as exact conditioning in a Gaussian Process. The task is pictured in Figure 8. The true function was purposefully chosen to be discontinuous to yield a model mismatch for a Gaussian Process with a smooth kernel. Figure 2 summarises all results.

Gaussian Process & Prior Network To illustrate that a parametric model fit with gradient-descent on an objective function can be implicitly Bayesian, we compare a Gaussian Proccess (GP) against linear models fit with gradient descent following the "prior networks" procedure described in Osband et al. (2018); this procedure entails full-batch gradient descent optimisation of the parameters on a negative log-likelihood objective (similarly to the canonical recipe for applying deep learning to regression), with the difference that the targets are augmented with random Gaussian noise and the weights are ℓ_2-regularised towards a random Gaussian sample. Osband et al. (2018) show this yields exact samples from the posterior of a Bayesian Linear Model. Hence, as we ensemble the predictions from more and more of these models, the resulting prediction rule should, in the limit, be implicitly Bayesian. To make the comparison clear, we chose the features and the (implicit) prior for the linear model in such a way that the resulting model would be equivalent to the Gaussian Process with a squared exponential kernel its compared against (Appendix D.1).

The results for a Gaussian Process and prior networks are shown in Figures 4 and 2**a**. As expected, the linear model approaches both the performance and the implicit Bayesianness of the GP as the ensemble size increases. Notably, even the exact conditioning Gaussian process is not perfectly implicitly Bayesian due to numerical precision errors, as evidenced by the non-zero variance of the log-joint. Nonetheless, this result demonstrates that it's possible to get close to the implicit Bayesianness of exact Bayesian methods with algorithms that resemble those used in deep learning. It hopefully illustrates that improving implicit Bayesianness of deep learning algorithms might be an achievable task.

Motivated by the results above, a natural question to ask might be: are there any *simple* strategies or design choices that can improve the implicit Bayesianness of deep learning models? Do these strategies tend to come at the trade-off of performance?

Ensembling of deep learning models We run the same experiment as above with a deep learning model – a 3-hidden layer multi-layer perceptron (MLP) optimised with

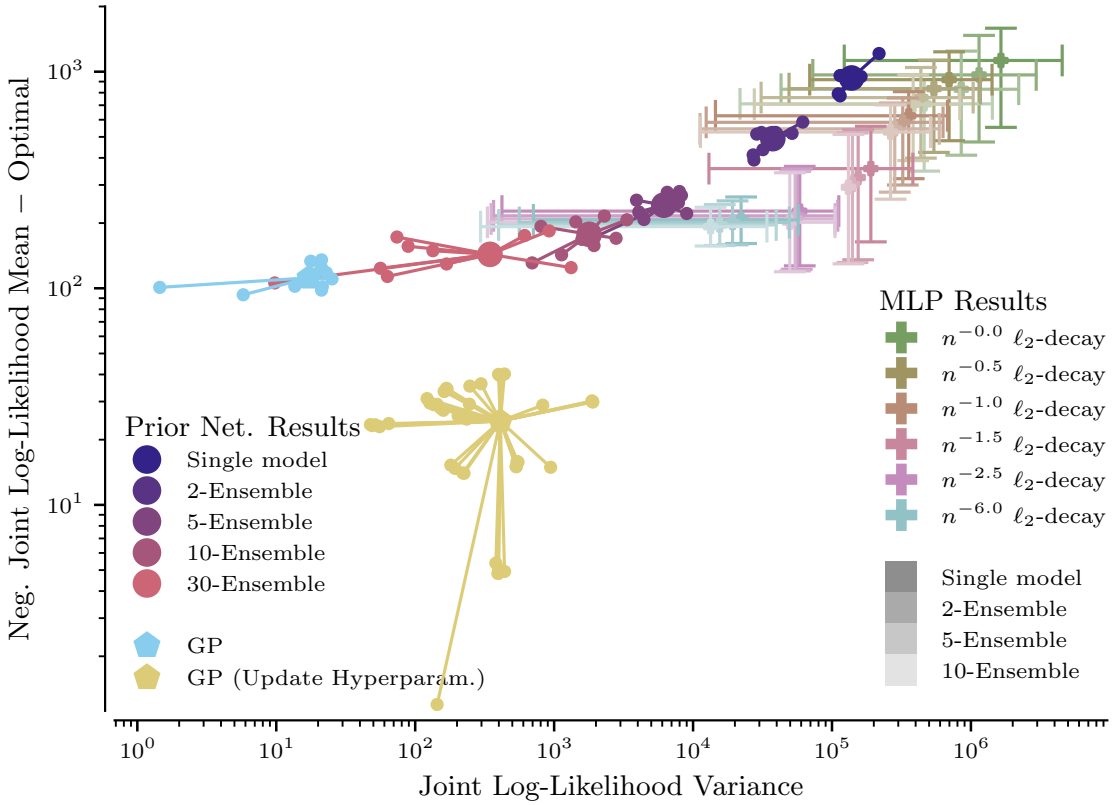

Figure 2: Joint log-likelihood variance as a measure of *implicit Bayesianness* (defined in (5)) vs. the negative sum of log-likelihood as a measure of *performance* (defined in (6)) on the regression task described in Section 4.2. Results show: **a)** a Gaussian Process (GP) with a squared exponential kernel ⬟, and a linear model fit with the prior networks procedure ● ... ●; The larger the linear model ensemble, the more the performance and implicit Bayesianness approaches the GP. **b)** a Gaussian Process, but with kernel hyperparameters optimised with marginal likelihood ⬟; **c)** Neural Networks (MLPs) with weight-decay scaled differently as a function of the training data size n ✚ ... ✚; results are plotted for both a single network, as well as ensemble of various sizes, with the lightness (e.g. ✚✚✚✚) indicating the ensemble size. The large dots represent the average over multiple dataset samples from the data-generating distribution, and the small dots/error-bars represent deviation of results for individual dataset samples ($(x_1, y_1), \ldots, (x_{100}, y_{100})$). The expected negative log-likelihood of the *optimal* predictions on this dataset (using the true distribution) is subtracted from the mean negative joint log-likelihood plotted.

Stochastic Gradient Descent (SGD). We compare a base MLP model against ensemble predictions (Lakshminarayanan et al., 2017). The results are highlighted in Figures 5 and 6. Ensembling appears to not only improve predictive performance, but also improve the measure of implicit Bayesianness, although not by as much as for the linear model.

ℓ_2 **Decay Schedules** A concern when defining a prediction rule by training a deep learning model for a fixed number of epochs is that the algorithm has effectively no notion of what the data-set size is. Training on two different datasets, the second one being a copy of the first one with each element repeated twice, would yield an identical training routine

(bar effects of the random seed). A common workaround is to use stronger regularisation on smaller datasets[8]. For example, the Maximum-a-Posteriori (MAP) estimation perspective of optimising the negative log-likelihood loss with ℓ_2-regularisation suggests that the ℓ_2 decay coefficient should be decayed as n^{-1} (Bishop, 2006). Can a strategy as simple as decaying the ℓ_2 regularisation coefficient as a function of the number of datapoints seen have a notable effect on the implicit Bayesianness of the resulting prediction rule? If it does, would increased implicit Bayesianness come at the cost of predictive performance? To investigate this, we run the same experiment as above, but with a decay schedule for the ℓ_2 regularisation coefficient of the form $c_\alpha n^{-\alpha}$ for different values of α. c_α is in each case set so that the value of ℓ_2 decay would match for $n = 100$ (see Appendix D). The results are shown in Figure 7. Surprisingly, the improvements to both implicit Bayesianness and predictive performance are quite substantial, and, on this task, greater than what's achievable with ensembling of up to 10 models. With the two design-choices considered — ensemble size and ℓ_2 decay schedule — more implicitly Bayesian prediction rules seem to generally perform better as well.

5. Future Directions

Measuring other properties of implicitly Bayesian predictions In this work, we only experimentally considered measuring exchangeability of a prediction rule. As discussed in Section 2.3, there are other desirable properties that an implicitly Bayesian prediction rule posses, and other conditions that characterise them. It would be interesting to investigate empirical metrics based on these different conditions, and measuring them in practice.

Martingale Posterior Sampling As shown in (Fong et al., 2021b), if the prediction rule is *c.i.d.*, we can obtain a functional uncertainty estimates that give a notion of reducible uncertainty. Although the method in Fong et al. (2021b) would be computationally extremely burdensome for deep learning-based prediction rules as defined in this work, it would be interesting to investigate whether the method could be adapted to be more computationally tractable. Furthermore, it's an interesting question as to whether improving any of the implicit Bayesian properties would then lead to more *useful* uncertainty estimates, for example when used in a Bayesian optimisation, active learning or reinforcement learning context.

Searching for implicitly Bayesian updates Turning implicit Bayesianness into a measurable metric opens the doors for black-box optimisation for that property. Prior work has successfully meta-learned optimisers for faster training of more performant deep learning models (Metz et al., 2022). By meta-optimising update rules for *both* good performance and implicit Bayesianness, training algorithms that are both performant **and** update their predictions in a coherent way could perhaps be learnt.

6. Conclusion

In conclusion, this paper proposes a new perspective on incorporating Bayesian principles into deep learning, shifting focus from explicit model specification to characterising and achieving predictions that are implicitly Bayesian.

8. Normally, this stronger regularisation might be arrived at through the means of cross-validation hyperparameter selection.

References

Jordan T. Ash and Ryan P. Adams. On Warm-Starting Neural Network Training, December 2020. URL http://arxiv.org/abs/1910.08475.

P. Berti, E. Regazzini, and P. Rigo. Well Calibrated, Coherent Forecasting Systems. *Theory of Probability & Its Applications*, 42(1):82–102, January 1998. ISSN 0040-585X, 1095-7219. doi: 10.1137/S0040585X97975988. URL http://epubs.siam.org/doi/10.1137/S0040585X97975988.

Patrizia Berti, Luca Pratelli, and Pietro Rigo. Limit theorems for a class of identically distributed random variables. *The Annals of Probability*, 32(3), July 2004. ISSN 0091-1798. doi: 10.1214/009117904000000676. URL http://arxiv.org/abs/math/0410105. Comment: Published by the Institute of Mathematical Statistics (http://www.imstat.org) in the Annals of Probability (http://www.imstat.org/aop/) at http://dx.doi.org/10.1214/009117904000000676.

Patrizia Berti, Luca Pratelli, and Pietro Rigo. Exchangeable Sequences Driven by an Absolutely Continuous Random Measure. *The Annals of Probability*, 41(3):2090–2102, 2013. ISSN 0091-1798. URL https://www.jstor.org/stable/23469556.

PATRIZIA Berti, EMANUELA Dreassi, LUCA Pratelli, and PIETRO Rigo. A predictive approach to Bayesian nonparametrics. *submitted, currently available at: http://local. disia. unifi. it/dreassi/mg. pdf*, 2019. URL https://mate.unipv.it/~rigo/mg.pdf.

Patrizia Berti, Emanuela Dreassi, Fabrizio Leisen, Pietro Rigo, and Luca Pratelli. Bayesian predictive inference without a prior, April 2021. URL http://arxiv.org/abs/2104.11643.

Patrizia Berti, Emanuela Dreassi, Fabrizio Leisen, Pietro Rigo, and Luca Pratelli. A probabilistic view on predictive constructions for Bayesian learning, January 2023. URL http://arxiv.org/abs/2208.06785.

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

Lester E. Dubins, Leonard J. Savage, William Sudderth, and David Gilat. *How to Gamble If You Must: Inequalities for Stochastic Processes*. Courier Corporation, August 2014. ISBN 978-0-486-78064-1.

Chung Hang Edwin Fong. The predictive view of Bayesian inference. *PhD Thesis*, 2021.

Edwin Fong and Brieuc Lehmann. A Predictive Approach to Bayesian Nonparametric Survival Analysis, February 2022. URL http://arxiv.org/abs/2202.10361. Comment: Accepted to AISTATS 2022.

Edwin Fong, Chris Holmes, and Stephen G Walker. Martingale posterior distributions. *arXiv preprint arXiv:2103.15671*, 2021a.

Edwin Fong, Chris Holmes, and Stephen G. Walker. Martingale posterior distributions, November 2021b. URL http://arxiv.org/abs/2103.15671. Comment: 62 pages, 22

figures, 3 tables; added discussion on frequentist consistency, included more context, added references.

Edwin Hewitt and Leonard J Savage. Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, 80(2):470–501, 1955.

J. Hoffman-Jorgensen. *Probability With a View Towards Statistics, Volume II*. Routledge, November 2017. ISBN 978-1-351-42154-6.

Olav Kallenberg. Spreading and predictable sampling in exchangeable sequences and processes. *The Annals of Probability*, pages 508–534, 1988.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, November 2017. URL http://arxiv.org/abs/1612.01474. Comment: NIPS 2017.

David A Lane and William D Sudderth. Coherent predictive inference. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 166–185, 1984.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

Luke Metz, James Harrison, C Daniel Freeman, Amil Merchant, Lucas Beyer, James Bradbury, Naman Agrawal, Ben Poole, Igor Mordatch, Adam Roberts, et al. Velo: Training versatile learned optimizers by scaling up. *arXiv preprint arXiv:2211.09760*, 2022.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized Prior Functions for Deep Reinforcement Learning, November 2018. URL http://arxiv.org/abs/1806.03335.

Richard Pettigrew. *Dutch book arguments*. Cambridge University Press, 2020.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, November 2005. ISBN 978-0-262-25683-4. doi: 10.7551/mitpress/3206.001.0001. URL https://direct.mit.edu/books/book/2320/Gaussian-Processes-for-Machine-Learning.

Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. Functional Variational Bayesian Neural Networks, March 2019. URL http://arxiv.org/abs/1903.05779. Comment: ICLR 2019.

Appendix A. Non-implicitly Bayesian agents are vulnerable to adversarial bets

Consider an agent that is following some prediction rule $(s_1, s_2, \dots)$. For any $n \in \mathbb{N}$, as mentioned before, this implies some joint probability mass function on $\mathcal{X}^n$. Let's denote this probability mass function by $q : \mathcal{X}^n \to [0, 1]$. We'll denote elements of the sequence of events by $x^{(n)} = (x_1, \dots, x_n) \in \mathcal{X}^n$.

We can represent a bet between the agent an adversary with a function $r : \mathcal{X}^n \to \mathbb{R}$, where, if the observed outcome is $x_1, \ldots, x_n$, the change to the agent's wealth after the bet is $r(x_1, \ldots, x_n)$; in other words, the agents recieves $|r(x_1, \ldots, x_n)|$ if $r(x_1, \ldots, x_n)$ is positive, and the agent pays $|r(x_1, \ldots, x_n)|$ to the adversary if the value of r is negative.

The bet r can be said to be *favourable* given a probability mass function P if the expected return is positive, i.e.:

$$\sum_{x^{(n)} \in \mathcal{X}^n} r(x^{(n)}) P(x^{(n)}) > 0 \tag{7}$$

The bet reward is of course anti-symmetric, so from the perspective of the adversary with a probability mass function $\bar{P}$, the expected return is:

$$\sum_{x^{(n)} \in \mathcal{X}^n} (-r(x^{(n)})) \bar{P}(x^{(n)}) \tag{8}$$

In what follows, it will be helpful to simplify the notation. If we enumerate all the elements of $\mathcal{X}^n$, we can represent functions from $\mathcal{X}^n$ to $\mathbb{R}$ as vectors in $\mathbb{R}^{|\mathcal{X}^n|}$. For example, if $\mathcal{X} = \{0, 1\}$, $n = 2$, and the chosen ordering of elements of $\mathcal{X}^n$ is $((0,0), (0,1), (1,0), (1,1))$, then the function r can be represented as a vector in $\mathbb{R}^4$:

$$\boldsymbol{r} = \begin{bmatrix} r(0,0) \\ r(0,1) \\ r(1,0) \\ r(1,1) \end{bmatrix}$$

for the reminder of this section (and this section only) we'll switch freely between the function notation and the vector notation for the same function as it should be apparent from the context which one is being considered.

Now, let's assume that both the agent and the adversary are willing to accept a bet on the next sequence of outcomes in $\mathcal{X}^n$ only if it's in expectation strictly favourable to them according to their beliefs. In the case of the agent with beliefs q, or $\boldsymbol{q}$ in vector representation, this can be concisely stated as $\boldsymbol{r} \cdot \boldsymbol{q} = \sum_{i=1}^{|\mathcal{X}^n|} r_i q_i > 0$. A bet that is favourable for both the agent with beliefs $\boldsymbol{q}$ and an adversary with beliefs $\bar{\boldsymbol{q}}$ is then said to be *admissible*.

It is also helpful to define a notion of a *minimal* bet. The bet is minimal if it is a linear combination of the beliefs of the agent $\boldsymbol{q}$ and the adversary $\bar{\boldsymbol{q}}$, i.e. if it can be represented as $\boldsymbol{r} = a\boldsymbol{q} + b\bar{\boldsymbol{q}}$ for some $a, b \in \mathbb{R}$. The rationale for this is that adding components to the bet that are orthogonal to the beliefs of the agent and the adversary will not change the expected return of the bet for either of them. In other words, for any bet $\boldsymbol{r}$ that is not minimal, there exists a smaller minimal bet $\boldsymbol{r'}$ such that the expected returns of $\boldsymbol{r}$ and $\boldsymbol{r'}$ are the same for both the agent and the adversary. Restricting ourselves to minimal bets hence simply ensures that the agents do not arbitrarily make bets that are not justified on the ground of their beliefs. In short, we'll want to state a result that says something along the lines of "if a sensible bet is made between the agent and the adversary, then *no-matter what the true distribution is* the expected return for the agent is strictly negative". If the bet isn't minimal, then we could find some true distribution that rewards one of the bettors despite them having no 'edge' or advantage in terms of their beliefs.

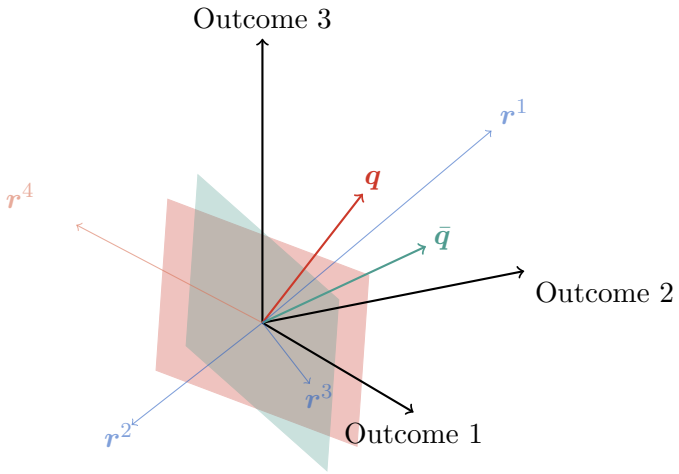

Figure 3: An illustration of the beliefs $\boldsymbol{q}$ of the agent and the beliefs $\bar{\boldsymbol{q}}$ of the adversary as vectors. The favourable bets for the agent are those that lie in the direction of $\boldsymbol{q}$ of the shaded plane orthogonal to $\boldsymbol{q}$. Similarly, the favourable bets for the adversary are those that lie in the direction of $-\bar{\boldsymbol{q}}$ of the shaded plane orthogonal to $\bar{\boldsymbol{q}}$. Multiple inadmissible bets $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3$ are shown, as well as one admissible bet $\boldsymbol{r}_4$.

We'll consider the case of an adversary who knows the beliefs of the agent, i.e. the probabilities $q : \mathcal{X}^n \to [0, 1]$, but doesn't know the true probabilities of the events. We'll consider the case where they construct an "exchangeable-ified" version of the agent's beliefs q as:

$$\bar{q}(x^{(n)}) = \frac{1}{n!} \sum_{\pi \in \Pi_n} q(x_{\pi(1)}, \ldots, x_{\pi(n)}), \tag{9}$$

where Π_n is the collection of all permutations of n elements. It should be clear that $\bar{q}$ is itself exchangeable.

We can show that the following holds:

Theorem 6 *Given an agent with beliefs $q : \mathcal{X}^n \to [0, 1]$ on the next sequence of n events, and an adversary with "exchangeable-ified" beliefs $\bar{q}$ as defined in (3), for any exchangeable distribution $p : \mathcal{X}^n \to [0, 1]$ that has common support with q all minimal and admissible bets r have strictly negative expected return for the agent under p:*

$$\boldsymbol{p} \cdot \boldsymbol{r} = \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) r(x^{(n)}) < 0 \tag{10}$$

Furthermore, either:

- *q is exchangeable and $\bar{q} = q$, and there are* no *admissible bets.*
- *q is not exchangeable and there exist admissible (and minimal) bets.*

In particular, the above holds when p is an *i.i.d.* distribution.

Proof

for any exchangeable function $f : \mathcal{X}^n \to \mathbb{R}$ we have that:

$$\begin{aligned}
\boldsymbol{p} \cdot \bar{\boldsymbol{q}} &= \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) \bar{q}(x^{(n)}) && (11)\\
&= \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) \left(\frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} q(\pi(x^{(n)})) \right) && (12)\\
&= \frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) q(\pi(x^{(n)})) && (13)\\
&= \frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} \sum_{x^{(n)} \in \mathcal{X}^n} p(\pi(x^{(n)})) q(\pi(x^{(n)})) && (14)\\
&= \frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) q(x^{(n)}) && (15)\\
&= \sum_{x^{(n)} \in \mathcal{X}^n} p(x^{(n)}) q(x^{(n)}) = \boldsymbol{p} \cdot \boldsymbol{q}, && (16)
\end{aligned}$$

and so:

$$\boldsymbol{p} \cdot \bar{\boldsymbol{q}} = \boldsymbol{p} \cdot \boldsymbol{q} \qquad\qquad \bar{\boldsymbol{q}} \cdot \bar{\boldsymbol{q}} = \boldsymbol{q} \cdot \bar{\boldsymbol{q}} \tag{17}$$

Suppose r is minimal. Then, $\boldsymbol{r}$ can be represented as:

$$\boldsymbol{r} = a\boldsymbol{q} + b\bar{\boldsymbol{q}},$$

for some constants a, b...

Hence, admissibility of $\boldsymbol{r}$ equates to:

$$\begin{aligned}
\boldsymbol{q} \cdot \boldsymbol{r} > 0 \qquad & \Rightarrow a(\boldsymbol{q} \cdot \boldsymbol{q}) + b(\boldsymbol{q} \cdot \bar{\boldsymbol{q}}) && > 0 && (18)\\
\bar{\boldsymbol{q}} \cdot \boldsymbol{r} > 0 \qquad & \Rightarrow a(\bar{\boldsymbol{q}} \cdot \boldsymbol{q}) + b(\bar{\boldsymbol{q}} \cdot \bar{\boldsymbol{q}}) = a(\bar{\boldsymbol{q}} \cdot \boldsymbol{q}) + b(\bar{\boldsymbol{q}} \cdot \boldsymbol{q}) && < 0 &\\
& \Rightarrow a + b && < 0, && (19)
\end{aligned}$$

as $\boldsymbol{q} \cdot \bar{\boldsymbol{q}} > 0$.

From eq. X we have that:

From eq. X we can rewrite the expected return for the agent as:

$$\boldsymbol{p} \cdot \boldsymbol{r} = a\boldsymbol{p} \cdot \boldsymbol{q} + b\boldsymbol{p} \cdot \bar{\boldsymbol{q}} = a\boldsymbol{p} \cdot \boldsymbol{q} + b\boldsymbol{p} \cdot \boldsymbol{q} = (a + b)(\boldsymbol{p} \cdot \boldsymbol{q}) \tag{20}$$

Now, $\boldsymbol{p} \cdot \boldsymbol{q} > 0$ as long as $\boldsymbol{p}$ and $\boldsymbol{q}$ have some common support, and by equation X $a + b < 0$ for any admissible $\boldsymbol{r}$, hence:

$$\boldsymbol{p} \cdot \boldsymbol{r} < 0, \tag{21}$$

as required.

II. Minimal and admissible bets exist if and only if $\boldsymbol{q} \neq \bar{\boldsymbol{q}}$. Suppose that $\boldsymbol{q} = \bar{\boldsymbol{q}}$. Then, by eq. X, an admissible bet must satisfy:

$$0 < \boldsymbol{r} \cdot \boldsymbol{q} \tag{22}$$

$$0 > \boldsymbol{r} \cdot \bar{v}q = \boldsymbol{r} \cdot \boldsymbol{q}, \tag{23}$$

which is a contradiction, and so $\boldsymbol{q} = \bar{\boldsymbol{q}}$ implies that no admissible bet exists.

Note that:

$$\begin{aligned}
\|\bar{\boldsymbol{q}}\|_2^2 &= \sum_{x^{(n)} \in \mathcal{X}^n} \left(\frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} q(\pi(x^{(n)})) \right)^2 \\
&\leq \sum_{x^{(n)} \in \mathcal{X}^n} \frac{1}{|\Pi_n|} \sum_{\pi \in \Pi_n} q(\pi(x^{(n)}))^2 && \triangle \text{ GM-HM inequality} \\
&= \sum_{x^{(n)} \in \mathcal{X}^n} q(x^{(n)})^2 = \|\boldsymbol{q}\|_2^2,
\end{aligned}$$

and so, $\boldsymbol{q} \cdot \boldsymbol{q} > \boldsymbol{q} \cdot \bar{\boldsymbol{q}}$ (strict inequality, as $\boldsymbol{q} \neq \bar{\boldsymbol{q}}$).

Hence, picking $a = \left(\frac{\boldsymbol{q} \cdot \bar{\boldsymbol{q}}}{\boldsymbol{q} \cdot \boldsymbol{q}} \right)^{\frac{1}{2}}$ and $b = -1$:

$$a(\boldsymbol{q} \cdot \boldsymbol{q}) + b(\boldsymbol{q} \cdot \bar{\boldsymbol{q}}) = (\boldsymbol{q} \cdot \bar{\boldsymbol{q}})^{\frac{1}{2}} \overbrace{(\boldsymbol{q} \cdot \boldsymbol{q})^{\frac{1}{2}}}^{> (\boldsymbol{q} \cdot \bar{\boldsymbol{q}})^{\frac{1}{2}}} - (\boldsymbol{q} \cdot \bar{\boldsymbol{q}}) > 0 \tag{24}$$

$$a + b = \underbrace{\left(\frac{\boldsymbol{q} \cdot \bar{\boldsymbol{q}}}{\boldsymbol{q} \cdot \boldsymbol{q}} \right)^{\frac{1}{2}}}_{<1} - 1 < 0, \tag{25}$$

gives an admissible (and minimal) bet as required.

■

Appendix B. Bayesian Linear Model Prediction Rule

$$\begin{aligned}
s_n \left(\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} \mid \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \ldots, \begin{bmatrix} x_n \\ y_n \end{bmatrix} \right) &= \mathcal{N}\left(y_{n+1}; \bar{\mu}, \bar{\Sigma} \right) \\
\bar{\mu} &= \left(x_{n+1} \Phi^\intercal + \sigma^2 \right) \left(\Phi \Phi^\intercal + \sigma^2 \right)^{-1} y_{1:n} \\
\bar{\Sigma} &= x_{n+1}^\intercal x_{n+1} - \left(x_n \Phi^\intercal + \sigma^2 \right) \left(\Phi \Phi^\intercal + \sigma^2 I \right)^{-1} \left(\Phi x_{n+1} + \sigma^2 \right) + \sigma^2
\end{aligned}$$

with $\Phi = \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix}^\intercal$ and $y_{1:n} = \begin{bmatrix} y_1 & \ldots & y_n \end{bmatrix}^\intercal$.

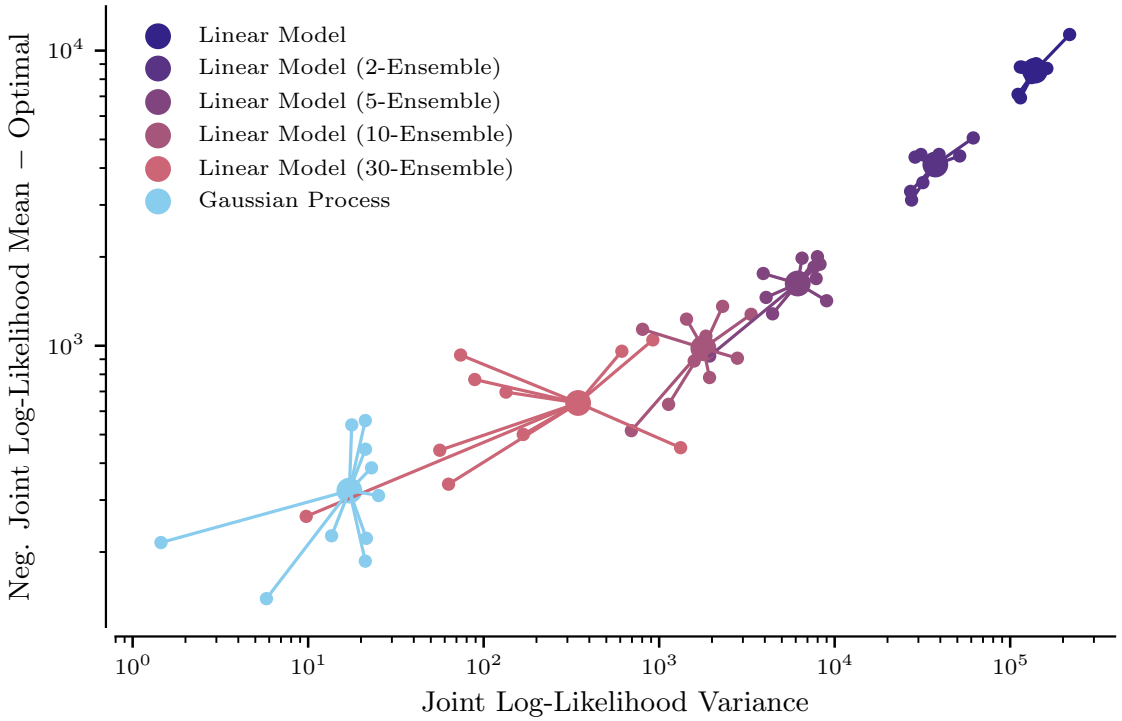

Figure 4: Joint log-likelihood variance as a measure of *implicit Bayesianness*(defined in (5)) vs. the negative sum of log-likelihood as a measure of *performance* (defined in (6)) plot for a Gaussian Process with a squared exponential kernel, and a linear model fit with the prior networks procedure on the regression task described in Section 4.2. The individual small dots depict the result for a particular collection of datapoints $((x_1, y_1), \ldots, (x_{100}, y_{100}))$ drawn from the data-generating distribution, and the large dots represent the mean over multiple dataset samples from the data-generating distribution. The expected negative log-likelihood of the *optimal* predictions on this dataset (i.e. using the true distribution) is subtracted from the mean negative joint log-likelihood plotted.

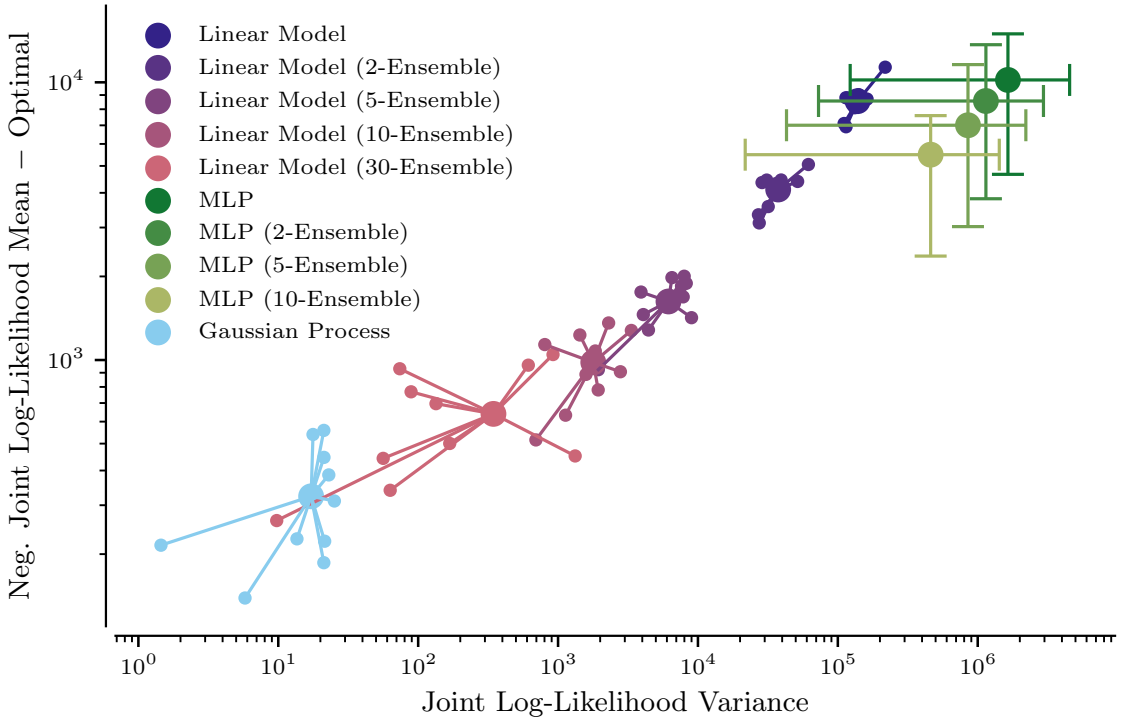

Figure 5: Comparison of the MLP results in Figure 6 and a Gaussian Process Figure 4 on the same plot.

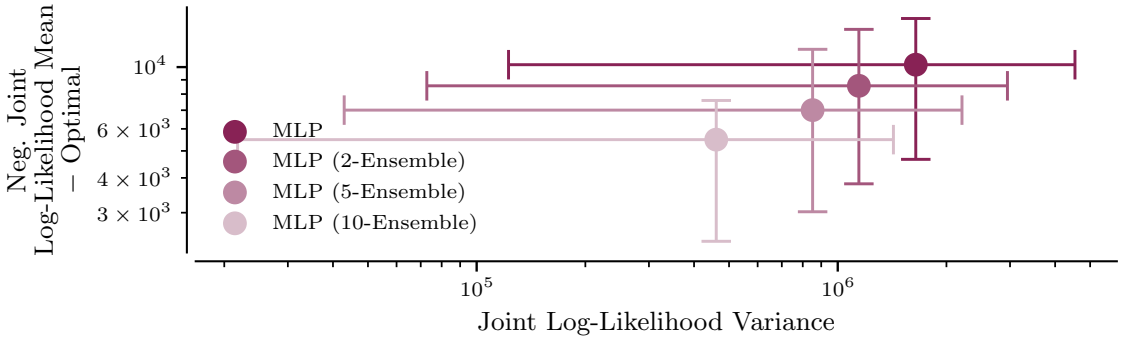

Figure 6: Joint log-likelihood variance as a measure of *implicit Bayesianness*(defined in (5)) vs. the negative sum of log-likelihood as a measure of *performance* (defined in (6)) plot for a MLP trained with SGD, as well as ensembles of such models, on the regression task described in Section 4.2. The error bars denote 10th and 90th percentiles over multiple dataset samples from the data-generating distribution.

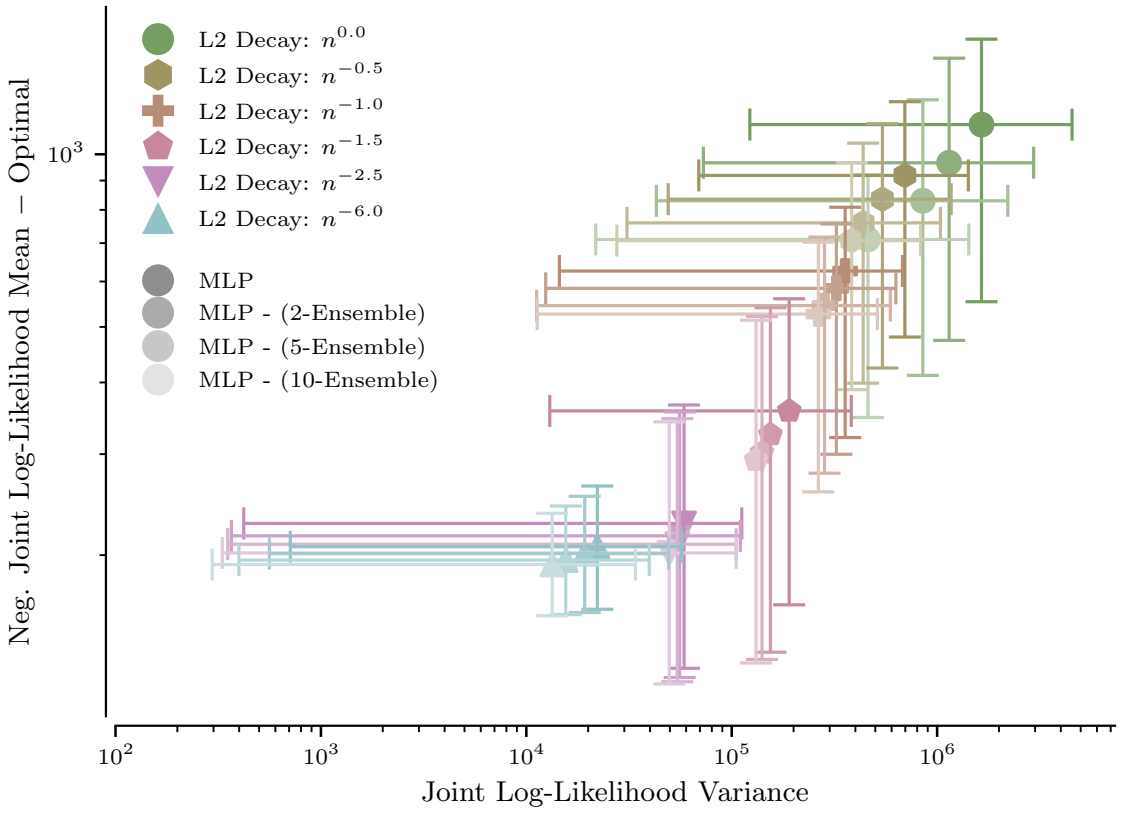

Figure 7: Comparison of different ℓ_2-decay schedules (with respect ot dataset size) for an MLP model trained with SGD measuring joint log-likelihood variance as a measure of *implicit Bayesianness*(defined in (5)) vs. the negative sum of log-likelihood as a measure of *performance* (defined in (6)). Results shown for the regression task described in Section 4.2. The error bars denote 10th and 90th percentiles over multiple dataset samples from the data-generating distribution.

Appendix C. Can a prediction rule at a fixed step be extended to an implicitely Bayesian prediction rule?

In deep learning, we would often taylor our approach to the amount of data available at hand. For example, we might use a small neural network for a small dataset, and a large neural network for a large dataset. If we have fewer than hundreds of datapoints, we might not even consider using a neural network at all. Hence, viewing training of a neural network as a prediction rule for all possible data sizes might not be the most natural lens.

One might wonder, assuming that one has only specified a prediction rule at step n only, can it be extended to a prediction rule for all n to satisfy some of the previously mentioned properties?

One reasonable guess might be that, if the prediction rule at step n is invariant to the ordering of the data, i.e. $s_n(\cdot|x_1, \ldots, x_n) = s_n(\cdot|x_{\pi(1)}, \ldots, x_{\pi(n)})$ for any permutation π of $\{1, \ldots, n\}$[9], then maybe it can be extended to an exchageable prediction rule. If that was the case, that would be good news: as long as we're making the prediction at step n only, we can claim that our prediction rule is implicitly Bayesian; once we start making predictions at other time-steps, we just need to figure out what an implicitely Bayesian extension is.

However, the invariance of the prediction rule at step n to the ordering of the data is not a sufficient condition for an implicitely Bayesian extension to exist. It's possible to devise simple counter-examples showing to the contrary.

Appendix D. Experimental Details

D.1. Gaussian Process and Prior Networks

For the Gaussian Process, we consider a squared exponential kernel of the form $k_{\text{SE}}(x, x') = a^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$. By default, the output variance a^2 and the length-scale ℓ are both set to 1. When the hyperparameters are being updated using margina likelihood, we set the ranges for both a^2 and ℓ to $(10^{-5}, 10^5)$ and optimise using LBFGS.

For the feature expansion for the Prior Network, we use a large number of Gaussian radial basis function features uniformly spaced over the relevant part of the input domain; in the limit, if the basis functions are adequately chosen, this should yield a Bayesian Linear Model equivalent to a squared exponential kernel GP Rasmussen and Williams (2005). Concretely, we use a Gaussian basis function $\phi_\beta(x) = c \exp(-\frac{(x-\beta)^2}{\ell^2})$ for an adequately chosen constant c (≈ 4.47) with 100 bases spaced equidistantly on the interval $[-6.0, 6.0]$.

The variance of the noise, both for the likelihood of the Gaussian Process and the linear model, were in both cases set to the true (known) variance for the task.

D.2. Ensembling

Models are ensembled by averaging their predictions: $\frac{1}{M}\sum_{j=1}^{M} s_i^{\epsilon_j}(x^*|x_1, \ldots, x_i)$ (equivalent to taking the `logmeanexp` operation of the log-probabilities for test data).

9. This is a weaker condition than exchangeability, as it only requires invariance to the ordering of the conditioning sequence.

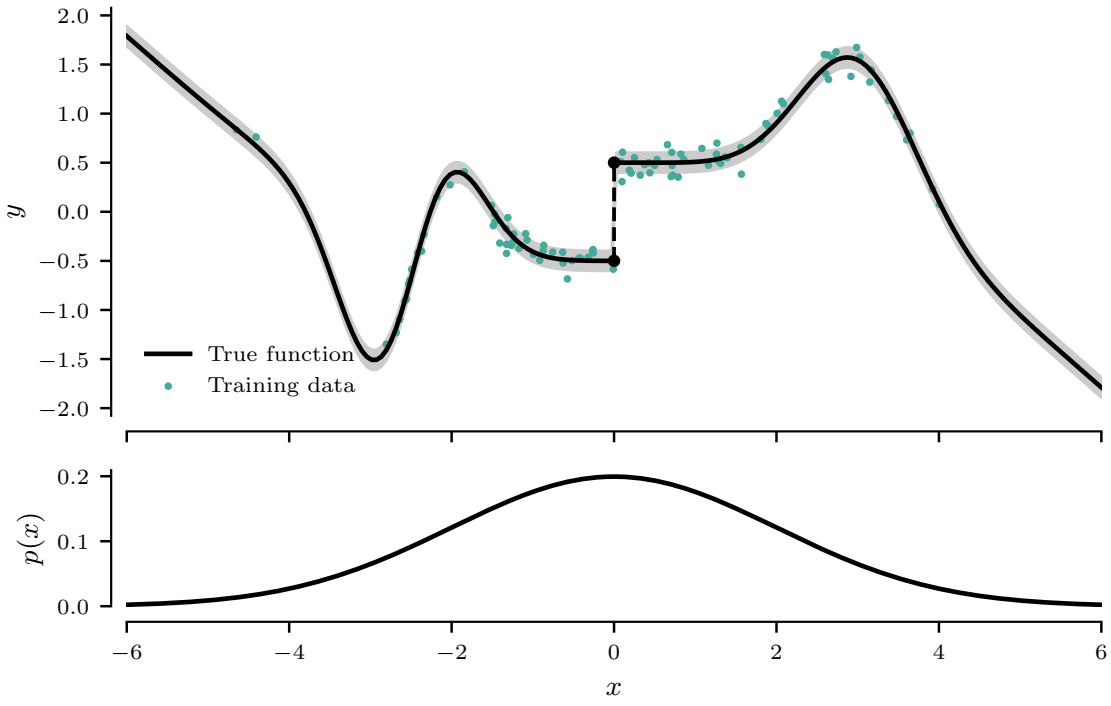

Figure 8: A 1D regression task. *(Bottom)*: The Gaussian density on the covariates. *(Top)*: The data-generating process illustrating the true function, with the filled area around it depicting the standard deviation of the homogeneous Gaussian noise.

D.3. MLP

The hyperparameters of the model (weight decay, learning rate) were tuned on a validation set, with a separate randomly sampled training set of size n, and fixed for all steps $i \leq n$ of the prediction rule s_i (except where explicitly stated that they were adjusted).

The neural network model considered in regression tasks is an MLP with three hidden layers, each of width 512, with a `ReLU` activation and no normalisation. The model was trained for 10000 iterations (independently of dataset size) with a batch-size of 64 (or less, if the dataset size is smaller).

D.3.1. ℓ_2-decay experiments

For different ℓ_2-decay experiments, we run the same experiments with same hyperparameters as above, but with a decay schedule in the dataset size i of the form $c_\alpha i^{-\alpha}$ for different values of α. c_α is in each case set so that the value of ℓ_2 decay would match for $i = n$, i.e. the final value of the weight-decay would be the optimal one as found with cross-validation.

For larger (negative) values of α, the initial weight-decay for small dataset sizes gets so large that the training becomes numerically unstable. Hence, we clip the maximum value of weight-decay to 2000, which is sufficient for the model to always learn the solution equivalent to all weights being set to 0.

For all results for all methods, for each dataset sequence $((x_1, y_1), \ldots, (x_n, y_n))$, we ignore the prediction for the first datapoint in the sequence (x_1, y_1), effectively setting the log-probability assigned by s_0 to 0 for all methods. This does contribute a small amount to increase in the variance of the log-joint even for exactly exchangeable methods. We do so as, for deep learning methods, the prediction with no training data is somewhat ill-defined.