INPUT-ADAPTIVE BAYESIAN MODEL AVERAGING

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper addresses prediction problems with multiple candidate models, where the goal is to combine their outputs. This task is especially challenging in heterogeneous settings, where different models may be better suited to different inputs. We propose Input-Adaptive Bayesian Model Averaging (IABMA), a Bayesian method that assigns model weights conditional on the input. IABMA employs an input-adaptive prior, and yields a posterior distribution that adapts to each prediction, which we estimate via amortized variational inference. We derive formal guarantees for its performance relative to any single predictor selected per input, and evaluate IABMA across regression and classification tasks, studying data from personalized cancer treatment, credit-card fraud detection, and UCI datasets. IABMA consistently delivers more accurate and better-calibrated predictions than both non-adaptive baselines and existing adaptive methods.

1 INTRODUCTION

Many modern applications require *adaptive predictions*. For instance, in personalized medicine, different patients may respond differently to the same treatment; in fairness-sensitive domains, predictions may need to adapt to subpopulations; and in fraud detection, behavioral data is often heteroskedastic and can vary substantially across inputs.

When the data is complex, selecting a single model that performs well across all inputs is particularly challenging. Moreover, doing so disregards the uncertainty inherent in model selection and often leads to overconfident predictions (Hoeting et al., 1999). A common strategy to mitigate this challenge is *model averaging* (MA), which produces an *ensemble* of models. Let $x \in \mathcal{X}$ be datapoints, $y \in \mathcal{Y}$ labels, and denote by $\mathcal{P}(\mathcal{Y})$ the space of probability distributions on $\mathcal{Y}$. Rather than relying on a single predictor $f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$, MA combines the predictive distributions of multiple models $f_1, \ldots, f_m$ into a weighted ensemble,

$$p_\alpha(y \mid x) \coloneqq \sum_{j=1}^{m} \alpha_j f_j(y \mid x), \tag{1}$$

thereby accounting for the possibility that multiple models can provide plausible explanations of the data.

However, this presents a new challenge: for some inputs, a subset of models may be poorly suited, while for others, a different subset may perform inadequately. Applying a *single global set of weights* across all inputs can therefore assign high weight to ill-suited models, degrading predictive performance. This motivates *adaptive averaging*, where the weights α_j are allowed to depend on the input x:

$$\alpha : \mathcal{X} \to \Delta^{m-1}, \qquad x \mapsto \alpha(x) = (\alpha_1(x), \ldots, \alpha_m(x)), \tag{2}$$

yielding the adaptive mixture

$$p_\alpha(y \mid x) \coloneqq \sum_{j=1}^{m} \alpha_j(x) f_j(y \mid x). \tag{3}$$

Previous adaptive approaches (see Section 1.1) addressed the task of specifying the adaptive weights $\alpha_j(x)$ from a frequentist point of view, often relying on maximum-likelihood estimates. In this work, we adopt a Bayesian perspective.

We assume that the set of predictors $\mathcal{F} \coloneqq \{f_1, \ldots, f_m\}$ is fixed, and model the selection of a model from this set as random. Thus, we introduce a selector function $g : \mathcal{X} \to \{e_1, \ldots, e_m\}$ with $\{e_j\}_{j=1}^m$ denoting the standard basis vectors of $\mathbb{R}^m$ so that $g(x) = e_j$ corresponds to selecting predictor f_j.

From a Bayesian perspective, the selector function g is random. A classical approach is thus to place a prior $p(g)$ on the space of selector functions $\mathcal{G} \coloneqq \{g : \mathcal{X} \to \{e_1, \ldots, e_m\}\}$. However, this ignores the intuition that different models may be preferable for different inputs.

Here we propose a probabilistic model, in which the identity of the selector function itself g depends on inputs x, implying an input-dependent prior $p(g \mid x)$. Unlike the classical formulation, which assigns a single global prior $p(g)$, and merely lets $g(x)$ vary across inputs, our approach directly models input-adaptive selection.

The joint distribution defined by our model yields a data-dependent posterior distribution that captures uncertainty over which predictors are most plausible for each input x. The resulting predictive distribution $p(y \mid x)$ then corresponds to a convex combination of the candidate models: the one identified by the posterior as most consistent with the data. Thus, we define the ensemble weights $\alpha_j(x)$ directly according to this posterior.

We analyze the advantages of this adaptive Bayesian model averaging framework, and derive finite-sample guarantees comparing its performance compared to that of any single predictor selected per input (Section 2.1). We then develop Input-Adaptive Bayesian Model Averaging (IABMA), a method that employs amortized variational inference to approximate the adaptive posterior (Section 3). This posterior depends jointly on the labeled training data $\mathcal{D}$ and the input x, and is induced by an input-adaptive, likelihood-based prior over selector functions. We evaluate IABMA across regression and classification benchmarks (Section 4), and show that IABMA achieves substantial gains in both accuracy and calibration compared to existing adaptive, and non-adaptive strategies.

1.1 Related Work

MA is regarded as the machine learning analogue of the "Condorcet's jury" theorem (Mennis, 2006), leveraging the "wisdom of the crowd" to mitigate the inherent uncertainty in model selection. Thus, MA is often used when there are alternative, potentially overlapping hypotheses and no clear justification for selecting a single preferred model. Applications include ecological research (Wintle et al., 2003; Thuiller, 2004; Richards, 2005; Dormann et al., 2008; Lauzeral et al., 2015; Zheng et al., 2024) and medicine (Jiang et al., 2021; Nanglia et al., 2022; Mahajan et al., 2023). More broadly, MA has been adopted in a wide range of machine learning tasks (e.g., Fernández-Delgado et al. (2014), Rokach (2010)).

As a form of model combination, MA is closely related to other ensemble techniques such as bagging (Breiman, 1996) and boosting (Freund, 1995). It is a variant of stacking procedure (Wolpert, 1992), in which outputs of base learners are combined to produce the final prediction.

MA has been shown to reduce prediction errors beyond those of the best individual component model (Dormann et al., 2018; Peng & Yang, 2022) and to mitigate overfitting (Dietterich et al., 2002; Polikar, 2006). In recent years, extensive surveys have reviewed MA (Kulkarni & Sinha, 2013; Woźniak et al., 2014; Gomes et al., 2017; González et al., 2020; Sagi & Rokach, 2018; Wu & Levinson, 2021), with some focusing specifically on decision trees (Rokach, 2016) or neural networks (Ganaie et al., 2022).

Most existing MA methods assign the same set of weights to all inputs. In contrast, dynamic model selection (Cao et al., 1995; Giacinto & Roli, 1999; Gunes et al., 2003; Didaci et al., 2005; Didaci & Giacinto, 2004) adapts the choice of model to each input instance. However, these methods select a single model rather than assigning instance-specific weights for averaging across multiple models.

Input-adaptive model averaging methods: Few methods assign input-dependent weights. These date back to Mixture of Experts (MoE) (Jacobs et al., 1991), where a gating network maps the input x to weights $\alpha_j(x)$, estimated by maxmizing the induced likelihood.

Rasmussen & Ghahramani (2001) extended this framework by using Gaussian Processes (GPs) as base models, providing nonparametric flexibility. They adopt a Bayesian perspective with a Dirichlet

Process (DP) prior, yielding an infinite mixture. However, here weights and base models are learned jointly, and thus only a single family of base predictors is considered.

Woods et al. (1997) proposed a dynamic scheme based on local accuracy estimates. For a test input x, its neighborhood is identified (typically via k-nearest neighbors), and each classifier's performance in this region is summarized as a local accuracy score. The classifier with the highest score is then selected to predict x.

Similarly, Chan & van der Schaar (2022) proposed an approach that assigns higher weight to models whose training domains better cover a test instance. Inputs are mapped into a learned low-dimensional space where models with similar predictions are closer together, and weights are set via kernel density estimation. Unlike Woods et al. (1997), where similarity is predefined, here it is learned from data. Motivated by Tenzer et al. (2022), the method assumes that models making random errors on an input are unlikely to agree.

Perhaps most relevant to our work is Bayesian hierarchical stacking (BHS) (Yao et al., 2022), which places priors on logit weights, and models them with hierarchical low-rank linear functions. The parameters are then estimated by maximizing the expected log predictive density.

Thus, prior work on adaptive model averaging has focused predominantly on methods targeting frequentist objectives, with relatively few Bayesian formulations. In contrast to previous approaches, our model assumes a fully Bayesian setting in which the selector itself is random and, crucially, is defined locally relative to each input x. This yields an input-dependent prior $p(g \mid x)$ rather than a global prior $p(g)$. In turn, this prior induces an adaptive posterior that corresponds exactly to the Bayes-optimal weights, providing a principled approach for adaptive model averaging.

2 Probabilistic formulation of adaptive model averaging

We cast adaptive model averaging as a probabilistic model selection. To reflect that some models may be better suited for different inputs, we assume the a probabilistic model where the *selection function* g is random and *input-dependent*:

$$x \sim p(x), \qquad g \sim p(g \mid x), \qquad y \sim p(y \mid x, g). \tag{4}$$

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ be a training set of i.i.d samples from a population distribution $p(x, y)$ on $\mathcal{X} \times \mathcal{Y}$, and consider a new input x. Under our formulation, the full joint distribution is

$$p(x, y, \mathcal{D}, g) = p(x) \prod_{i=1}^{n} [p(x_i)]\, p(g \mid x, x_{1:n}) \prod_{i=1}^{n} [p(y_i \mid x_i, g)]\, p(y \mid x, g). \tag{5}$$

We defer the precise specification of the adaptive prior $p(g \mid x, x_{1:n})$ to Section 3.1.

The predictive distribution for y given a new input x is then

$$p(y \mid x, \mathcal{D}) = \int p(y \mid x, \mathcal{D}, g)\, p(g \mid x, \mathcal{D})\, d\mu(g) = \int p(y \mid x, g)\, p(g \mid x, \mathcal{D})\, d\mu(g), \tag{6}$$

where $p(g \mid x, \mathcal{D})$ is a posterior distribution on the space of measurable functions $\mathcal{G} \coloneqq \{g : \mathcal{X} \to \{e_1, \ldots, e_m\}\}$[1].

A draw from the posterior $g \sim p(g \mid x, \mathcal{D})$ induces a random index $J(x)$, defined by the relation $g(x) = e_{j(x)}$. Using this index, we can rewrite equation 6 as

$$p(y \mid x, \mathcal{D}) = \int p(y \mid x, g)\, p(g \mid x, \mathcal{D})\, d\mu(g) = \sum_{j=1}^{m} f_j(y \mid x)\, p(J(x) = j \mid x, \mathcal{D}), \tag{7}$$

where the last equality follows by a standard change of measure argument, which for completeness is outlined in Appendix A.1.

This derivation shows that, under our model, the predictive distribution is a mixture of the candidate predictions $f_j(y \mid x)$ weighted by the push-forward posterior probabilities $p(J(x) = j \mid x, \mathcal{D})$. In other words, the input-adaptive weights $\alpha_j(x)$ arise directly from the probabilistic formulation itself, and they are precisely the posterior probabilities of each model being the generator at input x.

[1] Formally, $p(g \mid x, \mathcal{D})$ is a density with respect to some reference measure μ on $\mathcal{G}$.

2.1 LIKELIHOOD GUARANTEES

So far we have seen that the posterior probabilities $p(J(x) = j \mid x, \mathcal{D})$ arise naturally as input-adaptive weights under our model. In particular, they are the Bayes-optimal weights, as they recover the true predictive distribution.

We now show that this choice also comes with performance guarantees: the posterior-weights predictor not only reflects the correct probabilistic formulation, but in expectation achieves likelihood performance competitive with any input-specific single-model selector. The next theorem formalizes this result (for proof see Appendix A.2).

Theorem 2.1. *Denote $\mathcal{D}_i \coloneqq \{(x_t, y_t)\}_{t=1}^{i}$, and consider the posterior weights predictor $\hat{p}_\alpha^{(i)}$ assigning $\alpha_j(x; \mathcal{D}_i) = p(J(x) = j \mid \mathcal{D}_i, x)$ to the j-th predictor f_j. Assume that $\mathbb{E}[|\log f_j(Y \mid X)|] < \infty$ for all $f_j \in \mathcal{F}$. Then, for any measurable selector $j^* : \mathcal{X} \to \{1, \dots, m\}$ and any $n \geq 1$,*

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \hat{p}_\alpha^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1})\right] \geq \mathbb{E}\left[\log f_{j^*(x)}(y \mid x)\right] + \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \alpha_{j^*(x_i)}^{(i)}(x_i)\right], \quad (8)$$

where the expectations are taken w.r.t the population distribution $(x_i, y_i) \sim p(x, y)$.

Thus, the posterior weights predictor can match any per-input selector (i.e., a rule that may pick a different j for different x), up to a term depending on the gating weights assigned to the chosen model at each x.

Concretely, for the selector that picks the most probable model, $j^{(i)}(x) \in \arg\max_{1 \leq j \leq m} \alpha_j^{(i)}(x)$, the penalty becomes $\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\Big[\log \max_j \alpha_j^{(i)}(x_i)\Big]$, which vanishes as the posterior sharpens, i.e., when $\max_j \alpha_j^{(i)}(x_i) \to 1$ (in probability or almost surely).

A central difficulty, of course, is that the true posterior is unknown. Thus, in the next section, we introduce a variational approximation to $p(J(x) = j \mid \mathcal{D}_i, x)$ that preserves explicit dependence on both x and $\mathcal{D}$.

3 IABMA: INPUT-ADAPTIVE BAYESIAN MODEL AVERAGING

Our *goal* is to develop a method for estimating this posterior distribution over models. By doing so, we obtain an averaging scheme that is consistent with both the training data $\mathcal{D}$ and the specific input x, thereby approximating the true predictive distribution that we ultimately aim to recover.

We begin by formulating the modeling assumptions for an adaptive prior that is conditioned jointly on the training covariates and a new input. Building on this prior, we then develop a variational inference method to approximate the resulting posterior.

3.1 ADAPTIVE PRIOR

Based on the adaptive prior introduced in (Slavutsky & Blei, 2025), we posit a prior that encodes the plausibility of each model conditional on both the training covariate $x_{1:n}$ and a new input x at which prediction is sought. This prior is defined through an energy-based formulation.

Specifically, for a predictor f_j we consider the prior induced by the negative energy function

$$E(J = j; x_{1:n}, x) \coloneqq \int \sum_{i=1}^{n} \log p(y|x_i, f_j) + \log p(y|x, f_j)\, dy \quad (9)$$

$$p(J = j|x_{1:n}, x) \coloneqq \frac{1}{Z(f)} \exp\left(E(J = j; x_{1:n}, x)\right), \quad (10)$$

where the normalizing factor[2] is given by $Z(f) \coloneqq \sum_{j=1}^{m} \exp\left(E(J = j; x_{1:n}, x)\right)$.

[2]This definition requires integrability of $\exp\left(E(J = j; x_{1:n}, x)\right)$, and thus we assume that $\exp(E(J = j; x_{1:n}, x))$ is integrable for each j.

This prior allows beliefs about model plausibility to adapt to the new input x. Unlike a prior defined solely from the training data, which remains fixed across prediction points, our formulation updates the relative weight of each model once x is observed. This makes the prior *input-adaptive*, enabling model selection probabilities to shift dynamically with the prediction covariates. To build intuition, we next examine a simple analytical example.

A two-model Bernoulli example Suppose $y \in \{0, 1\}$, and consider two candidate logistic models

$$p(y = 1 \mid x, f_j) = \sigma(\beta_j x), \qquad \sigma(u) \coloneqq \frac{1}{1 + e^{-u}}, \tag{11}$$

with $j \in 1, 2$ and slopes $0 < \beta_2 < \beta_1$. In this setting, the energy function is given by

$$E(J = j; x_{1:n}, x) = \sum_{i=1}^{n} \sum_{y \in \{0,1\}} \log p(y \mid x_i, f_j) + \sum_{y \in \{0,1\}} \log p(y \mid x, f_j) \tag{12}$$

$$= \underbrace{\sum_{i=1}^{n} \log \left(\sigma(\beta_j x_i) \left[1 - \sigma(\beta_j x_i)\right]\right)}_{=:C_j} + \underbrace{\log \left(\sigma(\beta_j x) \left[1 - \sigma(\beta_j x)\right]\right)}_{=:\ell_j(x)} \tag{13}$$

and the adaptive prior is

$$p(J = j \mid x_{1:n}, x) = \frac{\exp(C_j + \ell_j(x))}{\sum_{k=1}^{m} \exp(C_k + \ell_k(x))} \tag{14}$$

Accordingly, the log-odds between the two models is

$$\log \frac{p(J = 1 \mid x_{1:n}, x)}{p(J = 2 \mid x_{1:n}, x)} = (C_1 - C_2) + \ell_1(x) - \ell_2(x). \tag{15}$$

Thus, the log-odds depend both on the difference between training baselines $C_1 - C_2$, and the change induced by conditioning also on the new input x is $\delta_x \coloneqq \ell_1(x) - \ell_2(x)$.

Concretely, suppose the baseline difference is fixed at $C_1 - C_2 = \log 5 \approx 1.61$, yielding $p(J = 1 \mid \mathcal{D}) = \sigma(\log 5) \approx 0.83$. Based solely on the training data, the prior thus strongly favors f_1. Now consider a new input $x = 1$ with $\beta_2 = 1$. As β_1 increases, the discrepancy $|\ell_1(1) - \ell_2(1)|$ grows, and $\delta_{x=1}$ shifts the likelihood ratio toward f_2. For example, when $\beta_1 = 3$, the prior shifts to a mild preference for f_1, at $\beta_1 = 5$ it flips to favor f_2, and by $\beta_1 = 9$ the preference for f_2 becomes very strong. These dynamics, along with additional parameter settings for coefficients and baseline differences, are shown in Figure 1.

This analysis highlights the interplay between the baseline preference and the input-specific adjustment introduced by x. It shows that, in extreme cases, even strong baseline beliefs can be overturned by the adaptive correction at the queried input.

Evaluation of the prior: Evaluating the proposed prior requires computing an integral over the outcome space $\mathcal{Y}$, and thus depends on whether the outcome space is discrete or continuous. When $\mathcal{Y}$ is *discrete* (e.g., in classification problems), the integral reduces to a finite sum over all possible outcome values. In this case, the evaluation is straightforward and can be computed exactly without approximation. When $\mathcal{Y}$ is *continuous*, (e.g., in regression problems), the integral cannot typically be computed in closed form and may even diverge unless we restrict the domain of integration. Thus, to approximate the prior, as in (Slavutsky & Blei, 2025), we employ Monte-Carlo integration where we sample K possible outcome values uniformly from a predefined integration range $[y_{\min}, y_{\max}]$ and average the Normal log-likelihood (centered at the model's prediction with unit variance) over the K samples.

3.2 Amortized Variational Posterior

Equipped with the adaptive prior, we now turn to the estimation of the posterior $p(J = j \mid x_{1:n}, y_{1:n}, x) = p(J = j \mid \mathcal{D}, x)$, which conditions not only on the covariates x and $x_{1:n}$, but also on the training labels $y_{1:n}$. This, in turn, will enable us to assign input-adaptive weights for

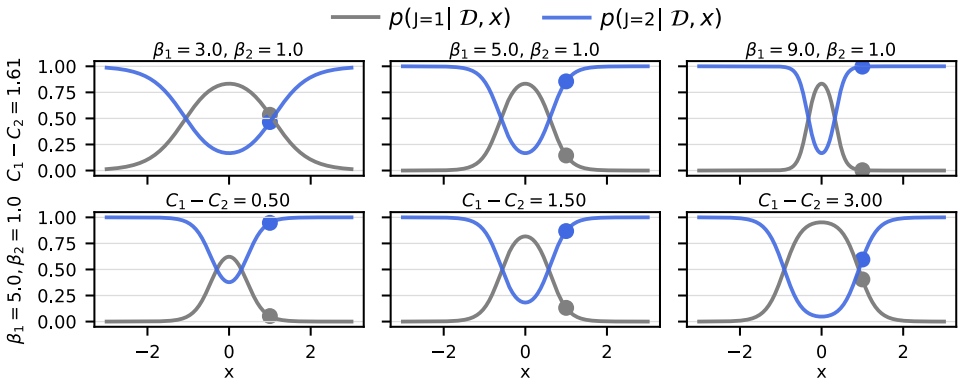

Figure 1: Illustration of the input-adaptive prior. Each panel shows the posterior probabilities $p(J = j \mid \mathcal{D}, x)$ as functions of x. *Top*: the baseline log-odds is fixed and β_1 varies; larger β_1 values increase the influence of x, producing stronger adaptive corrections. *Bottom*: β_1, β_2 are fixed while the baseline log-odds $C_1 - C_2$ varies; stronger baselines yield higher prior preference for f_1, but input-specific corrections can still substantially reshape the prior at certain x. The marked point ($x = 1$) highlights how the adaptive prior shifts the relative model probabilities compared to the baseline.

model averaging, bringing them closer to the ideal weights that recover the predictive distribution $p(y \mid x)$.

We do so by fitting variational distributions $q(f_j; x) \approx p(J = j \mid \mathcal{D}, x)$ parameterized as functions of the input x. This yields an *amortized posterior approximation*, which allows us to efficiently evaluate approximate posteriors at multiple inputs x.

In our case, in the context of a new input x, the true posterior distribution over predictors is Multinomial $p(J = j \mid \mathcal{D}, x) = \rho_j(x)$ for $j \in \{1, \ldots, m\}$, where each $\rho_j(x) > 0$ and $\sum_{j=1}^{m} \rho_j(x) = 1$. Thus, we set the variational family to be the set of all multinomial distributions.

$$\mathcal{Q}_x \coloneqq \{q = (q(J = 1; x), \ldots, q(J = m; x) \in \Delta^{m-1}\}. \tag{16}$$

For a given input x, our goal is to minimize the KL divergence

$$\min_{q \in \mathcal{Q}_x} D_{\mathrm{KL}}(q \| p) \coloneqq \sum_{j=1}^{m} q(J = j; x) \log \frac{q(J = j; x)}{p(J = j \mid \mathcal{D}, x)}. \tag{17}$$

Note that since the true posterior and the variational family share the same (categorical) form, the problem is well-specified: the KL depends only on estimating the probabilities $P(J = j; x)$.

3.3 Optimization

To minimize the KL divergence in Equation 17, we optimize the evidence lower bound (ELBO) on the log-likelihood (Kingma & Welling, 2014; Rezende & Mohamed, 2015; Blei et al., 2017). We parameterize the variational distribution with a neural network with weights θ, producing $h_\theta(x) = (q_\theta(J = 1; x), \ldots, q_\theta(J = m; x))$, and optimize θ rather than the output directly. Thus, our objective to fit the amortized posterior is

$$\mathcal{L}(\theta; x) = \mathbb{E}_{q_\theta}[\log p(y \mid x, f_j)] - D_{\mathrm{KL}}(q_\theta \| p(J | x_{1:n}, x)) \tag{18}$$

$$= \sum_{j=1}^{m} [q_\theta(J = j; x) \log f_j(y \mid x)] - \sum_{j=1}^{m} q_\theta(J = j; x) \log \frac{q_\theta(J = j; x)}{p(J = j \mid x_{1:n}, x)}. \tag{19}$$

Note that the expected log-likelihood $\mathbb{E}_{q_\theta}[\log p(y \mid x, f_j)]$ reduces to a weighted sum, so no sampling is required to evaluate our objective. The complete optimization procedure is summarized in Algorithm 1.

Algorithm 1 IABMA: Amortized Posterior Learning (IABMA)

Inputs: Training data $\mathcal{D}$; predictors $\{f_j\}_{j=1}^m$; initialization θ_0; learning rate η; iterations K.
Precompute: For all $i = 1, \ldots, n$ and predictor $j = 1, \ldots, m$, store $\log f_j(y_i \mid x_i)$.
for $k = 1$ **to** K **do**
  for $i = 1$ **to** n **do**
    for $j = 1$ **to** m **do**
      Prior: Compute $p(J = j|x_{-i}, x) \propto \exp\big(E(J = j; x_{-i}, x_i)\big)$
      Posterior: Compute $h_{\theta_{k-1}}(x) = (q_{\theta_{k-1}}(J = 1; x_i), \ldots, q_{\theta_{k-1}}(J = m; x_i))$
      ELBO: Compute

$$\mathcal{L}(x_i; \theta_{k-1}) = \sum_{j=1}^{m} q_{\theta_{k-1}}(J = j; x_i) \log f_j(y_i \mid x_i) - \sum_{j=1}^{m} q_{\theta_{k-1}}(J = j; x_i) \log \frac{q_{\theta_{k-1}}(J = j; x_i)}{p(J = j|x_{-i}, x_i)}.$$

    end for
    Update: $\overline{\mathcal{L}}(\theta_{k-1}) \leftarrow \frac{1}{n} \sum_i \mathcal{L}(x_i; \theta_{k-1})$
  end for
  Update: $\theta_k \leftarrow \theta_{k-1} + \eta \nabla_\theta \overline{\mathcal{L}}(\theta_{k-1})$
end for
Return: $\hat{\theta} := \theta_K$

Weight assignment: After training is complete (see Algorithm 1), with the estimate $\hat{\theta}$, for a new input x we compute $(q_{\hat{\theta}}(J = 1; x), \ldots, q_{\hat{\theta}}(J = m; x))$ and assign $\alpha_j(x) = q_{\hat{\theta}}(J = j; x)$. This yields a predicted value $\hat{p}_\alpha(y \mid x) = \sum_{j=1}^m \alpha_j(x) f_j(y \mid x)$.

4 Experiments

We evaluate IABMA via 7 experiments on classification and regression tasks, based on simulated data, 2 case-studies, and 4 UCI benchmark datasets.

We compare predictive distributions against (a) non-adaptive baselines: (i) best single predictor, (ii) uniform average over predictors, (iii) accuracy-weighted average, and (iv) classical Bayesian model averaging (BMA); and (b) adaptive methods (see Section 1.1): (i) Mixture of Experts (MoE) (Jacobs et al., 1991), (ii) Dynamic Local Accuracy (DLA) (Woods et al., 1997), (iii) Synthetic Model Combination (SMC) (Rasmussen & Ghahramani, 2001), and (iv) Bayesian Hierarchical Stacking (BHS) (Yao et al., 2022).

In each experiment we train candidate predictors and fit all model averaging methods on the training set. We test the performance of the resulting predictive distributions: for regression tasks we report R^2 and root mean square error (RMSE); for classification tasks we report accuracy and expected calibration error (ECE).

Hyperparameters for our method and all baselines were tuned via binary search to maximize average performance (accuracy for classification, RMSE for regression) on a held-out repetition excluded from the analysis. The selected values and further implementation details are provided in Appendix D, with additional data processing and predictor specifications in Section C. Code to reproduce all results is included with the submission and will be released publicly upon acceptance.

4.1 Simulation

We start with evaluating IABMA on a simple two-dimensional binary task: half of the observations were drawn from a Gaussian cluster centered at $(-1, 0)$, and labels were assigned by a linear rule $y = \mathbb{1}_{x_1+x_2>-1}$. The remaining observations were sampled around $(1, 0)$ on a ring; labels followed a circular rule $y = \mathbb{1}_{r<1}$ with $r = \sqrt{x_1^2 + x_2^2}$ measured from $(1, 0)$. We generated $n_{\text{train}} = 1,000$ and $n_{\text{test}} = 500$ examples and used only the 2-dimensional coordinates as features (region indicators were recorded for analysis but not used for training).

This example illustrates three subpopulations defined by their first dimension: (i) inputs that are perfectly separable by a linear boundary, (ii) inputs that are perfectly separable by a circular boundary,

and (iii) inputs for which it is ambiguous whether they belong to the first or the second group. Thus, and ideal weighting would put all weight on the best predictor for subpopulations (i) and (ii), and employ soft weighting at (iii). All averaging methods operated on the same base classifiers: polynomial logistic regression (degrees 2 and 3), Linear Discriminant Analysis, and tailored "soft-circle" models that predict class probability via a logistic function of radial distance to a learned center. Additional details are provided in section C.1

Results: Figure 2 shows that IABMA achieves highest accuracy and lowest ECE compared to all non-adaptive baselines, as well as all adaptive methods.

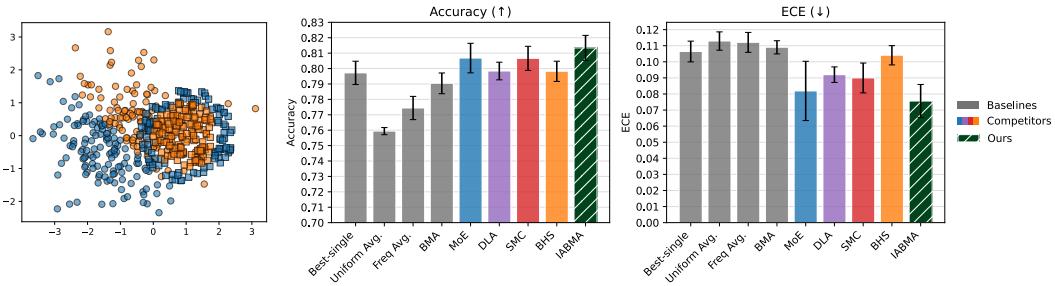

Figure 2: Simulation. Left: data (of one repetition). Results for accuracy (middle) and ECE (right) are reported for 10 repetitions. IABMA achieves highest accuracy and lowest ECE.

4.2 Case studies

4.2.1 Personalized cancer drug-response

An important example of heterogeneous data is personalized drug response prediction, where different models may perform better on different subpopulations. We evaluate IABMA on this task using the PRISM cancer drug response dataset. The data consists of pairings of molecule-cell line RNA sequence features. For each drug–cell pair we form a continuous response y so that larger values indicate greater sensitivity. We retain drugs with broad site coverage and construct inputs from the top variance genes. All averaging methods operate over the same four base regressors—Ridge, Histogram-based Gradient Boosting Tree, XGBoost, and a Multilayer perceptron (MLP), each with pre-processing tailored to model class. Additional details are provided in C.2.

Results: Figure 3 shows that IABMA achieves higher R^2 and lower RMSE compared to all other methods. Further analysis is presented in Figures 4–7 which display the weights assigned by each averaging method for randomly selected inputs. The results show that IABMA consistently favored the best (or nearly best) model, whereas other methods leaned toward other predictors, with MoE in particular overemphasizing MLP and XGB even when suboptimal.

4.2.2 Credit-card fraud detection

Another domain characterized by heterogeneous data is fraud detection, where the rarity of fraudulent cases poses an additional challenge. We evaluate IABMA on this task using the IEEE-CIS Fraud Detection dataset. The dataset consists from mixed Continuous (such as transaction amount) and high-cardinality categorical features (such as product category), and the target variable $y \in \{0, 1\}$ indicated where a transaction was fraud. All averaging methods operate over the same base classifiers: Logistic Regression with Lasso penalty, Histogram-based Gradient Boosting Tree, XGBoost (with class-imbalance weighting), and an MLP. Additional details are provided in Section C.3.

Results: Figure 3 shows that IABMA achieves higher accuracy and lower expected-calibration error compared to all other methods. Since in fraud prediction calibration matters within each bin, we analyzed per-bin confidence $|p - 0.5|$, and found that IABMA achieves the lowest error in all high-confidence bins (> 0.25). The corresponding analysis is shown in Figure B.2.

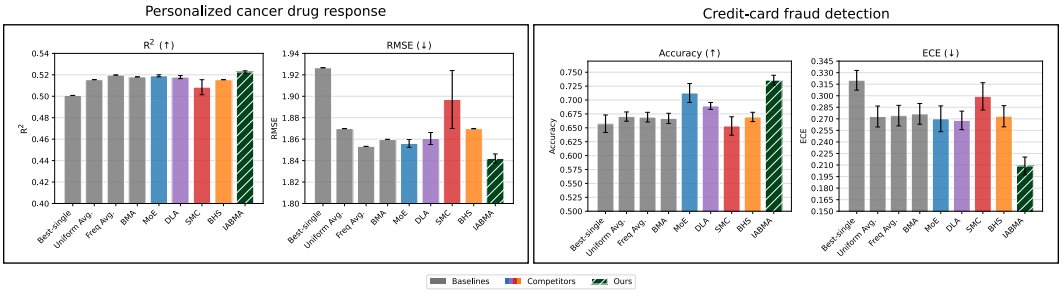

Figure 3: Experimental results for main case studies. Results are reported for 10 repetitions. IABMAachieves best results compared to all other averaging method on both case studies.

4.3 Experiments on UCI benchmark datasets

We evaluated IABMA on 4 UCI benchmark datasets spanning both classification and regression. For *classification* we used `spambase` and `credit-g`; for *regression* we used `bike-sharing` and `california-housing`. All pre-processing information is detailed in Section C.4

Across all datasets we trained a common set of base learners. For classification: Multinomial Naive Bayes, k–NN ($k = 3$), Random Forest, Extra Trees, and a linear SVM. For regression: Ridge (α=0.05), Lasso (α=0.05), k–NN (k=3, distance-weighted), Random Forest, and Extra-Trees. To encourage diversity, each model was trained on a subset of features ("feature bundles"). Full pre-processing steps, feature bundles, and model specifications are detailed in Section C.4.

Results: Unlike complex heterogeneous datasets, UCI benchmarks are highly normalized and thus show less variability across methods. Even so, IABMA outperforms all competitors in all four experiments on at least one metric, and in two cases on both. Results are reported in Table 1.

5 Conclusion

daptive model averaging in heterogeneous data settings, where different predictors may be preferable for different inputs. We introduced IABMA, framework that casts model averaging as probabilistic model selection conditioned on the input. Within this formulation, the posterior distribution over models provides the natural, Bayes-optimal choice of input-adaptive weights, thereby recovering the true predictive distribution. Our approach is grounded in an input-dependent prior on the selector function and implemented through amortized variational inference of the posterior.

We establish finite-sample bounds showing that the posterior-weights predictor achieves strong likelihood performance compared to any input-specific single-model selector. Empirically, we evaluate IABMA across regression and classification tasks, including personalized cancer treatment response, credit-card fraud detection, and standard UCI benchmarks. We show that IABMA consistently outperforms both non-adaptive baselines and existing adaptive methods, delivering more accurate and better calibrated predictions.

REFERENCES

David M Blei, Alp Kucukelbir, and Jon McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Jun Cao, Majid Ahmadi, and Malayappan Shridhar. Recognition of handwritten numerals with multiple feature and multistage classifier. *Pattern Recognition*, 28(2):153–160, 1995.

Alex Chan and Mihaela van der Schaar. Synthetic model combination: An instance-wise approach to unsupervised ensemble learning. *Advances in Neural Information Processing Systems*, 35: 27797–27809, 2022.

Luca Didaci and Giorgio Giacinto. Dynamic classifier selection by adaptive k-nearest-neighbourhood rule. In *Proceedings of the International Workshop on Multiple Classifier Systems*, pp. 174–183. Springer, 2004.

Luca Didaci, Giorgio Giacinto, Fabio Roli, and Gian Luca Marcialis. A study on the performances of dynamic classifier selection based on local accuracy estimation. *Pattern Recognition*, 38(11): 2188–2191, 2005.

Thomas G Dietterich et al. Ensemble learning. *The Handbook of Brain Theory and Neural Networks*, 2(1):110–125, 2002.

Carsten F Dormann, Oliver Schweiger, Paul Arens, Isabel Augenstein, S. T. Aviron, Debra Bailey, Jacques Baudry, Regula Billeter, Rob Bugter, Roman Bukacek, et al. Prediction uncertainty of environmental change effects on temperate european biodiversity. *Ecology Letters*, 11(3):235–244, 2008.

Carsten F Dormann, Justin M Calabrese, Gurutzeta Guillera-Arroita, Eleni Matechou, Volker Bahn, Kamil Bartoń, Colin M Beale, Simone Ciuti, Jane Elith, Katharina Gerstner, et al. Model averaging in ecology: A review of bayesian, information-theoretic, and tactical approaches for predictive inference. *Ecological Monographs*, 88(4):485–504, 2018.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

Yoav Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121 (2):256–285, 1995.

Mudasir A Ganaie, Minghui Hu, Ashwani Kumar Malik, Muhammad Tanveer, and Ponnuthurai N Suganthan. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115:105151, 2022.

Giorgio Giacinto and Fabio Roli. Methods for dynamic classifier selection. In *Proceedings of the 10th International Conference on Image Analysis and Processing*, pp. 659–664. IEEE, 1999.

Heitor Murilo Gomes, Jean Paul Barddal, Fabrício Enembreck, and Albert Bifet. A survey on ensemble learning for data stream classification. *ACM Computing Surveys*, 50(2):1–36, 2017.

Sergio González, Salvador García, Javier Del Ser, Lior Rokach, and Francisco Herrera. A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Information Fusion*, 64:205–237, 2020.

Veyis Gunes, Michel Menard, Pierre Loonis, and Simon Petit-Renaud. Combination, cooperation and selection of classifiers: A state of the art. *International Journal of Pattern Recognition and Artificial Intelligence*, 17(08):1303–1324, 2003.

Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: A tutorial (with comments by m. clyde, david draper and ei george, and a rejoinder by the authors). *Statistical Science*, 14(4):382–417, 1999.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

Zhencun Jiang, Zhengxin Dong, Lingyang Wang, and Wenping Jiang. Method for diagnosis of acute lymphoblastic leukemia based on vit-cnn ensemble model. *Computational Intelligence and Neuroscience*, 2021(1):7529893, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

Vrushali Y Kulkarni and Pradeep K Sinha. Random forest classifiers: A survey and future research directions. *International Journal of Advanced Computer*, 36(1):1144–1153, 2013.

Christine Lauzeral, Gaël Grenouillet, and Sébastien Brosse. The iterative ensemble modelling approach increases the accuracy of fish distribution models. *Ecography*, 38(2):213–220, 2015.

Palak Mahajan, Shahadat Uddin, Farshid Hajati, and Mohammad Ali Moni. Ensemble learning for disease prediction: A review. In *Healthcare*, volume 11, pp. 1808. MDPI, 2023.

Edmund A Mennis. The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies, and nations. *Business Economics*, 41 (4):63–65, 2006.

S Nanglia, Muneer Ahmad, Fawad Ali Khan, and N. Z. Jhanjhi. An enhanced predictive heterogeneous ensemble model for breast cancer prediction. *Biomedical Signal Processing and Control*, 72:103279, 2022.

Jingfu Peng and Yuhong Yang. On improvability of model selection by model averaging. *Journal of Econometrics*, 229(2):246–262, 2022.

Robi Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6 (3):21–45, 2006.

Carl Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. *Advances in Neural Information Processing Systems*, 14, 2001.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the International Conference on Machine Learning*, pp. 1530–1538. PMLR, 2015.

Shane A Richards. Testing ecological theory using the information-theoretic approach: Examples and cautionary results. *Ecology*, 86(10):2805–2814, 2005.

Lior Rokach. *Pattern Classification Using Ensemble Methods*, volume 75. World Scientific, 2010.

Lior Rokach. Decision forest: Twenty years of research. *Information Fusion*, 27:111–125, 2016.

Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

Yuli Slavutsky and David M Blei. Quantifying uncertainty in the presence of distribution shifts. *Advances in Neural Information Processing Systems*, 2025.

Yaniv Tenzer, Omer Dror, Boaz Nadler, Erhan Bilal, and Yuval Kluger. Crowdsourcing regression: A spectral approach. In *International Conference on Artificial Intelligence and Statistics*, pp. 5225–5242. PMLR, 2022.

Wilfried Thuiller. Patterns and uncertainties of species' range shifts under climate change. *Global Change Biology*, 10(12):2020–2027, 2004.

Brendan A Wintle, Michel A McCarthy, Chris T Volinsky, and Rodney P Kavanagh. The use of bayesian model averaging to better represent uncertainty in ecological models. *Conservation Biology*, 17(6):1579–1590, 2003.

David H Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

Kevin Woods, W. Philip Kegelmeyer, and Kevin Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19 (4):405–410, 1997.

Michał Woźniak, Manuel Grana, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17, 2014.

Hao Wu and David Levinson. The ensemble approach to forecasting: A review and synthesis. *Transportation Research Part C: Emerging Technologies*, 132:103357, 2021.

Yuling Yao, Gregor Pirš, Aki Vehtari, and Andrew Gelman. Bayesian hierarchical stacking: Some models are (somewhere) useful. *Bayesian Analysis*, 17(4):1043–1071, 2022.

Yue Zheng, Jun Wei, Wenming Zhang, Yiping Zhang, Tuqiao Zhang, and Yongchao Zhou. An ensemble model for accurate prediction of key water quality parameters in river based on deep learning methods. *Journal of Environmental Management*, 366:121932, 2024.

A PROOFS

A.1 CHANGE OF MEASURE ARGUMENT

Let $F : X_2 \to X_1$ be a measurable function between two measure spaces $(X_1, \mathcal{A}_1, \eta)$ and $(X_2, \mathcal{A}_2, \nu)$. Let $g : X_1 \to \mathbb{R}$ measurable function. Recall that the change of variables formula is given by

$$\int_{X_1} g \, dF_{\#}\eta = \int_{X_1} (g \circ F) \, d\eta. \tag{20}$$

where $F_{\#}\eta$ denotes the pushforward of η through F.

Applying this to our setting, recall that a draw from the posterior $g \sim p(g \mid x, \mathcal{D})$ induces a random index $j(x)$ defined by the relation $g(x) = e_{j(x)}$. Formally, the evaluation map

$$s_x : \mathcal{G} \to \{1, \dots, m\}, \qquad s_x(g) = j(x),$$

pushes the posterior measure $p(g \mid x, \mathcal{D})$ forward onto a distribution over indices. Using this push-forward, we can rewrite equation 6 as

$$\int_{\mathcal{G}} p(y \mid x, g) \, d\,\mathbb{P}(g \mid x, \mathcal{D}) = \int_{\mathcal{G}} f_{s_x(g)}(y \mid x) \, d\,\mathbb{P}(g \mid x, \mathcal{D}) \tag{21}$$

$$= \int_{\{1,\dots,m\}} f_j(y \mid x) \, d\,(E_{x\#}\mathbb{P})(j \mid x, \mathcal{D}) \tag{22}$$

$$= \sum_{j=1}^{m} f_j(y \mid x) \, p(j \mid x, \mathcal{D}). \tag{23}$$

A.2 PROOF OF THEOREM 2.1

Theorem. *Denote $\mathcal{D}_i \coloneqq \{(x_t, y_t)\}_{t=1}^{i}$, and consider the posterior weights predictor $\hat{p}_\alpha^{(i)}$ assigning $\alpha_j^{(i)}(x) \coloneqq p(J(x) = j \mid \mathcal{D}_i, x)$ to the j-th predictor f_j. Assume that $\mathbb{E}[|\log f_j(Y \mid X)|] < \infty$ for all $f_j \in \mathcal{F}$. Then, for any measurable selector $j^* : \mathcal{X} \to \{1, \dots, m\}$ and any $n \geq 1$,*

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \hat{p}_\alpha^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1})\right] \geq \mathbb{E}\left[\log f_{j^*(x)}(y \mid x)\right] + \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \alpha_{j^*(x_i)}^{(i)}(x_i)\right], \tag{24}$$

where the expectations are taken w.r.t the population distribution $(x, y) \sim p(x, y)$.

Proof. Define the posterior-weights predictor

$$\hat{p}_\alpha^{(i)}(y \mid x, \mathcal{D}_{i-1}) = \sum_{j=1}^{m} \alpha_j^{(i)}(x) f_j(y \mid x) \tag{25}$$

For a fixed input x_i and a fixed predictor f_k we have that

$$\log \hat{p}_\alpha^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1}) = \log\left(\sum_{j=1}^{m} \alpha_j^{(i)}(x_i) f_j(y_i \mid x_i)\right) \tag{26}$$

$$\geq \log\left(\alpha_k^{(i)}(x_i) f_k(y_i \mid x_i)\right) \tag{27}$$

$$= \log f_k(y_i \mid x_i) + \log \alpha_k^{(i)}(x_i). \tag{28}$$

Taking $\mathbb{E}_{(x,y)\sim p(x,y)}[\cdot \mid x_i, \mathcal{D}_{i-1}]$, since $f_{j^*(x)}(y_i \mid x_i)$ is independent of $\mathcal{D}_{i-1}$,

$$\mathbb{E}\left[\log \hat{p}_\alpha^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1})\right] \geq \mathbb{E}\left[\log f_k(y_i \mid x_i)\right] + \mathbb{E}\left[\log \alpha_k^{(i)}(x_i) \mid \mathcal{D}_{i-1}\right]. \tag{29}$$

This holds for any $1 \leq k \leq m$, hence for $k = j^*(x_i)$,

$$\mathbb{E}\left[\log \hat{p}_\alpha^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1}) \mid \mathcal{D}_{i-1}\right] \geq \mathbb{E}\left[\log f_{j^*(x)}(y_i \mid x_i)\right] + \mathbb{E}\left[\log \alpha_{j^*(x_i)}^{(i)}(x_i) \mid \mathcal{D}_{i-1}\right]. \tag{30}$$

Taking $\mathbb{E}[\cdot \mid \mathcal{D}_{i-1}]$, by the law of total expectation,

$$\mathbb{E}\left[\log \hat{p}_{\alpha}^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1})\right] \geq \mathbb{E}\left[\log f_{j^*(x_i)}(y_i \mid x_i)\right] + \mathbb{E}\left[\log \alpha_{j^*(x_i)}^{(i)}(x_i)\right]. \tag{31}$$

Averaging over i, we get

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \hat{p}_{\alpha}^{(i)}(y_i \mid x_i, \mathcal{D}_{i-1})\right] \geq \mathbb{E}\left[\log f_{j^*(x_i)}(y_i \mid x_i)\right] + \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\log \alpha_{j^*(x_i)}^{(i)}(x_i)\right].$$

□

B Additional Experimental Results

In what follows we provide a deeper analysis of the performance of adaptive model averaging methods on the two case-studies.

B.1 Cancer Treatment Response

To illustrate how different methods allocate weights, we sampled 16 cases as follows: for each classifier f_j, we randomly selected four examples from those where IABMA assigned the highest weight to f_j. Figures 4–7 display the weights assigned by each averaging method for Ridge, XGB, HGB, and MLP. For each case, we also report the RMSE achieved by the individual classifiers. This analysis shows that in all cases, IABMA places the largest weight on the model with either the lowest error or a near-tied second. By contrast, competing methods tend to favor other predictors. In particular, MoE consistently prioritizes MLP or XGB, even in instances where these models are locally suboptimal.

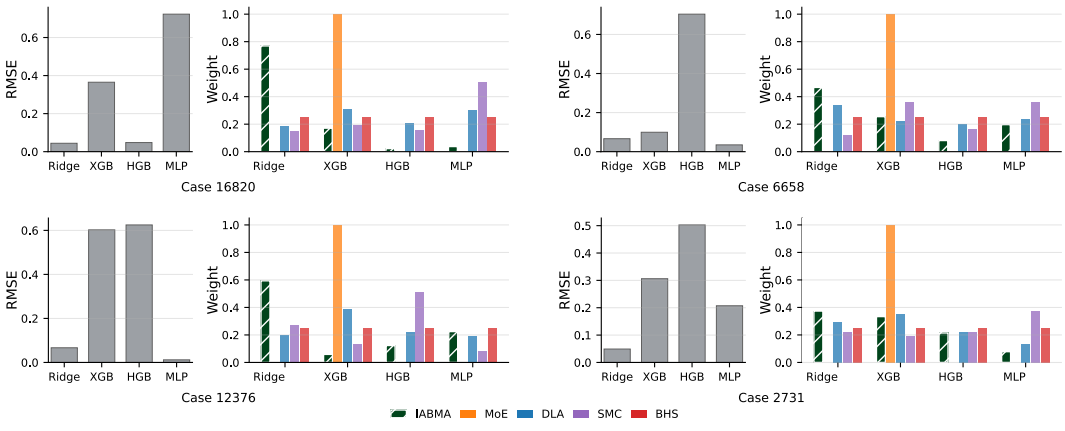

Figure 4: Cases where IABMA assigns the highest weight to Ridge.

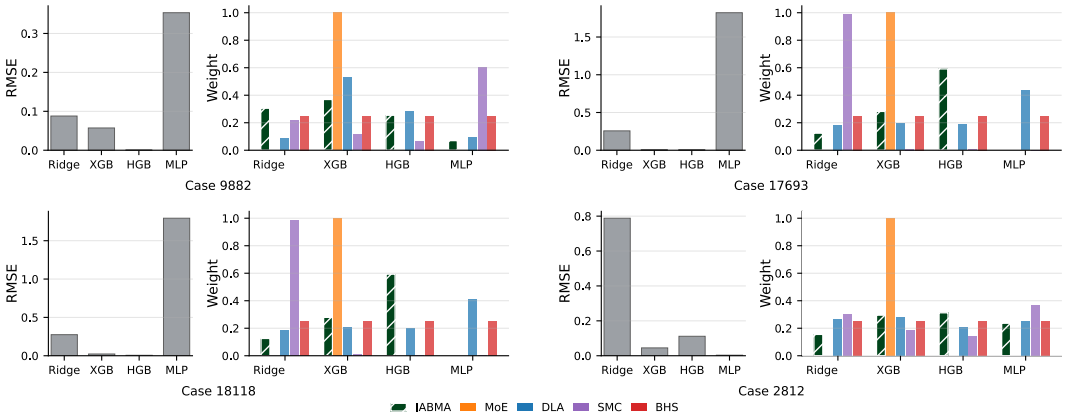

Figure 5: Cases where IABMA assigns the highest weight to XGB.

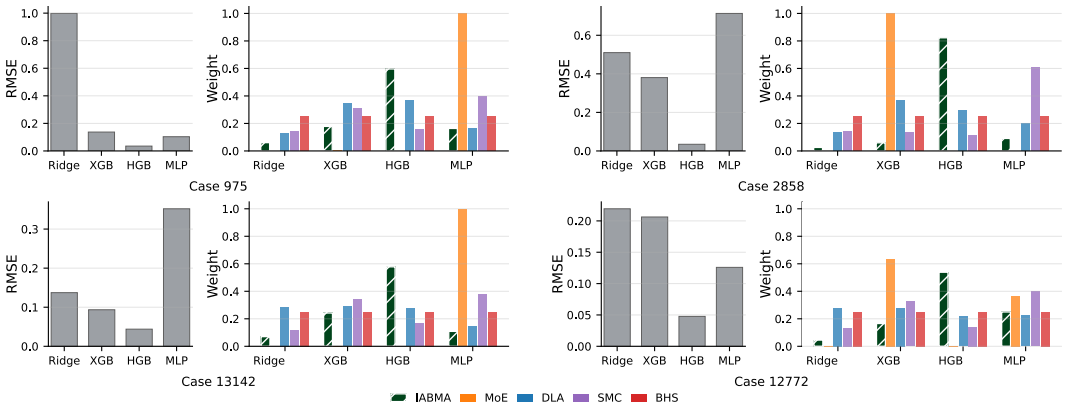

Figure 6: Cases where IABMA assigns the highest weight to HGB.

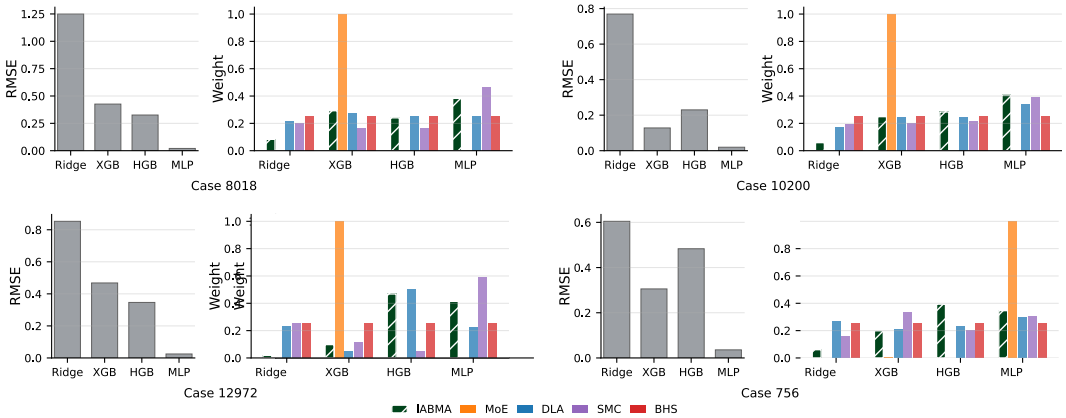

Figure 7: Cases where IABMA assigns the highest weight to MLP.

Table 1: UCI benchmarks: mean (sd) across runs.

Dataset	Metric	Best single	Uniform Avg.	Freq Avg.	BMA	MoE	DLA	SMC	BHS	IABMA
Bike-sharing	R2 (↑)	0.706 (0.022)	0.752 (0.010)	0.773 (0.012)	0.774 (0.014)	0.706 (0.022)	0.781 (0.013)	0.756 (0.010)	0.752 (0.010)	**0.794** (0.010)
Bike-sharing	RMSE (↓)	0.582 (0.033)	0.491 (0.021)	0.448 (0.020)	0.447 (0.021)	0.581 (0.033)	0.446 (0.020)	0.483 (0.020)	0.491 (0.021)	**0.433** (0.018)
Cal.-housing	R2 (↑)	0.772 (0.022)	0.840 (0.018)	0.840 (0.017)	0.812 (0.017)	0.778 (0.024)	0.840 (0.018)	0.817 (0.066)	0.840 (0.018)	**0.844** (0.014)
Cal.-housing	RMSE (↓)	0.036 (0.004)	0.025 (0.003)	0.025 (0.003)	0.029 (0.003)	0.035 (0.004)	0.025 (0.003)	0.029 (0.010)	0.025 (0.003)	**0.024** (0.003)
Credit-g	Accuracy (↑)	0.634 (0.036)	0.676 (0.029)	0.662 (0.038)	0.648 (0.036)	0.624 (0.036)	0.668 (0.039)	0.626 (0.046)	0.682 (0.039)	**0.684** (0.047)
Credit-g	ECE (↓)	0.260 (0.034)	0.172 (0.022)	**0.169** (0.020)	0.174 (0.023)	0.296 (0.035)	0.173 (0.025)	0.222 (0.038)	0.176 (0.025)	0.175 (0.020)
Spambase	Accuracy (↑)	0.699 (0.110)	0.702 (0.094)	0.738 (0.044)	0.760 (0.024)	0.760 (0.035)	0.729 (0.052)	0.757 (0.032)	0.646 (0.132)	**0.764** (0.032)
Spambase	ECE (↓)	0.114 (0.022)	0.163 (0.049)	0.148 (0.042)	0.169 (0.018)	**0.095** (0.034)	0.171 (0.051)	0.222 (0.023)	0.180 (0.061)	0.146 (0.025)

B.2 Credit card fraud

Credit card fraud prediction is a highly sensitive area, with risks of false alarms and misreporting, calibration is crucial not only overall but also within each bin. To this end, we analyzed the confidence measure $|p - 0.5|$ where p is the estimated probability, which captures certainty for both positive and negative events, and compared the bin-wise errors across averaging methods. Figure B.2 shows that in all high-confidence bins (confidence > 0.25), IABMA attains the lowest error, showing that most miscalculated predictions occur in low confidence instances.

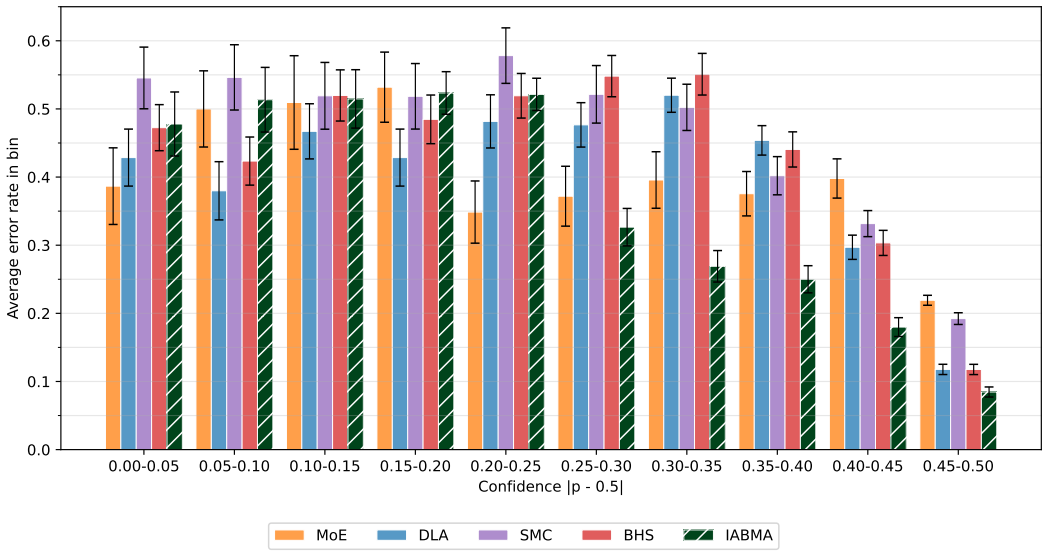

Figure 8: Calibration across confidence bins in credit-card fraud prediction

B.3 UCI benchmark datasets

Results are reported in Table 1.

C EXPERIMENTAL DETAILS

C.1 SIMULATION

C.1.1 DATA AND PROCESSING

We generated a two-dimensional binary dataset with two subpopulations governed by different decision rules. For the *linear* subpopulation, we drew $n_{\text{lin}} = n_{\text{train}}/2$ training points from a Gaussian cloud centered at $(-t, 0)$ (with $t = 1$),

$$X^{(\text{lin,train})} \sim \mathcal{N}\big((-t, 0),\, 0.1\, I_2\big),$$

and assigned labels by a linear rule $y = \mathbb{1}\{x_1 + x_2 > -t\}$. For the *circular* subpopulation, we drew $n_{\text{circ}} = n_{\text{train}} - n_{\text{lin}}$ points on a ring around $(t, 0)$ by sampling $\theta \sim \text{Unif}(0, 2\pi)$ and $r = \sqrt{U}$ with $U \sim \text{Unif}(0, 2)$, and set

$$X^{(\text{circ})} = (t, 0) + \big(r\cos\theta,\, r\sin\theta\big), \qquad y = \mathbb{1}\{r < 1\}.$$

We used $n_{\text{train}} = 1,000$ and $n_{\text{test}} = 500$; the train/test splits were generated independently.

Only the two coordinates (x_1, x_2) were provided as features. A region indicator $z \in \{0\ (\text{linear}),\ 1\ (\text{circular})\}$ was recorded for analysis but was not used during training.

C.1.2 CANDIDATE PREDICTORS

All averaging methods were evaluated on the same 3 base classifiers:

1. *Polynomial logistic regression (degrees 2 and 3).* We fit logistic regression with polynomial features of degree $d \in \{2, 3\}$ (no bias term in the expansion).
2. *Linear Discriminant Analysis (LDA).* A linear generative classifier fit on the raw coordinates, providing a single linear boundary.
3. *Soft-circle classifiers (two instances).* Each instance modeled the positive-class probability as a logistic function of radial distance to a fixed center,

   $$p_{\text{circle}}(y{=}1 \mid x) \;=\; \sigma\big(\gamma\,(R - \|x - c\|)\big), \quad c = (0.8\,t, 0),\ R = 1.0,\ \gamma = 5.0,$$

   yielding smooth circular decision regions around $(t, 0)$.
   We instantiated two copies of each model to allow the averaging procedure to allocate weight across similar experts.

C.2 PRISM CANCER EXPERIMENT

C.2.1 DATA AND PROCESSING

We used the publicly available PRISM cancer drug response dataset. The primary data[3] was combined with an RNA-seq expression matrix[4], cell-line metadata[5], and tissue labels[6]. All files are available from https://depmap.org/portal/data_page/.

The PRISM file reports drug–cell line responses with identifiers of the form ACH-#. We normalized all identifiers to the canonical zero-padded format (ACH-XXXXXX). Non-Continuous entries and all observations lacking a primary cancer site were excluded. Responses correspond to log-fold changes (LFC), clipped to the range $[-6, 6]$, and the prediction target was defined as $y = -v$, where v is the clipped LFC.

We focused on the 40 drugs with the greatest site-level heterogeneity. Specifically, we computed the between-site variance of y and retained compounds observed in at least 3 distinct sites, with at least 5 samples per site and at least 40 samples overall. A minimum per-site coverage threshold of 20

[3] Repurposing_Public_23Q2_Extended_Primary_Data_Matrix.csv
[4] OmicsExpressionProteinCodingGenesTPMLogp1.csv
[5] Cell_lines_annotations_20181226.txt
[6] Model.csv

samples was enforced. To avoid domination by a few large tissues, we capped each site at $1.1 \times s$, where s is its sample size. This yielded approximately $18{,}460$ drug–cell line pairs (slight variation across random splits), of which 80% were used for training and 20% for testing.

Gene expression features were restricted to the 100 highest-variance genes. Each gene was standardized to mean 0 and variance 1 based on training statistics. The final feature matrix consisted of standardized gene expression values and a categorical compound indicator.

The full processing code was submitted with this paper and will be released publicly upon acceptance.

C.2.2 CANDIDATE PREDICTORS

All averaging methods were evaluated on averaging the same four regression models with reprocessing pipelines tailored per model:

1. *Ridge regression (ℓ_2 regularized linear model).* Gene features were imputed (median), standardized to zero mean and unit variance, and combined with a dense one-hot encoding of the compound identity.
2. *Histogram-based Gradient Boosting regressor (HGB).* Tree-based model trained on raw gene values (median imputation only) together with a sparse one-hot encoding of the compound identity.
3. *XGBoost regressor (XGB).* Gradient-boosted decision trees with squared-error objective, trained using the same pre-processing as HGB. We used 400 estimators, learning rate 0.05, maximum depth 8, subsample ratio 0.9, and column subsample ratio 0.8, with ℓ_1 and ℓ_2 regularization.
4. *Multi-layer perceptron (MLP).* A feed-forward neural network with hidden layers of size (128, 64), ReLU activations, learning rate 10^{-3}, batch size 64, and early stopping based on a 10% validation split. Inputs were preprocessed as for Ridge (dense, imputed, standardized gene features and dense one-hot drug encoding).

C.3 IEEE-CIS FRAUD EXPERIMENT

C.3.1 DATA AND PROCESSING

We used the IEEE-CIS credit-card fraud dataset, available at https://www.kaggle.com/c/ieee-fraud-detection/data.

We removed rows with missing target (`isFraud`) and features with more than 50% missing values. To limit explosion in feature dimension, infrequent categories were grouped into a shared rare category.

In each repetition 80% of the data was used for training and 20% for testing. The training data was then reduced to obtained class balance, while in test data class imbalance was maintained. To reduced covariate shift in the train-test split we stratified jointly on (`ProductCD`, `card4`) crossed with per-row missingness bins and `TransactionAmt` quantile bins, with a fallback "RARE" bucket for very small strata. This procedure yielded a stable empirical mix of products, card networks, and spending levels. Specifically, to control the empirical mix of products, card networks, and spending levels we stratified jointly on (`ProductCD`, `card4`) crossed with per-row missingness bins and `TransactionAmt` quantile bins.

Continuous features were median-imputed and where appropriate, standardized to zero mean and unit variance. Categorical features were imputed to the most frequent level and one-hot encoded, with infrequent categories pooled into a rare-level. Class imbalance was addressed within each classifier as noted below.

C.3.2 CANDIDATE PREDICTORS

All averaging methods were evaluated over the same following base classifiers.

1. *Logistic Regression (ℓ_1-penalized).* We fit a penalized logistic model to the processed feature set, using an ℓ_1 penalty with strength to encourage sparsity and robustness to correlated predictors. We used a saga solver, ℓ_1 penalty with regularization strength of 0.05, maximal number of iterations as 4000, and tolerance of 10^{-3}.
2. *XGBoost (XGB).* We trained a gradient-boosted ensemble of shallow decision trees using histogram-based splits and early stopping. Depth, learning rate, and number of estimators were selected via a held-out validation set. Hyper parameters were set as maximal bin of 256, 300 estimators, maximal depth of 5, learning rate 0.1, row subsampling of 0.3, feature subsampling of 0.7, and ℓ_2 penalty with strength 1.0.
3. *Histogram-based Gradient Boosting (HGB).* We train boosted trees with a histogram grow policy, subsampling of observations and features, and ℓ_2 regularization. Class imbalance was addressed via the standard positive-class weight $\frac{n_{\text{neg}}}{n_{\text{pos}}}$, estimated from the training examples. Hyperparameters (learning rate, depth, estimators, subsampling ratios) were fixed based on validation performance and kept constant across comparisons. Hyperparameters were set to maximal depth of 4, learning rate 0.07, and ℓ_2 regularization with strength 0.5, and at most 350 iterations.
4. *Multi-layer perceptron (MLP).* We used a feed-forward network with two hidden layers of sizes 384, 192 and ReLU activations, trained with weight decay and early stopping on a validation split. Weight decay was set to $\alpha = 3 \cdot 10^{-3}$, batch size 512, adaptive learning rate with initial value of 10^{-3}, early stopping with validation fraction 0.12 and no change for 12 iterations, maximal number of iterations as 300, and tolerance 10^{-4}.

C.4 UCI EXPERIMENTS

C.4.1 DATA AND PROCESSING

We evaluated IABMA on standard UCI tasks retrieved from OpenML. We chose datasets with relatively large number of observations and features. For *classification*, we used `spambase` (target: `class`) and `credit-g` (target: `class`). For *regression*, we used `bike-sharing` (target: `cnt`) and `california-housing` (target: `MedHouseVal`).

We replaces common "unknown" tokens (e.g., `?`, `NA`, `NaN`, `unknown`) with missing values, stripping whitespace on string columns in each dataset, and dropped features whose missing rate exceeded 40%.

We used an $80\%/20\%$ train-test split in each repetition. For classification, we performed stratified sampling on the label to preserve class proportions in the test set, and then balanced only the training split by downsampling the majority class to the minority size. For regression, we created an approximately balanced split by binning the continuous target into 12 quantile bins and stratifying on those bins. All pre-processing statistics (imputation, scaling, and one- hot vocabularies) were computed on the training partition and applied unchanged to the test data.

To encourage diversity among base models, we formed several heterogeneous, partially overlapping *feature bundles* and trained each model on a bundle tailored to its strengths. Bundles were constructed from the training data as follows:

- **B1:** up to 3 Continuous features with highest absolute Pearson correlation with the target (continuous median-imputed for this computation).
- **B2:** up to 3 highest-variance Continuous features.
- **B3:** up to 3 categorical features with highest cardinality.
- **B4:** up to 5 remaining low-cardinality categorical variables.
- **B5:** all categorical features.
- **B6:** all Continuous features.
- **B7:** the union of **B1** and **B3**.

Continuous features in non-tree models were median-imputed and standardized. Categorical features were imputed to the most frequent level and one-hot encoded with a minimum frequency threshold of 10 to pool rare levels; unknown categories at test time were ignored.

C.4.2 CANDIDATE PREDICTORS

All averaging methods were evaluated on the same base models

D IMPLEMENTATION DETAILS

In all our experiments the posterior network for IABMA and the gating network for MoE were implemented as feed-forward neural networks with hidden layers of size (64, 32, 16) and ReLU activations. We used Adam optimizer for MoE and IABMA across all experiments.

Hyperparameters for our method and all baselines were tuned via binary search to maximize average performance (accuracy for classification, RMSE for regression) on a held-out repetition excluded from the analysis. The selected values and running times by experiment and method are reported below.

D.1 HYPERPARAMETERS OF ENSEMBLE METHODS

Table 2: Hyperparameters of Mixture-of-Experts

Hyperparameter	Synthetic	PRISM (Cancer)	Fraud (IEEE-CIS)	UCI
Learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Batch size	64	128	64	64
Epochs	10	20	10	10

Table 3: Hyperparameters of Dynamic Local Accuracy (DLA).

Hyperparameter	Synthetic	PRISM (Cancer)	Fraud (IEEE-CIS)	UCI
Neighborhood size k	50	50	50	50
Temperature T	0.8	1.0	1.0	1.0
Smoothing α	1.0	1.0	1.0	1.0

Table 4: Synthetic Mixture of Experts (SMC).

Hyperparameter	Synthetic	PRISM (Cancer)	Fraud (IEEE-CIS)	UCI
Confident-cover threshold	0.6	0.6	0.6	0.6
Cover quantile (reg.)	–	0.30	–	0.30
Min coverage per model	20	20	20	20
Cov. reg. (reg. mix)	0.9 (Gaussian scores)	0.9	0.9	0.7

Table 5: Bayesian Hierarchical Stacking (BHS).

Hyperparameter	Synthetic	PRISM (Cancer)	Fraud (IEEE-CIS)	UCI
Temperature T	1.0	1.0	1.0	1.0
Prior weight	1.0	1.0	1.0	1.0
Slab scale s_0	5.0	5.0	5.0	5.0
Learning rate	5×10^{-3}	5×10^{-3}	5×10^{-3}	10^{-3}
Batch size	64	128	64	64
Epochs	10	20	10	10

D.2 RUNTIMES

Table 6: Input-Adaptive Bayesian Model Averaging (IABMA)

Hyperparameter	Synthetic	PRISM (Cancer)	Fraud (IEEE-CIS)
Learning rate	10^{-3}	10^{-3}	10^{-3}
Batch size	64	128	64
Epochs	10	30	10
KL weight λ_{KL}	0.05	0.2	0.2

Table 7: IABMA (PosteriorNet) hyperparameters per UCI dataset.

Hyperparameter	`Spambase` (clf)	`Credit-g` (clf)	`Bike-sharing` (reg)	`Cal housing` (reg)
Learning rate	5×10^{-3}	5×10^{-3}	1×10^{-3}	1×10^{-3}
Batch size	64	64	64	64
Epochs	10	10	10	10
KL weight λ_{KL}	0.1	0.1	0.8	3.0

Table 8: Method runtimes (seconds): mean (sd) across 10 repetitions.

Experiment	**MoE**	**DLA**	**SMC**	**BHS**	**IABMA**
Cancer	147.359 (5.282)	22.269 (0.454)	0.072 (0.114)	28.572 (1.167)	252.985 (5.571)
Fraud	439.502 (129.487)	8.246 (1.719)	688.473 (155.629)	16.622 (3.139)	461.312 (121.168)
Simulation	5.664 (0.104)	0.218 (0.008)	0.079 (0.004)	1.040 (0.079)	5.889 (0.038)
Bike-Sharing	25.080 (3.780)	0.868 (0.179)	0.007 (0.001)	21.364 (1.063)	29.663 (4.094)
Cal. housing	8.510 (0.987)	0.350 (0.041)	0.006 (0.001)	7.281 (0.324)	9.815 (1.029)
Credit-g	3.178 (0.049)	0.439 (0.017)	0.174 (0.007)	1.184 (0.148)	3.345 (0.048)
Spambase	16.420 (0.287)	0.642 (0.025)	0.822 (0.159)	1.781 (0.147)	18.651 (5.122)