Expert Systems With Applications 310 (2026) 131255

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Variational oblique predictive clustering trees

Viktor Andonovikj [a,b,*], Sašo Džeroski [a], Biljana Mileva Boshkoska [a,c], Pavle Boškoski [a,c]

[a] Jožef Stefan Institute, Jamova cesta 39, 1000, Ljubljana, Slovenia
[b] Jožef Stefan International Postgraduate School, Jamova cesta 39, 1000, Ljubljana, Slovenia
[c] Faculty of Information Studies in Novo mesto, Ljubljanska cesta 31b, 8000, Novo mesto, Slovenia

ARTICLE INFO

Keywords:
Variational inference
Predictive clustering
Interpretable models
Structured output prediction
Uncertainty quantification.

ABSTRACT

Oblique predictive clustering trees (SPYCTs) are semi-supervised multi-target prediction models mainly used for structured output prediction (SOP) problems. They are computationally efficient and when combined in ensembles they achieve state-of-the-art results. However, one major issue is that it is challenging to interpret an ensemble of SPYCTs without the use of a model-agnostic method. We propose variational oblique predictive clustering trees, which address this challenge. The parameters of each split node are treated as random variables, described with a probability distribution, and they are learned through the Variational Bayes method. We evaluate the model on several benchmark datasets of different sizes. The experimental analyses show that a single variational oblique predictive clustering tree (VSPYCT) achieves competitive, and sometimes better predictive performance than the ensemble of standard SPYCTs. We also present a method for extracting feature importance scores from the model. Finally, we present a method to visually interpret the model's decision making process through analysis of the relative feature importance in each split node.

1. Introduction

In the evolving field of machine learning, the fusion of semi-supervised learning and decision tree models has led to the development of models like predictive clustering trees (PCTs) Kocev et al. (2013) and their advanced variant, SPYCTs Stepišnik and Kocev (2021). These models leverage labeled and unlabelled data to enhance predictive accuracy and efficiency. SPYCTs, in particular, have pioneered the use of oblique splits for SOP, allowing for more intricate decision boundaries beyond the capabilities of traditional axis-parallel splits. In this work we use the term SOP to denote predictive tasks where each example's label is itself a vector, set or hierarchy (e.g., multi-target regression, hierarchical multi-label, multi-output classification). Despite their advantages, a significant challenge with SPYCTs is their reliance on ensemble methods for optimal predictive performance. Ensemble methods, such as random forests and gradient-boosted trees, achieve high predictive performance by reducing variance and/or bias through the aggregation of diverse models Breiman (2001), Friedman (2001), Kocev et al. (2020). These methods ensure robustness by capturing slightly different patterns or residuals from the data. While effective in improving accuracy, ensembles tend to obscure the model's interpretability-a fundamental attribute of decision trees-and do not inherently provide a means to quantify prediction uncertainty, which is crucial for informed decision-making across various applications.

To address these limitations, we introduce the VSPYCT model. VSPYCT integrates the variational Bayes method Blei et al. (2017) to facilitate complex decision-making within a singular model framework, thus maintaining the interpretability inherent to decision trees without necessitating ensembles. Unlike ensembles, which explicitly combine multiple SPYCT trees, a single VSPYCT attempts to incorporate uncertainty by sampling from the posterior distribution of tree parameters. Furthermore, by embedding model-specific uncertainty quantification directly into the decision tree structure through Bayesian inference, VSPYCT enhances the depth of insight into the model's decision process and confidence levels. Combining the robustness of SPYCT ensembles with the clarity and interpretability of a single tree model, alongside introducing uncertainty quantification, our research provides a comprehensive tool as an alternative with a goal to overcome the current limitations of predictive clustering methodologies.

Our proposed VSPYCT framework diverges from traditional ensemble approaches by directly incorporating Bayesian principles into oblique decision trees' architecture. This is illustrated in Fig. 1. The model avoids conventional ensemble strategies, which aggregate

* Corresponding author.
E-mail addresses: viktor.andonovikj@ijs.si (V. Andonovikj), saso.dzeroski@ijs.si (S. Džeroski), biljana.mileva@ijs.si (B. Mileva Boshkoska), pavle.boskoski@ijs.si (P. Boškoski).

https://doi.org/10.1016/j.eswa.2026.131255
Received 29 April 2025; Received in revised form 22 June 2025; Accepted 15 January 2026
Available online 22 January 2026

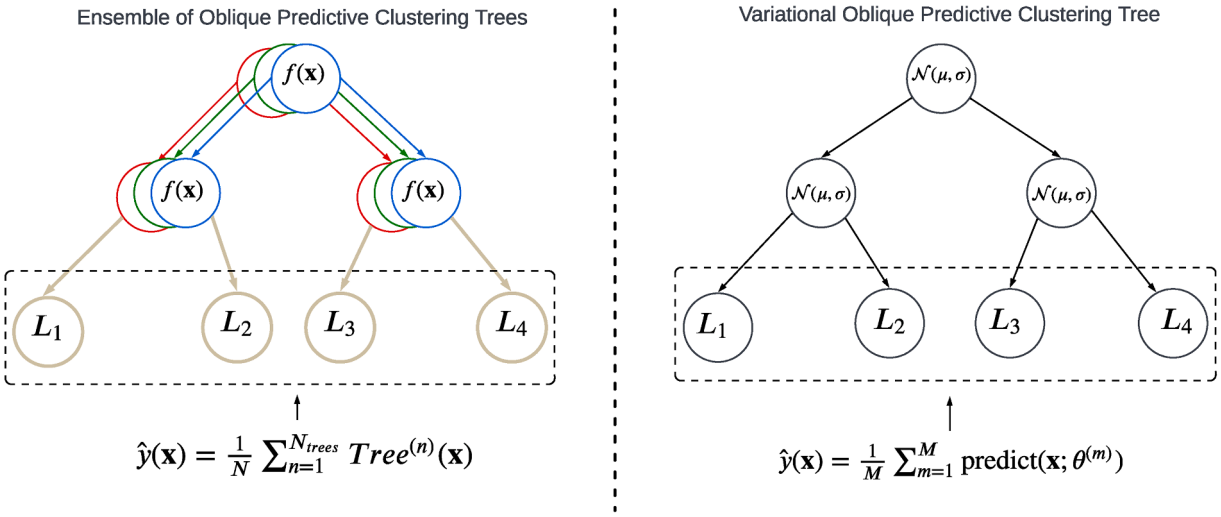

Fig. 1. Visual contrast between a bagged ensemble of oblique predictive clustering trees (left) and the proposed single Variational SPYCT (right). The shading of hyper-planes illustrates parameter uncertainty that is captured by the variational posterior, rather than by averaging many independent trees.

multiple trees' outputs, for a probabilistic treatment of model parameters, offering refined mechanisms for uncertainty quantification and predictive performance.

Despite the progress in predictive clustering methods, existing approaches either sacrifice interpretability by relying on ensembles or fail to directly incorporate uncertainty quantification. Recent works, such as SPYCTs Stepišnik and Kocev (2021) and option predictive clustering trees (OPCTs) Stepisnik et al. (2020), have addressed some limitations but lack the integrated capability of uncertainty quantification, critical for real-world decision-making contexts. Hence, there is a clear research gap in developing methods that simultaneously ensure interpretability, accuracy, and robust uncertainty estimation.

In contrast to prior work, VSPYCTs combine oblique splits from predictive clustering trees with a Bayesian inference framework using variational Bayes. Unlike traditional ensemble methods or single deterministic trees, VSPYCTs explicitly model uncertainty in split parameters, providing inherent uncertainty quantification without sacrificing interpretability. This integration represents a significant novelty by directly embedding probabilistic modeling into oblique splits, ensuring a single interpretable model can achieve performance comparable to or surpassing state-of-the-art ensembles across various prediction tasks.

The complete implementation is available at https://github.com/viksa98/variational_spyct.

2. Related work

Advanced modeling methods, including fractional calculus and artificial neural networks, have been applied extensively across diverse domains for solving complex modeling and prediction problems Abbas et al. (2025a,b), Abbas and Nazar (2024), Andonovikj et al. (2022), Ramzan et al. (2025).

PCTs have been an essential development in extending decision tree capabilities to various predictive modelling tasks, including structured output prediction. PCTs are versatile and can be combined into ensembles to achieve state-of-the-art performance Kocev et al. (2013). However, as the dimensionality of the output space increases, the learning time of PCTs scales poorly, posing challenges in tasks like hierarchical multi-label classification where outputs can consist of hundreds of potential labels.

PCTs are celebrated for their interpretability and efficiency, yet they often fall short in predictive performance due to their myopic, greedy selection of splits. The introduction of OPCTs Stepisnik et al. (2020) aimed to address this limitation by incorporating multiple alternative splits within option nodes. This approach mitigates the inherent myopia and achieves competitive performance with ensemble methods like bagging and random forests while still maintaining a degree of interpretability. OPCTs, despite mitigating myopia and enhancing performance, still face challenges in maintaining interpretability when extended to complex tasks.

On the other hand, SPYCTs Stepišnik and Kocev (2021) utilise oblique splits that incorporate linear combinations of features, allowing splits to correspond to arbitrary hyperplanes in the input space. This makes SPYCTs highly efficient for high-dimensional data and capable of leveraging data sparsity. Experimental evaluations on numerous benchmark datasets have shown that SPYCTs achieve performance on par with state-of-the-art methods while being significantly faster than standard PCTs Andonovikj et al. (2024).

Despite these advancements, ensemble models such as random forests and gradient-boosting machines remain popular due to their superior predictive accuracy. However, their complexity often obscures interpretability, making it challenging for users to understand the model's decision-making process. Efforts to bridge this gap include the iForest visual analytics system Zhao et al. (2019), which summarises decision paths and tree structures within random forests, thereby making the ensemble's decisions more transparent. Additionally, the Tree Space Prototypes approach Tan et al. (2020) uses representative points (prototypes) for each class, offering a more intuitive understanding of ensemble classifiers than traditional feature-based explanations. Visual analytics systems like iForest and prototype-based approaches offer some clarity but do not fully resolve the complexity issue.

In critical applications, such as medical diagnosis, the need for transparent and trustworthy predictions is paramount. Methods for explaining classifier predictions and estimating the reliability of regression predictions, as demonstrated in breast cancer recurrence prediction, provide users with additional insights and build trust by clarifying the decision-making process. Similarly, the MAPLE Plumb et al. (2018) model combines local linear modelling with a dual interpretation of random forests, offering faithful self-explanations and maintaining high predictive accuracy. MAPLE addresses the accuracy-interpretability trade-off and provides both example-based and local explanations, making it a comprehensive tool for understanding model behaviour.

Table 1
Comparative summary of predictive clustering methods.

Model	Interpretability	Handles High-dimensional Data	Uncertainty Quantification
PCT Kocev et al. (2013)	High	Limited	No
OPCT Stepisnik et al. (2020)	Moderate to High	Moderate	No
SPYCT Stepišnik and Kocev (2021)	Moderate	High	No
Ensemble methods	Low	High	Partial (via variance)
VSPYCT (ours)	**Moderate to High**	**High**	**Yes (inherent, Bayesian)**

Methods like MAPLE and other interpretability frameworks strike a balance between accuracy and transparency, but they often fall short in handling high-dimensional, sparse data efficiently.

To provide a clearer overview of the related predictive clustering approaches discussed, we summarise and contrast their interpretability, capability to handle high-dimensional data, and uncertainty quantification properties in Table 1.

While significant progress has been made in improving decision tree models and their ensembles, several weaknesses remain. Many of these approaches lack inherent mechanisms for uncertainty quantification, which is crucial for applications requiring high reliability and decision certainty.

Our proposed VSPYCT model addresses these critical challenges by integrating Bayesian principles directly into the predictive clustering tree framework. Such example where Bayesian principles, specifically the variational Bayes method, have been directly integrated in a predictive model can be found in Boškoski et al. (2021), Lopez et al. (2023) This approach maintains the clarity and simplicity of a single tree model while achieving on-par performance with the state-of-the-art methods. By introducing uncertainty quantification directly into the decision-making process, VSPYCT enhances the reliability and applicability of the model across various domains where decision certainty is crucial.

3. The variational bayes method

In conducting a stochastic analysis, the process of inferring model parameters hinges on applying Bayes' theorem. For observations x produced by a system defined by parameters θ, Bayes' theorem is given as:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \tag{1}$$

The model-prescribed likelihood $p(x|\theta)$ and the selected prior $p(\theta)$ are generally well-defined. The primary computational challenge arises in determining the posterior distribution $p(\theta|x)$ due to the denominator $p(x)$, which is derived as:

$$p(x) = \int_{\theta} p(x|\theta)p(\theta)\, dx, \tag{2}$$

This integral rarely yields a closed-form solution. For cases with multiple dimensions, the computational burden often renders Monte Carlo methods infeasible. The VB approach offers an approximation to this issue by closely mimicking the true posterior $p(\theta|x)$ with an approximate distribution $q_{\omega^*}(\theta)$. This variational distribution is part of a family of distributions $q_{\omega}(\theta) \in \mathcal{Q}$, parameterised by $\omega \in \Omega$, where Ω represents the potential latent parameter values. Typically, the mean-field variational family is utilised, assuming independence among the parameters ω Blei et al. (2017). To find the optimal ω^*, one minimises the Kullback-Leibler (KL) divergence $KL(q_{\omega}(\theta) \parallel p(\theta|x))$:

$$\omega^* = \arg\min_{\omega\in\Omega} KL(q_{\omega}(\theta) \parallel p(\theta|x)) \tag{3}$$

Given the unknown nature of the true posterior, a rearrangement is needed to compute the KL divergence, simplifying to (Murphy (2023) Chapter 10):

$$KL(q_{\omega}(\theta) \parallel p(\theta|x)) = \underbrace{-\mathbb{E}_q\left[\log p(x,\theta) - \log q_{\omega}(\theta)\right]}_{\text{ELBO}} + \log p(x) \tag{4}$$

Maximising the Evidence Lower Bound (ELBO) serves to inversely minimise the KL divergence between the approximate posterior $q_{\omega}(\theta)$ and the true posterior $p_{\omega}(\theta)$. In the decomposition of the KL divergence, the term $\log p(x)$, known as the marginal likelihood or evidence, remains constant with respect to the variational parameters θ. Since this term does not involve θ, it does not influence the optimisation of the variational distribution and is thus excluded from the ELBO maximisation process. By maximising the ELBO, we aim to tighten the bound provided by the KL divergence, effectively making the variational posterior a better approximation of the true posterior. Typically, the ELBO maximisation criterion is non-convex, with no assurance of global extremum convergence by optimisation algorithms (Murphy (2023) Chapter 10). Various algorithms have been proposed for this challenge, including variational EM Bernardo et al. (2003), stochastic variational inference Hoffman et al. (2013), Sashank et al. (2018), amortised variational inference Gershman and Goodman (2014), Le et al. (2017), and semi-amortised inference Kim et al. (2018).

The adoption of an approximate variational distribution introduces bias contingent on the chosen variational family $\mathcal{Q}$, necessitating a decision grounded in empirical evidence or expert knowledge. This bias arises because the family $\mathcal{Q}$ may not be capable of capturing the true complexity of the posterior distribution $p_{\omega}(\theta)$. As a result, the accuracy of the approximation depends significantly on how well $\mathcal{Q}$ aligns with the true underlying distribution. The choice of $\mathcal{Q}$ thus not only affects the efficiency of the inference but also the quality and reliability of the model. Despite this bias, VB's computational efficiency significantly surpasses traditional methods like Markov Chain Monte Carlo (MCMC). In our case, the ADAM optimiser Kingma and Ba (2014) is used to facilitate the ELBO optimisation.

4. Methodology

As the foundation of the VSPYCT model, the underlying architecture closely follows that of the SPYCT model, particularly in its tree construction process. Similar to SPYCT, VSPYCT constructs a decision tree with an oblique structure, where each split is defined by a linear combination of input features, rather than relying on single-feature thresholds as in axis-aligned trees. The key distinction between VSPYCT and SPYCT lies in the optimisation step used to determine the parameters of these oblique splits. While SPYCT utilises a deterministic approach to optimise split parameters, VSPYCT introduces a probabilistic framework through variational Bayes (VB) optimisation.

The construction of the VSPYCT begins with a root node and proceeds iteratively, expanding the tree by adding internal nodes or leaf nodes in a manner identical to SPYCT. Each node in the tree represents a binary decision point, where instances are split based on the oblique hyperplane defined by the linear model:

$$f(\mathbf{x}) = \sigma\left(\mathbf{w}^{\top}\mathbf{x} + b\right) \tag{5}$$

where $\mathbf{x} \in \mathbb{R}^N$ is the feature vector, $\mathbf{w} \in \mathbb{R}^N$ represents the weight vector, $b \in \mathbb{R}$ is the bias term, and $\sigma(\cdot)$ denotes the sigmoid function, which

ensures the output is a probability, determining whether an instance is sent to the right or left child node. While the tree structure and the decision-making process follow the same principles as in SPYCT, the optimisation of $\mathbf{w}$ and b in VSPYCT differs fundamentally. In VSPYCT, these parameters are not fixed but are instead inferred through variational Bayes optimisation. The introduction of this probabilistic element distinguishes VSPYCT from SPYCT, enabling it to maintain the interpretability of a single decision tree while incorporating the robustness typically associated with ensemble methods.

4.1. Learning splits using variational bayes

The goal is to update these prior distributions to approximate the posterior distributions given the observed data, thereby capturing the uncertainty in the model parameters. The weights $\mathbf{w}$ and bias b in the linear model, as in (5), are initially modelled with Gaussian prior distributions:

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad b \sim \mathcal{N}(0, 1) \tag{6}$$

The VB method aims to approximate the true posterior distributions of these parameters with a variational distribution $q(\mathbf{w}, b|D)$. We assume a simpler distributional form - a diagonal Gaussian:

$$q(\mathbf{w}, b|D) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)\mathcal{N}(b|\mu_b, \sigma_b^2), \tag{7}$$

where $\boldsymbol{\mu}_w$, $\boldsymbol{\Sigma}_w$, μ_b, and σ_b^2 are the parameters of the variational distribution, learned through the optimisation process.

The optimisation of the variational parameters is performed by maximising the ELBO, which serves as a lower bound to the log marginal likelihood $\log p(D)$. The ELBO is given by:

$$\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathbb{E}_{q(\mathbf{w},b|D)}\left[\log p(D|\mathbf{w}, b)\right] - KL(q(\mathbf{w}, b|D) \parallel p(\mathbf{w}, b)) \tag{8}$$

The first term represents the expected log-likelihood of the data under the variational distribution, while the second term is the Kullback-Leibler (KL) divergence between the variational distribution and the prior. The ELBO is maximised using the ADAM optimiser, with the goal of refining the variational parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ to closely approximate the true posterior.

The quality of a split is assessed using an impurity function, which is defined as:

$$\text{Impurity} = \sum_{i=1}^{N} \left(\rho_i \text{Var}_\rho(\mathbf{Y}) + \lambda_i \text{Var}_\lambda(\mathbf{Y})\right), \tag{9}$$

where $\rho_i = \sigma(\mathbf{w}^\top \mathbf{x}_i + b)$ is the probability of an instance being routed to the right child node, and $\lambda_i = 1 - \rho_i$ is the probability of routing to the left child node. The variances $\text{Var}_\rho(\mathbf{Y})$ and $\text{Var}_\lambda(\mathbf{Y})$ represent the variance of the target variables in the right and left child nodes, respectively, weighted by their selection probabilities. The objective is to minimise the impurity by optimising the parameters $\mathbf{w}$ and b using the VB method, thereby ensuring that the split at each node effectively separates the data into homogeneous subsets. The overall procedure for learning a split in VSPYCT is detailed in Algorithm 1. The algorithm involves initialising the variational parameters, performing Monte Carlo sampling to estimate the ELBO, and iteratively updating the parameters until convergence.

We selected Gaussian priors for split parameters due to their computational tractability and effectiveness in Bayesian inference frameworks. The impurity function used for evaluating splits leverages the variance reduction weighted by selection probabilities, a common choice ensuring effective partitioning of data into homogeneous subsets. Feature importance is calculated using the normalised expected weight-to-variance ratio, reflecting both the strength and reliability of each feature's contribution.

While parameter uncertainty is effectively addressed, structural uncertainty is inherently more challenging to capture in a single tree. The variational Bayes approach approximates the posterior over parameters, but the inclusion of priors on tree structure (e.g., split selection) is an area for further exploration.

Algorithm 1 Variational Learning of Split Parameters in VSPYCT with Impurity Minimisation.

Input: $D = \{\mathbf{X}, \mathbf{Y}\}$ (dataset), $\theta = \{\mathbf{w}_0, b_0\}$ (initial parameters), E (epochs), λ (learning rate), β (batch size), σ (selection probability)
Output: $\Theta = \{\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w, \mu_b, \sigma_b^2\}$ (variational parameters)
procedure LEARNSPLIT($D, \theta, E, \lambda, \beta, \sigma$)
  Initialise $\Theta = \{\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w, \mu_b, \sigma_b^2\}$ ▷ Initialise variational parameters
  Initialise optimiser: $\alpha \leftarrow \lambda$
  for $e \in \{1, \ldots, E\}$ **do** ▷ Iterate over epochs
    for each mini-batch $\mathcal{B} \subseteq D$ of size β **do**
      Sample $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$, $b \sim \mathcal{N}(\mu_b, \sigma_b^2)$
      Compute impurity $\Omega(\mathbf{X}, \mathbf{Y}; \mathbf{w}, b)$ as:

$$\Omega(\mathbf{X}, \mathbf{Y}; \mathbf{w}, b) = \sum_{\text{right}} \rho(\mathbf{w}^\top \mathbf{x} + b) \cdot \text{Var}_\rho(\mathbf{Y}) + \sum_{\text{left}} (1 - \rho(\mathbf{w}^\top \mathbf{x} + b)) \cdot \text{Var}_\lambda(\mathbf{Y})$$

      Set the observed impurity value: $\Omega_{\text{obs}} = \frac{\Omega(\mathbf{X}, \mathbf{Y}; \mathbf{w}, b)}{2}$ ▷ Scaled to remain lower than current impurity and guide minimisation
      Condition on the observed impurity using:

$$\log p(\Omega_{\text{obs}}|\Omega(\mathbf{X}, \mathbf{Y}; \mathbf{w}, b))$$

      Compute ELBO $\mathcal{L}(\mathcal{B}; \Theta)$ as per (8)
      Compute gradient ∇_ΘELBO
      Update $\Theta \leftarrow \Theta + \alpha \nabla_\Theta$ELBO ▷ Gradient descent update
    end for
    if early stopping criterion satisfied **then**
      break
    end if
  end for
  return Θ
end procedure

4.2. Making a prediction

The procedure for making a prediction for a new instance using the VSPYCT model is detailed in Algorithm 2. The process involves traversing the tree from the root node to a leaf node, with decisions at each internal node being made based on the parameters sampled from the learned variational posterior distributions. The final prediction is determined by the prototype value stored in the reached leaf node.

Mathematically, at each internal node, the oblique split is determined by evaluating the linear combination of input features, with the parameters $\Theta = (\mathbf{w}, b)$ drawn from their respective variational posterior distributions. The decision to branch left or right is based on the probability computed using the sigmoid function applied to this linear combination. This method accounts for the uncertainty in the model parameters, providing a probabilistic framework for decision-making. The prediction $\hat{y}$ returned by the model is the prototype of the reached leaf node, where the prototype is defined as the mean of the target values associated with the training instances that were routed to that leaf during training.

The posterior sampling allows the model to integrate over the uncertainty of the parameters at each node, providing smoothed and robust decision boundaries. The model captures variability in the decision process, reducing the risk of overfitting to a single parameter set. While this approach effectively accounts for parameter uncertainty, the reliance on a single tree structure limits the model's ability to replicate the diversity inherent in ensemble methods. Ensembles, achieve variance reduction

by combining predictions from multiple distinct structures, a capability that a single VSPYCT cannot fully emulate.

$$\hat{y} = \frac{1}{|\mathbf{Y}|} \sum_{i=1}^{|\mathbf{Y}|} y_i \tag{10}$$

where y_i represents the target value of the i-th instance in $\mathbf{Y}$. This value $\hat{y}$ serves as the prediction for any new instance that reaches the particular leaf node.

Algorithm 2 Prediction Process in VSPYCT using Monte Carlo Sampling.

Input: Feature vector $\mathbf{x}$, decision tree T rooted at node Ω, number of samples M
Output: Prediction $\hat{y}$
function MC_PREDICT(Ω, $\mathbf{x}$, M)
  Initialise $\hat{y}_{\text{sum}} = 0$
  for $m = 1$ to M **do**
    $\hat{y}_{\text{sum}} \leftarrow \hat{y}_{\text{sum}} + \text{Predict}(\Omega, \mathbf{x}, m)$
  end for
  return $\hat{y} = \frac{\hat{y}_{\text{sum}}}{M}$
end function
function PREDICT(Ω, $\mathbf{x}$, m)
  if Ω is a leaf node **then**
    return $\hat{y}_\Omega = \frac{1}{|D_\Omega|} \sum_{i \in D_\Omega} y_i$
  else
    Sample $\Theta^{(m)} = (\mathbf{w}^{(m)}, b^{(m)})$ from $q(\mathbf{w}, b|D)$
    Compute $\rho(\Theta^{(m)}, \mathbf{x}) = \sigma(\mathbf{w}^{(m)\top}\mathbf{x} + b^{(m)})$
    if $\rho(\Theta^{(m)}, \mathbf{x}) \leq 0.5$ **then**
      return PREDICT(Ω_{left}, $\mathbf{x}$, m)
    else
      return PREDICT(Ω_{right}, $\mathbf{x}$, m)
    end if
  end if
end function
$\hat{y} \leftarrow$ MC_PREDICT(Ω, $\mathbf{x}$, M)
Return $\hat{y}$

4.3. Time complexity analysis

We analyse the time complexity of learning a split in VSPYCT and compare it with SPYCT. Our analysis focuses on the computational expense associated with determining a split, as this represents the principal distinction between the two approaches. In the predictive clustering framework, the set of clustering attributes K, which contributes to the variance calculation, encompasses both the features and the target variables. As a result, K equals $D + T$, with D representing the feature count and T denoting the target count. The time complexity for learning in SPYCTs can be expressed as $\mathcal{O}(N I_o(D + K))$, wherein N is the number of training instances, and I_o is the iteration count needed for optimisation Stepišnik and Kocev (2021). Notably, the computational demand for the SPYCTs's gradient-based version increases linearly with the attribute count.

The introduction of variational Bayes method in the architecture of the model adds a new layer of computational complexity. Specifically, the complexity of learning a split in VSPYCT is influenced by the following factors: the need to sample parameter sets from the approximative posterior distributions, the iterative optimisation required to refine these estimates towards minimising the ELBO, and the dimensionality imposed by the number of features. The combined computational complexity for learning a split in VSPYCT can thus be expressed as $\mathcal{O}(M N I_{vb}(D + K))$, where:

- M represents the number of Monte Carlo samples needed to approximate the posterior distributions adequately.
- N is the number of data points evaluated during each iteration.
- D indicates the number of features, each contributing to the parameter space that must be sampled and optimised.
- K indicates the number of clustering attributes, used to calculate the variance in each of the node.
- I_{vb} indicates the number of iterations required for the variational inference process to achieve convergence.

This complexity reflects the multiplicative impact of sampling, feature dimensionality, and iterative optimisation, highlighting how the variational approach scales with the size and feature richness of the dataset.

Comparing VSPYCTs to SPYCTs, the VB method increases the computational load due to the necessity of Monte Carlo sampling and iterative ELBO optimisation. However, this additional complexity is counterbalanced by the enhanced model interpretability, and uncertainty quantification offered by VSPYCTs.

4.4. Feature importance

The procedure for computing feature importance scores in VSPYCT is inspired by the methodology introduced in the original SPYCT model Stepišnik and Kocev (2021), while extending it to incorporate the model's probabilistic nature. Let $\mathbb{E}[\mathbf{w}_s]$ and $\text{Var}(\mathbf{w}_s)$ denote the element-wise mean and variance of the weights at node s. We define:

$$\mathbf{u}_s = \frac{\mathbb{E}[\mathbf{w}_s]}{\text{Var}(\mathbf{w}_s) + \epsilon}$$

where ϵ is a small constant added for numerical stability.

The feature importance vector $\text{Imp}(T)$ is then computed as:

$$\text{Imp}(T) = \sum_{s \in T} \left(\frac{s_n}{N}\right) \cdot \frac{|\mathbf{u}_s|}{\|\mathbf{u}_s\|_2} \tag{11}$$

Here, $\frac{s_n}{N}$ weights the contribution of node s by the fraction of training samples passing through it, and $\|\mathbf{u}_s\|_2$ is the L2 norm of $\mathbf{u}_s$. This formulation integrates both the magnitude and the uncertainty of the weight parameters, potentially yielding a more robust measure of feature importance under a probabilistic model. The vector $\mathbf{w}_s$ specifies the weights that define the oblique split at node s. The expected value of the weights, $\mathbb{E}[\mathbf{w}_s]$, is computed through Monte Carlo simulation, where weights are sampled M times from their approximation of the posterior distribution, and the mean of these samples is taken as the representative value for each weight. This integrates the stochastic nature of the model parameters into the calculation of feature importance. The repeated sampling and averaging process ensures that the estimated importance reflects both the central tendency and the variability of the weight parameters, crucial for interpreting the model's decision-making process under uncertainty.

Consider a simple dataset where two features, x_1 and x_2, predict the likelihood of purchasing a car. In a VSPYCT model, the influence of each feature at a node depends not only on the magnitude of its mean weight but also on how this mean compares to the feature's weight variance.

Suppose that at the root node, the mean weights associated with x_1 and x_2 are $\mathbb{E}[w_{x_1}] = 0.2$ and $\mathbb{E}[w_{x_2}] = 0.8$, and their respective variances are $\text{Var}(w_{x_1})$ and $\text{Var}(w_{x_2})$. The importance at this node will now reflect the ratio $\frac{\mathbb{E}[w]}{\text{Var}(w)}$, such that a feature with a relatively high mean weight but also a high variance will not necessarily dominate one with a slightly lower mean but substantially lower variance. Thus, if x_2 maintains a higher $\frac{\mathbb{E}[w]}{\text{Var}(w)}$ ratio than x_1, it indicates that x_2 is not only influential but also more reliable at this particular node.

However, the overall feature importance is derived by aggregating these normalised ratios across all splits within the tree. Even if x_2 appears strongly influential at the root node, the final importance calculation considers both the magnitude and the stability of the weights across all nodes. A feature consistently demonstrating a favourable $\frac{\mathbb{E}[w]}{\text{Var}(w)}$ ratio throughout multiple splits will be deemed more important globally. In this way, the final importance assessment accounts for both the strength

Table 2
Investigated parameter configurations for VSPYCT.

Parameter	Range tested	Description
Monte Carlo samples (M)	10 - 100	Approximation of posterior distributions
Variational iterations (I_{vb})	50 - 500	Convergence of ELBO optimization
Batch size (β)	32 - 256	Mini-batch size for optimization
Learning rate (λ)	0.001 - 0.01	Step size for gradient updates

Table 3
Properties of the benchmark STR datasets.

Dataset	N	D
era OpenML (2026)	1000	5
cpmp OpenML (2026)	2108	27
qsar234 OpenML (2026)	2145	1026
concrete_compressive_strength OpenML (2026)	1030	9
yprop OpenML (2026)	8885	252
puma8NH OpenML (2026)	8192	8
spacega OpenML (2026)	3107	6
bike_sharing OpenML (2026)	17,379	6
quake OpenML (2026)	2178	3
ailerons OpenML (2026)	13,750	40

and certainty of each feature's contribution to the model's decision-making process.

5. Experimental setting

The experimental evaluation was conducted to compare the VSPYCT model with several benchmarks, including the SPYCT model, OPCTs Stepisnik et al. (2020), and ensemble of PCTs. Specifically, we considered both a single SPYCT and an ensemble of SPYCTs, an ensemble of PCTs, and the OPCT model. We chose these methods as primary comparisons because they have been extensively evaluated in the literature, demonstrating strong performance across various SOP tasks Andonovikj et al. (2024), Stepišnik and Kocev (2021).

The comparison encompassed a range of predictive modelling tasks, including single-target regression, multi-target regression, binary classification, and multi-class classification. By evaluating against a single SPYCT tree, an ensemble of SPYCTs, OPCT tree, and an ensemble of PCTs, we aimed to assess how VSPYCT's predictive performance compares against a spectrum of models, from single decision trees to more complex ensembles and option trees.

5.1. Parameter configuration summary

Table 2 summarises the parameter ranges tested in this study and their relevance to the optimisation of VSPYCT.

After evaluating performance and computational trade-offs, the final chosen parameter values were set to $M = 100$, $I_{vb} = 100$, $\beta = 128$, and $\lambda = 0.01$.

5.2. Data

The experimental evaluation was performed on a diverse collection of datasets, each reflecting a distinct predictive modelling task: single-target regression (STR), multi-target regression (MTR), binary classification (BC), and multi-class classification (MCC). The properties of these benchmark datasets are detailed in Table 3, Table 4, Table 5, and Table 6. Each table lists the number of examples (N) and the number of features (D) for all datasets, along with the number of targets (T) for MTR datasets and the number of classes (C) for MCC datasets.

5.3. Evaluation

The performance of the models was evaluated using appropriate metrics for each predictive modelling task. For single-target regression, the

Table 4
Properties of the benchmark MTR datasets.

Dataset	N	D	T
wq OpenML (2026)	1060	16	14
atp1d OpenML (2026)	337	411	6
atp7d OpenML (2026)	296	411	6
scpf OpenML (2026)	1137	23	3
osales OpenML (2026)	639	411	12
sf1 OpenML (2026)	323	10	3
sf2 OpenML (2026)	1066	10	3
rf1 OpenML (2026)	9125	64	8
rf2 OpenML (2026)	9125	576	8
slump OpenML (2026)	103	7	3

Table 5
Properties of the benchmark BC datasets.

Dataset	N	D
scene Mulan (2026)	2407	299
ova_breast OpenML (2026)	1545	10,935
ova_lung OpenML (2026)	1545	10,935
bio_response OpenML (2026)	3751	1777
chronic_kidney_disease OpenML (2026)	400	25
compas_two_years OpenML (2026)	5278	13
spambase OpenML (2026)	4601	57
gina_agnostic OpenML (2026)	3468	970
credit-g OpenML (2026)	1000	20
diabetesOpenML (2026)	768	8

Table 6
Properties of the benchmark MCC datasets.

Dataset	N	D	C
amazon_reviews OpenML (2026)	1500	10,000	50
dermatology OpenML (2026)	366	34	6
micro_mass OpenML (2026)	360	1300	10
balance OpenML (2026)	625	4	3
yeast Mulan (2026)	1484	8	10
wine OpenML (2026)	178	13	3
mfeat_zernike OpenML (2026)	2000	47	10
har OpenML (2026)	10,299	561	6
gas_drift OpenML (2026)	13,910	128	6
semeion OpenML (2026)	1593	257	10

Mean Absolute Error (*MAE*) was employed as the evaluation metric:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

where y_i denotes the true value, $\hat{y}_i$ is the predicted value, and N is the total number of instances. A lower *MAE* indicates better predictive accuracy.

For multi-target regression, a normalised Mean Absolute Error (*NMAE*) was used to ensure comparability across targets with different value ranges. Letting y_{ij} and $\hat{y}_{ij}$ be the true and predicted values for the j-th target of the i-th instance, and T be the total number of target variables, the *NMAE* is defined as:

$$NMAE = \frac{1}{T}\sum_{j=1}^{T}\left(\frac{\frac{1}{N}\sum_{i=1}^{N}|y_{ij} - \hat{y}_{ij}|}{\max_i(y_{ij}) - \min_i(y_{ij})}\right)$$

If $\max_i(y_{ij}) - \min_i(y_{ij}) = 0$ for a given target j, this denominator is replaced by 1, effectively leaving that target unscaled.

A lower *NMAE* indicates better predictive performance, as it accounts for the range of each target variable to produce a comparable error measure across multiple targets.

For binary and multi-class classification tasks, the *F1* score was used to measure model performance. In the case of multi-class classification, the *F1* score was macro-averaged across all classes. A higher *F1* score indicates better classification performance, as it reflects a balance between precision and recall.

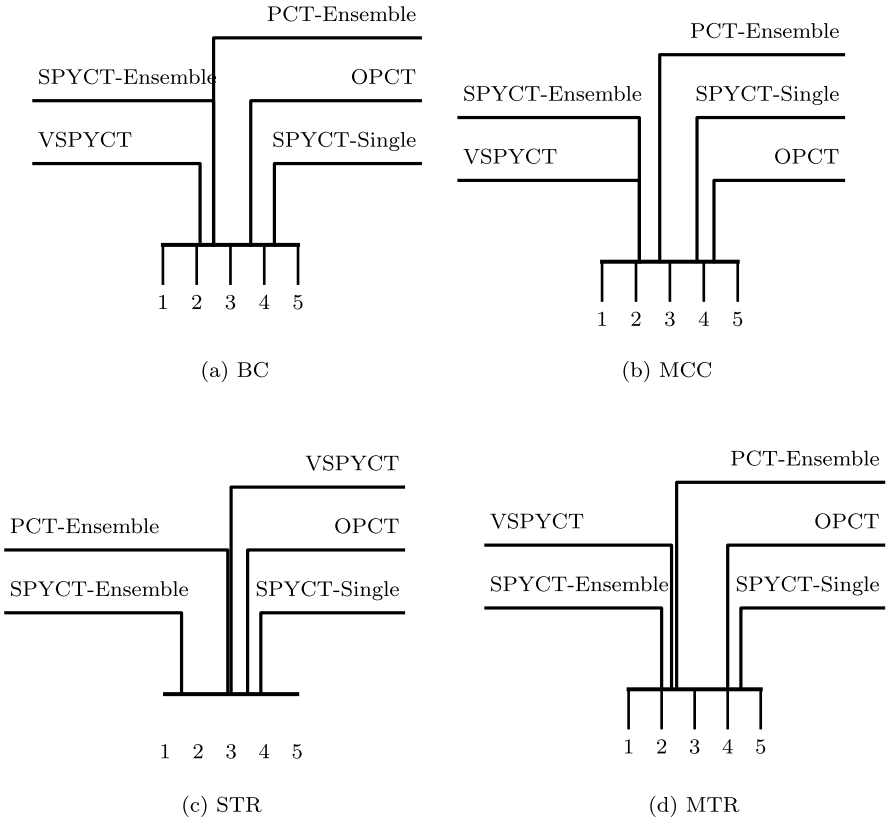

Fig. 2. Average rank of predictive performance. Lower ranks indicate better performance.

All evaluations were conducted using 5-fold cross-validation, to ensure a comprehensive assessment of the model's performance across different subsets of the data, reducing the potential for bias and overfitting.

6. Results

Fig. 2 shows the average rankings of the evaluated methods VSPYCT, OPCT, single SPYCT, ensemble of SPYCTs, and ensemble of PCTs across four task types. For improved readability, the actual performance of the models on each dataset is given in A.

In BC tasks, VSPYCT ranks first, outperforming ensembles of SPYCTs and PCTs, which tie for second. This suggests that VSPYCT's probabilistic modeling effectively handles simple binary boundaries, surpassing both ensemble variance reduction and structural options (OPCT). In MCC tasks, VSPYCT and the ensemble of SPYCTs lead, followed by the ensemble of PCTs, single SPYCT, and OPCT, indicating that VSPYCT's uncertainty-aware splits can handle more complex class structures as well as ensembles can.

For STR tasks, both ensemble-based approaches-those combining multiple SPYCTs and those combining multiple PCTs-achieve the highest ranks, followed by VSPYCT. It can be seen that ensembles appear particularly effective for precise single-target continuous predictions. However, in MTR scenarios, VSPYCT surpasses the ensemble of PCTs, indicating that its probabilistic representation of oblique splits can better capture complex inter-target relationships. By contrast, OPCT and single SPYCT lag behind, indicating that their individual enhancements-option nodes or a single-tree oblique structure-do not offer sufficient complexity or flexibility to match the performance gains achieved by either ensembles or the probabilistic modelling of VSPYCT.

6.1. Discussion

The results underscore the interplay between model complexity, uncertainty quantification, and variance reduction. Ensemble methods consistently perform well in scenarios demanding high precision (e.g., STR) or broad generalisation over multiple outputs (MTR). VSPYCT, capitalises on its Bayesian inference framework to excel in tasks where modeling uncertainty and complex decision boundaries are crucial (e.g., BC and MCC), and remains competitive even in more challenging regression contexts.

These findings underscore that VSPYCT is not a universal replacement for ensembles, but rather a complementary alternative whose benefits depend on the task at hand. When dealing with classification problems-binary or multi-class-where interpretability and uncertainty quantification are critical, VSPYCT offers a potent blend of predictive performance, clarity, and inherent uncertainty modeling. Its capability to handle intricate class boundaries and its flexibility in representing parameter uncertainties make it a valuable method for domains demanding transparency and robustness.

In continuous-output scenarios, ensembles still exhibit an edge, suggesting that if raw predictive precision is paramount and interpretability or uncertainty modeling are secondary considerations, ensemble models may remain the preferred choice. Nevertheless, VSPYCT provides a single-tree model that narrows the performance gap considerably and is particularly appealing when balancing predictive accuracy with the need for more direct insight into the model's decision process.

While the present work marginalises only over split-parameter posteriors, future work will adopt Bayesian non-parametric priors over tree topologies to capture structural, as well as parameter, uncertainty. This would allow the model to reason not only about decision-boundary uncertainty, but also about the tree structure itself (i.e., how many splits to create and where to place them). Possible approaches include integrating structural priors, or developing ensemble-like variational methods that maintain a distribution over multiple plausible tree structures. Such extensions would provide a more comprehensive Bayesian treatment of decision trees and further enhance both predictive reliability and interpretability.

An explicit empirical comparison of training time and scalability between VSPYCT and deterministic approaches was beyond the scope of this study, due to implementation-specific and hardware-dependent factors.

Advantages of VSPYCT:

- Integrated interpretability and uncertainty quantification.
- Competitive predictive performance compared to ensembles.
- Effective handling of complex classification boundaries.

Disadvantages and Limitations:

- Higher computational complexity due to variational inference.
- Potential scalability issues with extremely large or high-dimensional datasets.
- Limited capability to represent structural uncertainty compared to ensembles.

6.2. Analysis of VSPYCT performance under varying dataset properties

In order to identify which types of datasets and predictive tasks VSPYCT handles well or struggles with, we correlated each dataset's final performance metric (MAE/NMAE or F1) against its size (N) and number of features (D), as summarised in Table 7.

Table 7
Spearman correlation between dataset size (N), number of features (D), and VSPYCT performance (MAE/NMAE for regression and F1 for classification).

Task	Corr. w.r.t. N		Corr. w.r.t. D	
	ρ	p	ρ	p
STR	−0.345	0.328	−0.188	0.602
MTR	−0.231	0.521	0.068	0.853
BC	0.286	0.424	−0.456	0.185
MCC	−0.152	0.676	−0.055	0.881

Using Spearman correlation to relate VSPYCT's scores to N and D, we observed only weak or moderate (and statistically insignificant) relationships overall. For single-target regression, for instance, a small negative correlation with N and D indicates that increasing dataset size or dimensionality does not systematically improve or degrade VSPYCT's predictive error – exemplified by its success on both large, high-dimensional tasks such as *qsar234*, as well as on relatively smaller, simpler problems like *quake*. In multi-target regression, similarly negligible correlations suggest that VSPYCT's performance does not critically depend on the sheer scale of the data; it produces low NMAE on cases such as *scpf* (modest size and dimensionality) as well as *rf1* and *rf2* (substantial size and feature counts), revealing that its oblique splits can effectively model multiple continuous targets even when dimensionality is high. Turning to binary classification, where higher F1 scores are better, a slight positive correlation with N implies that larger datasets might offer marginal gains, while a mild negative correlation with D reflects that VSPYCT can maintain robust performance despite high-dimensional problems (e.g., *ova_lung* with over 10 000 features). Finally, the multi-class classification results also show near-zero correlation with both N and D, meaning VSPYCT can handle diverse complexities without clear dependence on data size or feature count alone, as seen by strong F1 on both smaller sets (e.g., *dermatology*) and larger ones (e.g., *gas_drift*). Across all four predictive tasks, the absence of any strong or statistically significant correlation with N or D underscores that VSPYCT's method of oblique, probabilistic splits adapts broadly: it is more successful in classification tasks, and less consistently dominant in single-target regression problems demanding extremely low errors (for example, on *bike_sharing* where ensembles tend to excel). Nonetheless, even in regression, VSPYCT frequently remains competitive, especially on datasets with moderate variance, making it an appealing single-model alternative when interpretability and uncertainty quantification are valued.

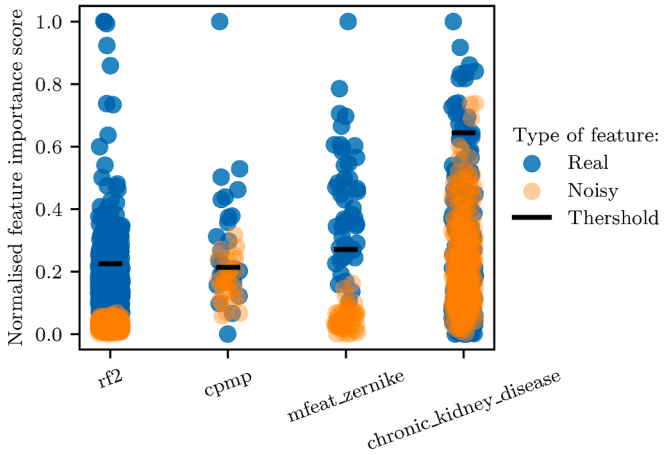

Fig. 3. Feature importance scores for both the original and noise-augmented features. Blue markers represent the importance of original features, while orange markers correspond to noise features. The horizontal threshold line marks the point where removing additional features starts degrading model performance. Features below this threshold are non-essential, as their removal improves the evaluation score-demonstrating that noise features (orange) contribute little to model performance. In contrast, features above the threshold are crucial, as removing them further degrades performance.

6.3. Robustness to spurious features

A critical step in validating the proposed VSPYCT model is to test its robustness against spurious features. In many real-world scenarios, the data may contain variables that do not contribute to the predictive task at hand – hence, it is paramount to ensure that the model assigns low importance to such non-informative features.

Fig. 5 highlights the feature importance scores obtained from the VSPYCT model, focusing on the most influential features that contribute to predicting the unemployment duration. The horizontal bars show the relative importance of each feature, with *Entry month* emerging as the most critical factor, followed by *Months of work experience* and *ISCO_Health associate professionals*.

As illustrated in Fig. 3, the VSPYCT-derived importance measures generally assign higher scores to features that provide genuine predictive information. To rigorously evaluate the extent to which the model remains robust to irrelevant variables, let the original dataset contain d features, denoted by the set $\mathcal{F} = \{f_1, f_2, \ldots, f_d\}$. We augment this dataset by adding d synthetic noise features $\mathcal{F}' = \{f_{d+1}, \ldots, f_{2d}\}$, each of which is drawn independently from a Gaussian distribution with mean 0 and variance 1:

$$f_{d+j} \sim \mathcal{N}(0, 1), \quad \text{for } j = 1, \ldots, d.$$

Hence, the total number of features in the augmented dataset is $2d$.

Fig. 3 reveals that most of the original (real) features exhibit a broad range of importance values. This heterogeneity is expected because many real-world datasets naturally include some highly informative predictors, as well as others that provide minimal predictive power.

To assess the significance of individual features in our model, we conducted an iterative feature removal process inspired by Guyon and Elisseeff (2003). We aim to identify the subset of features that meaningfully contribute to the model's predictive performance while eliminating those that do not. Starting with the full set of features, we trained our model and recorded its evaluation score. Then, we systematically removed features in ascending order of their importance scores-starting with the least important feature and retraining the model after each removal. This iterative process continued until all features were removed. We set a threshold, which distinguishes important from non-important features, as the feature importance score at which the model performance begins to degrade.

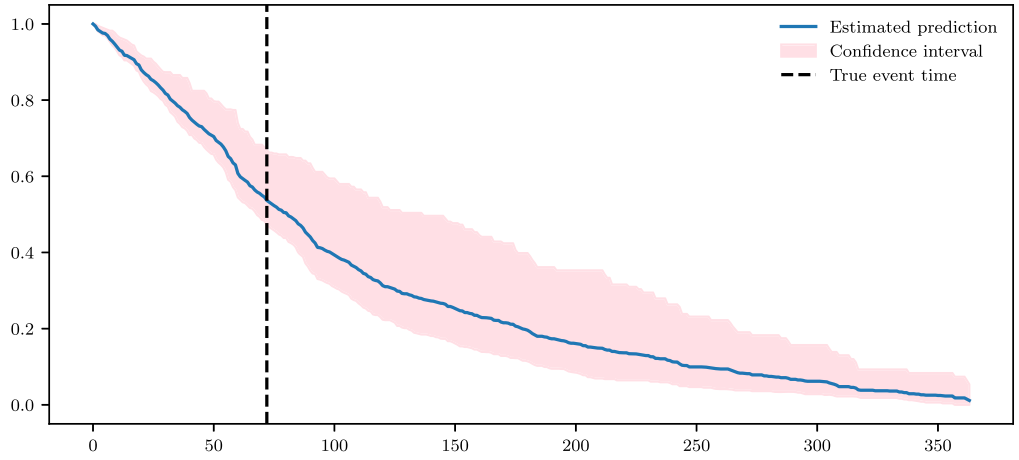

Fig. 4. Survival curve predicted by the VSPYCT model on the Unemployment dataset. The blue line represents the estimated mean survival time, while the shaded area indicates the confidence interval (10th to 90th percentile) around the prediction. The vertical dashed line marks the actual time of the observed event.

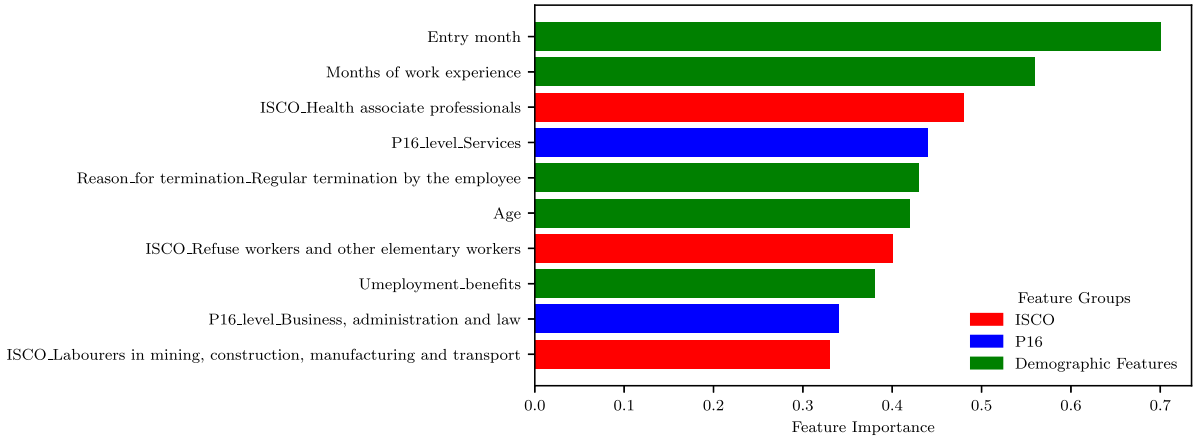

Fig. 5. Feature importance scores extracted from the VSPYCT model applied to the Unemployment dataset. The horizontal bars represent the relative importance of each feature, with the most influential feature being "Entry month".

Importantly, the noise features in Fig. 3 typically receive lower importance values, confirming that VSPYCT can detect their lack of relevance. Formally, if $I(f)$ denotes the estimated importance of a feature f, then

$$I(f_j) \ll I(f_k) \quad \text{for at least a subset of } f_j \in \mathcal{F}' \text{ and } f_k \in \mathcal{F}.$$

In other words, noise features $\{f_{d+1}, \ldots, f_{2d}\}$ are consistently outperformed by at least some subset of the original features.

Noteworthy, a few original features may show importance values that are comparable to or even lower than those of some noise features. This outcome does not necessarily imply that the noise features carry genuine signal; rather, it highlights that certain original features are also uninformative for the target variable. These results confirm that the VSPYCT-derived feature importance scores reliably rank truly predictive variables above most spurious ones, underscoring the model's robustness to irrelevant inputs.

6.4. Practical interpretation of results

The empirical results presented earlier can be further understood by considering their practical implications and physical meaning in real-world contexts. For regression tasks, such as predicting concrete compressive strength, the improved accuracy of VSPYCT indicates that probabilistic modeling of uncertainty in feature weights enables better handling of measurement variability or experimental errors common in physical systems. This modeling approach allows practitioners, for example civil engineers, to better anticipate the reliability of material properties predicted by the model.

Similarly, classification results can be interpreted from a practical perspective by analysing the decision boundaries constructed by VSPYCT. For instance, in medical diagnosis (such as chronic kidney disease), the high predictive performance reflects the model's capacity to clearly delineate clinically meaningful groups, effectively capturing critical diagnostic thresholds that physicians rely on for clinical decision-making.

Moreover, the results obtained for employment duration prediction offer a direct practical interpretation: demographic and occupational splits made by VSPYCT align closely with known socio-economic factors affecting employment outcomes. Thus, the proposed model effectively incorporates and reveals practically meaningful dynamics from the data, enhancing both interpretability and applicability.

6.5. Interpretability

The dataset used for the interpretability analysis consists of 74,086 anonymised instances of jobseekers from the Slovenian Public Employment Service (PES). We will refer to it as the Unemployment dataset. It includes a variety of personal and professional characteristics, such as age, gender, education, and work experience. The target variable is the time until the jobseeker either becomes employed or exits the study, which is measured in days. The dataset is challenging due to the different forms of its attributes (categorical, numerical, temporal) and the

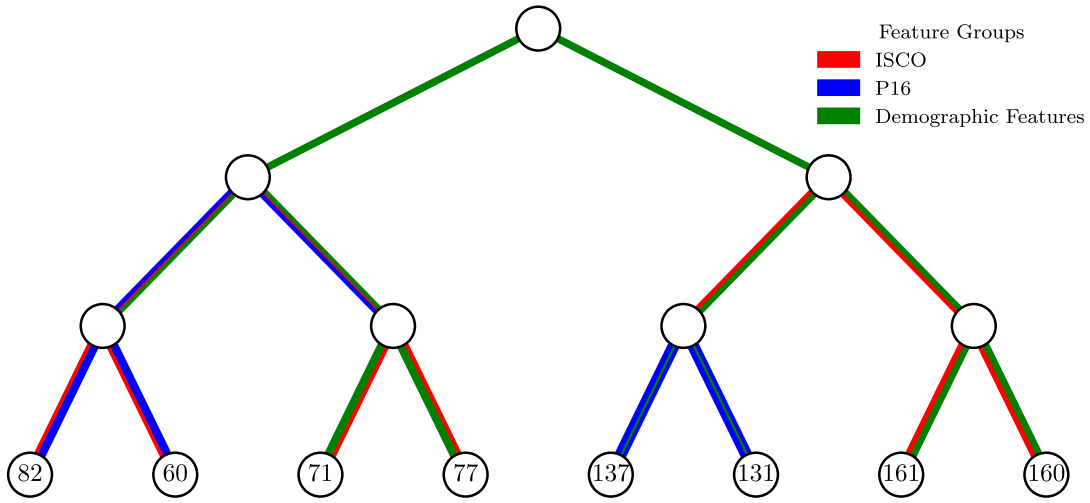

Fig. 6. The VSPYCT model applied to the Unemployment dataset. The colouring distinguishes the different feature groups of the data set.

presence of censored data, where some jobseekers' outcomes are not fully observed.

The goal is to predict the time-to-event (employment) for jobseekers, with some records being right-censored-meaning the exact time of the event is not observed. Each data instance is described by features $\mathbf{X}_i$, observed time t_i, and a censoring indicator δ_i, where $\delta_i = 1$ indicates the event was observed, and $\delta_i = 0$ indicates a right-censored record. This setup, with its inherent missing data due to censoring, makes it suitable for semi-supervised learning approaches. Detailed information on the dataset and the semi-supervised multi-target regression framework can be found in Andonovikj et al. (2024).

Fig. 4 presents the survival curve predicted by the VSPYCT model for a specific jobseeker in the Unemployment dataset. The curve shows the estimated probability of remaining unemployed over time, with the blue line representing the mean prediction. The shaded area around the curve illustrates the uncertainty in the prediction, captured as a confidence interval between the 10th and 90th percentiles. This visualisation effectively communicates the model's predictions along with its confidence, providing valuable insights into the jobseeker's likelihood of finding employment over time. The dashed vertical line marks the actual observed time of employment, allowing for an immediate comparison between the predicted and actual outcomes.

Fig. 6 presents the visual representation of the VSPYCT applied to the Unemployment dataset. It provides clear insights into how different feature groups influence predictions of unemployment duration. Each node represents a decision point, with the edges coloured based on the dominant feature group-*ISCO* (occupational classifications), *P16* (education-related features), or demographic features-used in the split. This colour-coded representation highlights the relative importance of feature groups across different levels of the tree.

Demographic features dominate higher levels of the tree, indicating their broad and primary influence on unemployment duration. Features such as *Months of work experience* and *Age* frequently appear at the root or upper nodes, suggesting that prior professional experience and age are strong, general predictors of re-employment likelihood. For instance, individuals with less work experience are often routed towards branches with longer unemployment times, reflecting the challenges faced by those lacking experience in securing jobs.

In contrast, ISCO features become more prominent in lower-level splits, capturing occupation-specific influences on unemployment. Branches dominated by ISCO-related splits suggest that certain occupational groups have distinct re-employment trends. For example, occupations such as *Health associate professionals* or *Labourers in mining, construction, manufacturing, and transport* are likely associated with specific labor market demands that either shorten or extend unemployment durations.

P16 features emerge in intermediate levels of the tree, reflecting the nuanced role of education in employment outcomes. For instance, splits involving education levels such as *Business, administration, and law* or *Services* suggest that individuals' educational backgrounds influence their unemployment duration in ways that interact with occupational or demographic factors.

The leaf nodes, representing the predicted expected survival times (i.e., unemployment duration in days), provide a clear endpoint for each pathway. Branches leading to longer survival times often involve demographic splits, highlighting structural barriers such as insufficient work experience or higher age, while branches associated with shorter durations often incorporate ISCO and P16 splits, reflecting the effect of occupation- and education-specific factors.

In essence, the VSPYCT model's tree structure, with its intuitive colour-coded feature categories and quantified predictions at the leaf nodes, offers a powerful tool for understanding the multifaceted nature of unemployment. It bridges the gap between complex machine learning models and practical socio-economic applications, providing a clear, actionable framework for addressing unemployment through informed, data-driven strategies.

6.5.0.1. Practical use for domain experts. After training, VSPYCT provides three complementary outputs that can be inspected with standard analysis tools:

1. **Node summaries:** for every split we report the posterior mean, standard deviation, and 95% credible interval of each coefficient, allowing analysts to verify which variables dominate within a specific branch.
2. **Tree file:** the complete tree is exported in a plain, machine-readable graph format so that users can open it in any network-visualisation environment and interactively traverse, fold, or unfold decision paths.
3. **Global relevance ranking:** the importance scores (see Fig. 5) order the features by their expected effect on the prediction, helping experts prioritise factors for monitoring or intervention.

7. Conclusion

We introduced VSPYCTs, a novel framework combining Bayesian inference with predictive clustering trees. VSPYCTs uniquely incorpo-

rate variational Bayes into oblique splits, providing built-in uncertainty quantification and preserving interpretability.

Our experimental evaluations demonstrated that VSPYCTs outperform ensemble methods in binary and multi-class classification tasks, while remaining competitive in multi-target and single-target regression scenarios. For example, VSPYCT achieved the highest average rank in binary classification tasks and tied for best in multi-class classification, confirming its robustness and generalisation capabilities.

The primary advantages of VSPYCT include inherent interpretability, integrated uncertainty quantification, and competitive predictive performance without requiring ensembles. However, its main disadvantage is increased computational complexity due to variational inference optimisation and Monte Carlo sampling.

Future research should explore methods to reduce computational complexity, integrate structural priors to handle structural uncertainty, and extend VSPYCT to broader domains and larger datasets, improving scalability and general applicability.

CRediT authorship contribution statement

Viktor Andonovikj: Writing – review & editing, Formal analysis, Conceptualization; **Sašo Džeroski:** Supervision, Visualization, Writing – review & editing; **Biljana Mileva Boshkoska:** Writing – review & editing, Supervision, Conceptualization, Formal analysis; **Pavle Boškoski:** Writing – review & editing, Supervision, Conceptualization, Formal analysis, Data curation.

Data availability

The data that has been used is confidential.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors acknowledge the research core funding No. P2-0001, and V5-24020 financially supported by the Slovenian Research and Innovation Agency, and the Ministry of the Economy, Tourism and Sport of the Republic of Slovenia.

Appendix A. Detailed results

Detailed results of the performance of the models on the different datasets are given in Tables A.1, A.2, A.3, and A.4

Table A.1
MAE scores for STR (lowest is best).

Dataset	SPYCT-single	SPYCT-ensemble	VSPYCT	PCT-Ensemble	OPCT
era	1.296625	**1.249352**	1.287840	1.277622	1.277424
cpmp2015	1137.738259	997.261786	918.481147	1211.124409	**848.721320**
qsar234	0.572577	**0.415656**	0.428724	0.428721	0.441020
puma8NH	2.795463	**2.632242**	2.711678	2.734472	2.769841
yprop	0.020126	0.019693	0.019969	**0.019682**	0.023687
space_ga	0.105716	0.090193	0.104501	**0.084793**	0.099387
bike_sharing	7.866288	3.101784	11.288880	10.610927	**2.509829**
quake	0.144549	0.143972	**0.143805**	0.147743	0.172350
concrete_compressive_strength	4.500918	**4.121670**	7.740371	4.164038	4.216046
ailerons	0.000135	**0.000124**	0.000127	0.000133	0.000161

Table A.2
NMAE scores for MTR (lowest is best).

Dataset	SPYCT-single	SPYCT-ensemble	VSPYCT	PCT-Ensemble	OPCT
wq	0.184654	**0.167491**	0.183236	0.169793	0.175325
atp7d	0.321386	0.157284	0.167780	**0.147780**	0.176141
atp1d	0.229711	**0.104547**	0.125300	0.106302	0.115277
scpf	0.031613	0.027816	**0.027291**	0.047811	0.052046
osales	0.068986	0.064457	**0.057865**	0.060054	0.058278
sf2	0.042276	0.038311	**0.037199**	0.039234	0.041858
sf1	0.097729	0.095937	**0.092266**	0.108493	0.109708
slump	0.182132	**0.165759**	0.239199	0.175965	0.243881
rf1	0.197493	0.183436	**0.166669**	0.167392	0.168493
rf2	0.212579	**0.176247**	0.193914	0.177062	0.194180

Table A.3
F1 scores for BC (highest is best).

Dataset	SPYCT-single	SPYCT-ensemble	VSPYCT	PCT-Ensemble	OPCT
scene	0.944059	**0.986098**	0.972439	0.882295	0.935982
ova_breast	0.928256	0.946192	0.959834	**0.966468**	0.956409
bioresponse	0.729417	0.777438	0.768986	**0.777509**	0.753144
ova_lung	0.802281	0.905439	**0.920463**	0.886492	0.904865
chronic_kidney_disease	0.961777	0.991777	**1.000000**	0.997178	0.997178
compas_two_years	0.645271	0.641390	0.652540	**0.666824**	0.615251
spambase	0.903289	**0.934069**	0.926491	0.924987	0.912740
gina_agnostic	0.856799	0.885768	0.858579	0.909199	**0.922184**
credit-g	0.632353	0.657097	**0.681566**	0.615338	0.547619
diabetes	0.698525	0.739710	0.715029	**0.778026**	0.694843

Table A.4
F1 scores for MCC (highest is best).

Dataset	SPYCT-single	SPYCT-ensemble	VSPYCT	PCT-Ensemble	OPCT
amazon_reviews	0.143184	0.331889	0.172791	**0.333139**	0.293184
dermatology	0.852633	0.978408	**0.980117**	0.893882	0.922443
micro_mass	0.845445	0.940100	**0.954908**	0.940397	0.905830
balance	0.645499	0.618803	**0.825152**	0.595340	0.585561
yeast	0.443345	0.535824	0.471005	**0.575664**	0.443048
wine	0.975580	0.965350	0.966350	**1.000000**	0.945825
mfeat_zernike	0.786454	**0.804786**	0.792283	0.759459	0.693469
har	0.969661	0.982193	**0.984686**	0.942095	0.942095
gas_drift	0.974804	**0.993256**	0.988948	0.988421	0.981730
semeion	0.846045	**0.923838**	0.868518	0.897004	0.811956

Appendix B. Evaluation score in relation to the number of removed features

The following graphs show how the predictive performance of the VSPYCT model changes as features are iteratively removed, starting from the least important.

Fig. B.1 illustrates the effect of iterative feature removal on the *rf2* dataset.

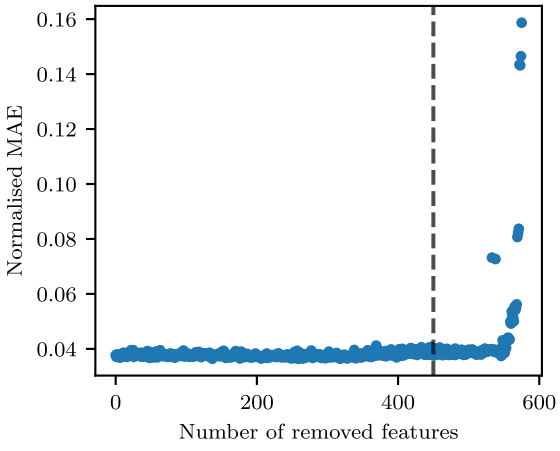

Fig. B.1. Dataset: rf2.

Fig. B.2 shows the performance degradation on the *cpmp2015* dataset.

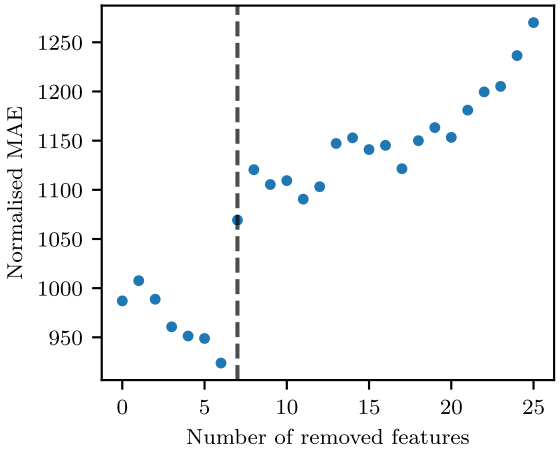

Fig. B.2. Dataset: cpmp2015.

Fig. B.3 depicts the model's predictive performance as features are removed from the *mfeat_zernike* dataset.

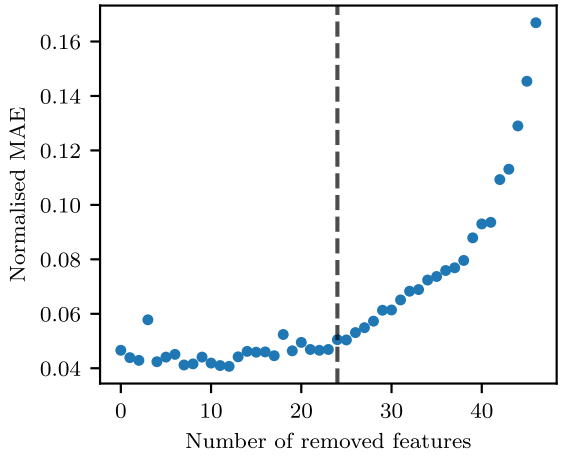

Fig. B.3. Dataset: mfeat_zernike.

Fig. B.4 demonstrates the impact of feature removal on predictive accuracy for the *chronic_kidney_disease* dataset.

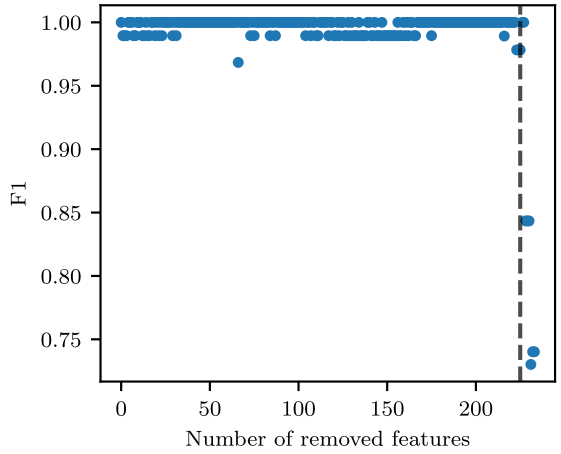

Fig. B.4. Dataset: chronic_kidney_disease.

References

Abbas, S., Ahmad, M., Nazar, M., Saleem, S., Isyanov, R., Aljedani, J., & Garalleh, H. A. L. (2025a). Artificial neural network analysis of heat and mass transfer in fractional casson flow. *Case Studies in Thermal Engineering*, *69*, 105946. https://doi.org/10.1016/j.csite.2025.105946

Abbas, S., Inam, I., Alharthi, A. M., AL-Khasawneh, M. A. S., Az-Zo'bi, E. A., & Garalleh, H. A. L. (2025b). Fractional magnetohydrodynamic casson fluid flow with thermal radiation and buoyancy effects: A constant proportional caputo model. *Boundary Value Problems*, *2025*(1). https://doi.org/10.1186/s13661-025-02036-4

Abbas, S., & Nazar, M. (2024). Fractional analysis of unsteady magnetohydrodynamics jeffrey flow over an infinite vertical plate in the presence of hall current. *Mathematical Methods in the Applied Sciences*, *48*(1), 253–272. https://doi.org/10.1002/mma.10326

Andonovikj, V., Boskoski, P., Evkoski, B., Redek, T., & Mileva Boshkoska, B. (2022). Community analysis in slovenian labour network 2010–2020. *Journal of Decision Systems*, *31*(sup1), 308–318. https://doi.org/10.1080/12460125.2022.2070944

Andonovikj, V., Boškoski, P., Džeroski, S., & Boshkoska, B. M. (2024). Survival analysis as semi-supervised multi-target regression for time-to-employment prediction using oblique predictive clustering trees. *Expert Systems with Applications*, *235*, 121246. https://doi.org/10.1016/j.eswa.2023.121246

Bernardo, J. M., Bayarri, M. J., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A., West, M. et al. (2003). The variational bayesian EM algorithm for incomplete data: With application to scoring graphical model structures. *Bayesian Statistics*, *7*(453–464), 210.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, *112*(518), 859–877. https://doi.org/10.1080/01621459.2017.1285773

Boškoski, P., Perne, M., Rameša, M., & Boshkoska, B. M. (2021). Variational bayes survival analysis for unemployment modelling. *Knowledge-Based Systems*, *229*, 107335. https://doi.org/10.1016/j.knosys.2021.107335

Breiman, L. (2001). *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/a:1010933404324

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5). https://doi.org/10.1214/aos/1013203451

Gershman, S., & Goodman, N. (2014). Amortized inference in probabilistic reasoning. In *Proceedings of the annual meeting of the cognitive science society*. (*vol. 36*).

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*(Mar), 1157–1182.

Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*.

Kim, Y., Wiseman, S., Miller, A., Sontag, D., & Rush, A. (2018). Semi-amortized variational autoencoders. In *International conference on machine learning* (pp. 2678–2687). PMLR.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv:1412.6980.

Kocev, D., Ceci, M., & Stepišnik, T. (2020). Ensembles of extremely randomized predictive clustering trees for predicting structured outputs. *Machine Learning*, *109*(11), 2213–2241. https://doi.org/10.1007/s10994-020-05894-4

Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, *46*(3), 817–833. https://doi.org/10.1016/j.patcog.2012.09.023

Le, T. A., Baydin, A. G., & Wood, F. (2017). Inference compilation and universal probabilistic programming. In *Artificial intelligence and statistics* (pp. 1338–1348). PMLR.

Lopez, L., Rudner, T. G. J., & Shamout, F. E. (2023). Informative priors improve the reliability of multimodal clinical data classification. arXiv:2312.00794.

Mulan (2026). Mulan: A java library for multi-label learning. http://mulan.sourceforge.net/datasets.HTML. Accessed: 2020-04-15.

Murphy, K. P. (2023). Probabilistic machine learning: Advanced topics. MIT press.

OpenML (2026). OpenML. https://www.openml.org. Accessed: 2020-04-15.

Plumb, G., Molitor, D., & Talwalkar, A. S. (2018). Model agnostic supervised local explanations. *Advances in Neural Information Processing Systems*, *31*.

Ramzan, M., Shafique, A., Abbas, S., Ali, R., Lamoudan, T., Jan, R., Norberdiyeva, M., Turabova, B., & Garalleh, H. A. L. (2025). Mathematical analysis of prandtl number with artificial neural network and fractional operator for nanofluid flow. *Case Studies in Thermal Engineering*, *71*, 106164. https://doi.org/10.1016/j.csite.2025.106164

Sashank, J. R., Satyen, K., & Sanjiv, K. (2018). On the convergence of adam and beyond. In *International conference on learning representations*. (*vol. 5*).

Stepisnik, T., Osojnik, A., Dzeroski, S., & Kocev, D. (2020). Option predictive clustering trees for multi-target regression. *Computer Science and Information Systems*, *17*(2), 459–486. https://doi.org/10.2298/csis190928006s

Stepišnik, T., & Kocev, D. (2021). Oblique predictive clustering trees. *Knowledge-Based Systems*, *227*, 107228. https://doi.org/10.1016/j.knosys.2021.107228

Tan, S., Soloviev, M., Hooker, G., & Wells, M. T. (2020). Tree space prototypes: Another look at making tree ensembles interpretable. In *Proceedings of the 2020 ACM-IMS on foundations of data science conference* FODS '20. ACM. https://doi.org/10.1145/3412815.3416893.

Zhao, X., Wu, Y., Lee, D. L., & Cui, W. (2019). Iforest: Interpreting random forests via visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, *25*(1), 407–416. https://doi.org/10.1109/tvcg.2018.2864475.