Gaussian Soft Decision Trees for Interpretable Feature-Based Classification

Jaemin Yoo[1] and Lee Sael[2*]

[1] Seoul National University, Seoul, South Korea
jaeminyoo@snu.ac.kr
[2] Ajou University, Suwon, South Korea
sael@ajou.ac.kr
* Corresponding author

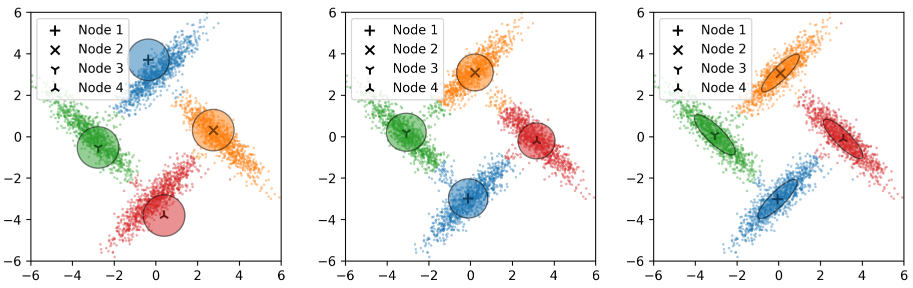
Abstract. How can we accurately classify feature-based data such that the learned model and results are more interpretable? Interpretability is beneficial in various perspectives, such as in checking for compliance with exiting knowledge and gaining insights from decision processes. To gain in both accuracy and interpretability, we propose a novel tree-structured classifier called Gaussian Soft Decision Trees (GSDT). GSDT is characterized by multi-branched structures, Gaussian mixture-based decisions, and a hinge loss with path regularization. The three key features make it learn short trees where the weight vector of each node is a prototype for data that mapped to the node. We show that GSDT results in the best average accuracy compared to eight baselines. We also perform an ablation study of the various structures of covariance matrix in the Gaussian mixture nodes in GSDT and demonstrate the interpretability of GSDT in a case study of classification in a breast cancer dataset.

Keywords: Gaussian Soft Decision Trees · Interpretable machine learning · Feature-based classification · Tabular data · Gaussian mixtures

1 Introduction

The interpretability of a model and its predictions is often an important factor in choosing machine learning models in various domains. Interpretable machine learning allows us to understand a decision process or the cause of a decision that can advance our understanding of the problem at hand [13, 18]. Furthermore, in specific domains such as biology and medicine, there are numerous feature-based data where each feature is meaningful and conveys unique information. These data require interpretable models, which is why less accurate models with interpretable structures such as decision trees are still being widely used.

Decision trees and linear models are representative models where the decision process and the importance of features are intrinsically human-understandable, but have limited representation power. Recent advancement of decision trees is soft decision trees (SDT) [6] that improve the representation power of decision trees by performing soft decisions using all input features at each node. However, the interpretability of an SDT is limited, because it requires a large depth



(a) Identity. (b) Only diagonal. (c) Low-Rank Perturbed.

Fig. 1: Comparison of Gaussian covariances of various structures on a synthetic dataset. Each point represents a data example with its label as a color, and each ellipse represents a leaf distribution learned by GSDT. The diagonal covariance with low-rank perturbations in (c) matches the true distribution better than the (a) identity and (b) diagonal covariances, resulting in higher accuracy.

to learn complex decision rules, involving many branches for interpreting each prediction. Unlike decision trees that use only one feature at each branch with a hard threshold, the large depth of SDTs leads to a complex decision process that is difficult to interpret even with the tree structure.

In this work, we propose Gaussian Soft Decision Trees (GSDT), our novel tree model that parameterizes each tree node as a Gaussian mixture, boosting the limited accuracy of previous tree-structured models without sacrificing the interpretability. Each edge in GSDT represents a multivariate Gaussian distribution parameterized by the learnable mean and covariance, which summarizes the examples that pass through it as an interpretable prototype. This makes it possible for GSDT to naturally adopt a multi-branched structure where each edge is learned and interpreted independently of the other edges in the branch. As a result, GSDT shows at least 4.8% higher average accuracy and 4× smaller depth compared to SDT-based models in six feature-based datasets. GSDT outperforms even black box models such as random forests and multilayer perceptrons that are not interpretable but have large representation power.

The contributions of this work are summarized as follows:

– **Model:** We propose GSDT, our novel tree-structured model that supports high interpretability as well as high accuracy by modeling the internal nodes as Gaussian mixtures. We propose various options of modeling the Gaussian covariance and compare them as in Figure 1.
– **Experiments:** We demonstrate the superior performance of GSDT by extensive experiments on six feature-based datasets, where both interpretability and classification accuracy are important.
– **Case study:** We analyze the structure and learned parameters of GSDT and shows its superior interpretability by demonstrating the decision process on an actual example on a breast cancer dataset with visualizations.

The rest of this paper is organized as follows. We introduce related works in Section 2. We propose GSDT in Section 3 with theoretical analysis in Section 4. We show experimental results in Section 5 and conclude at Section 6. The codes and datasets are publicly available at https://github.com/leesael/GSDT.

2 Related Works

Soft Decision Trees. Soft decision trees (SDT) [11] are tree-structured models that perform soft decisions. The internal nodes of SDTs are generalized linear classifiers [12] that pass input features through the tree structure, and the leaves learn fixed distributions over classes. With the improved representation power and interpretable nature, SDTs have been applied for various applications such as generative learning [10] and distilling the knowledge of deep neural networks [6]. EDiT [21] is a variant of SDTs, which improves the interpretability of SDTs by imposing sparsity on tree nodes and weight vectors.

Hierarchical Gaussian Mixture Models. Hierarchical Gaussian mixture models (HGMM) [5, 15, 19] are Gaussian mixture models (GMM) structured as a tree. HGMMs improve the efficiency of GMMs by stacking multiple layers of Gaussian components, instead of increasing the number of components horizontally. However, such models use all Gaussian components for the prediction of each example $\mathbf{x}$, making it difficult to interpret the decision process; it is required to examine all components in the model for explaining each decision.

Kernel Methods. Various machine learning algorithms adopt kernel functions to generalize linear decisions by mapping input features to another space where clear separations of classes are possible [1, 9]. SVM with the radial basis function (RBF) kernel [2] is one of the most famous kernel methods, which learns a decision boundary based on the Euclidean distance between features. Kernel logistic regression [22] generalizes logistic regression by applying kernel functions to the weight vectors instead of examples. Kernel methods improve the accuracy of linear models, but degrade the interpretability due to the nonlinearity.

3 Proposed Approach

We introduce Gaussian Soft Decision Trees (GSDT), our novel tree model that makes Gaussian mixture-based decisions at the internal nodes to maximize the accuracy while gaining in interpretability.

3.1 Overview

GSDT is represented as a multi-branched tree of depth d, where each node has b children. Each internal node i computes the probability of passing a feature $\mathbf{x}$ to its child node j as a function f_{ij} such that the sum of outgoing probabilities is one. GSDT passes $\mathbf{x}$ through all branches in the tree until it reaches the b^{d-1} leaf nodes where the arrival probability vector $\mathbf{r}(\mathbf{x})$ is computed. In other words,

$r_j(\mathbf{x})$ represents the probability of $\mathbf{x}$ arriving at leaf node j and is computed as the multiplication of all decision probabilities in the path from the root.

Each leaf node j has a probability distribution $\mathbf{q}_j \in \mathbb{R}^{|\mathcal{Y}|}$, where $\mathcal{Y}$ is the set of target classes, which does not change for the input $\mathbf{x}$ once it is learned. The k-th element of $\mathbf{q}_j$, which is the prediction for class $k \in \mathcal{Y}$, is defined as

$$q_{jk} = \frac{\exp(u_{jk})}{\sum_{l \in \mathcal{Y}} \exp(u_{jl})}, \tag{1}$$

where $\mathbf{u}_j$ is a parameter vector that represents an unnormalized probability. In other words, each leaf learns fixed knowledge as a result of training based on the examples that are passed to that leaf with high arrival probabilities.

The parameters in all internal and leaf nodes are learned by a gradient-based approach for minimizing the following loss function:

$$l_M(\mathbf{x}, y) = \sum_{j \in \mathcal{N}_d} r_j(\mathbf{x}) l_{\text{cls}}(\mathbf{u}_j, y), \tag{2}$$

where $\mathcal{N}_d$ is the set of all leaf nodes, and $l_{\text{cls}}(\mathbf{u}_j, y)$ is a loss function that measures the difference between the prediction at node j and the true label y.

In the inference phase, GSDT chooses the path that leads to the leaf node j having the maximum arrival probability $r_j(\mathbf{x})$ and returns the distribution $\mathbf{q}_j$ it has learned during the training. Interpreting the single most probable path is more straightforward than interpreting all possible paths at each prediction. The complexity of the inference is also reduced from $O(b^d)$ (considering all branches) to $O(d)$. This is the main difference from HGMM [5, 15, 19] and ensemble models [12] that involve all experts in a tree or a forest at every prediction to boost the performance, making the decision processes not interpretable.

3.2 Gaussian Decisions

The main characteristic of GSDT is the modeling of decisions as Gaussian mixtures. Each node i models its child j as a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ are learnable mean and covariance, respectively. It then computes the likelihood of $\mathbf{x}$ being sampled from the distribution $\mathcal{N}_j$ of each child j and passes $\mathbf{x}$ to the next layer following the computed likelihoods.

In other words, the probability f_{ij} of passing $\mathbf{x}$ to node j from node i is

$$f_{ij}(\mathbf{x}) = \frac{\exp(\mathcal{L}(\theta_j \mid \mathbf{x}))}{\sum_k \exp(\mathcal{L}(\theta_k \mid \mathbf{x}))}, \tag{3}$$

where $\mathcal{L}(\theta_j \mid \mathbf{x})$ is the log likelihood of $\mathbf{x}$ being generated from $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, which is defined as follows:

$$\mathcal{L}(\theta_j \mid \mathbf{x}) = -\frac{1}{2}\left((\mathbf{x} - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) + \log\det(\boldsymbol{\Sigma}_j) + d\log(2\pi)\right). \tag{4}$$

However, it is computationally expensive to learn the full covariance matrix $\boldsymbol{\Sigma}_j$ for all nodes due to the inverse and determinant operations in Equation (4), as $\mathbf{\Sigma}_j$ is a $m \times m$ matrix where m is the number of features. Thus, we introduce two simpler structures for learning the covariance matrices.

Diagonal covariance. A naive approach is to assume a diagonal covariance for every node and determine its elements by a vector $\boldsymbol{\sigma}_j$ such that $\sigma_{jt} = \mathit{\Sigma}_{jtt}$ for all t. Since $\boldsymbol{\sigma}_j$ should contain only positive values, we introduce a free parameter $\bar{\boldsymbol{\sigma}}_j$ and apply the softplus function [4] as follows:

$$\mathbf{\Sigma}_j^{(\text{diagonal})} = \text{diag}(\boldsymbol{\sigma}_j), \tag{5}$$

where $\boldsymbol{\sigma}_j = \log(1+\exp(\bar{\boldsymbol{\sigma}}_j))$, and $\text{diag}(\cdot)$ makes a diagonal matrix from a vector. This approach is the simplest but neglects the correlations between features.

Diagonal covariance with low-rank perturbations. A more principled approach is to generalize the diagonal covariances by adding low-rank perturbations [17] with a small number of parameters by the choice of a rank k:

$$\mathbf{\Sigma}_j^{(\text{perturbed})} = \text{diag}(\boldsymbol{\sigma}_j) + \mathbf{U}\mathbf{U}^\top, \tag{6}$$

where $\mathbf{U} \in \mathbb{R}^{m \times k}$ is a rectangular matrix learned as a free parameter, k is given as a hyperparameter, and $\boldsymbol{\sigma}_j$ is the same as in Equation (5). It efficiently makes the covariance matrix have non-diagonal entries for feature correlations only by the additional mk parameters included in the $\mathbf{U}$ matrix.

$\log\det(\mathbf{\Sigma}_j)$ and the inverse $\mathbf{\Sigma}_j^{-1}$ are computed efficiently thanks to the matrix determinant lemma and the Woodbury matrix identity, respectively [7]:

$$\log\det(\mathbf{\Sigma}_j) = \log\det(\mathbf{I}_m + \mathbf{U}^\top\mathbf{A}^{-1}\mathbf{U}) + \log\det(\mathbf{A}), \tag{7}$$

$$\mathbf{\Sigma}_j^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{I}_k + \mathbf{U}^\top\mathbf{A}^{-1}\mathbf{U})^{-1}\mathbf{U}^\top\mathbf{A}^{-1}, \tag{8}$$

where $\mathbf{A} = \text{diag}(\boldsymbol{\sigma}_j)$, and $\mathbf{I}_m \in \mathbb{R}^{m \times m}$ and $\mathbf{I}_k \in \mathbb{R}^{k \times k}$ are identity matrices with different sizes. Equations (7) and (8) are easily differentiable with respect to both $\mathbf{A}$ and $\mathbf{U}$, allowing the updates of parameters in gradient-based optimization.

3.3 Training with Path Regularization

The training process of GSDT consists as three parts: parameter initialization, a loss function for each leaf node, and regularization for better performance.

Initialization. We initialize the leaf and internal nodes with different strategies considering the property of GSDT. For the leaf nodes, we randomly initialize the logit $\mathbf{u}_i$ of every node i following the standard normal distribution $\mathcal{N}(0,1)$. For the internal nodes, we set the Gaussian mean $\boldsymbol{\mu}_i$ of every node i to zero to allow examples to be equally distributed to all leaf nodes at the early iterations of training. This allows the predictions of leaf nodes to have a sufficient variance needed to guide the training of internal nodes while minimizing the randomness of internal nodes whose parameters should be tuned carefully.

Loss function. We use the hinge loss [3] as the function l_{cls} of Equation (2), which is typically used by maximum-margin classifers, as follows:

$$l_{\text{cls}}(\mathbf{u}_j, y) = \sum_{k \in \mathcal{Y} \setminus \{y\}} \max(0, 1 + u_{jk} - u_{jy}), \tag{9}$$

Algorithm 1: Post-optimization of the leaf Gaussians of GSDT.

```
Input: A trained GSDT $M$, a set $\mathcal{D}$ of training features, a learning rate $\alpha$ for
    the covariances, and the number $n$ of iterations
1: for leaf node $j$ in $M$ do
2:     $\mathcal{X}_j \leftarrow \{\mathbf{x} \in \mathcal{D} \mid \arg\max_k r_k(\mathbf{x}) = j\}$
3:     $\boldsymbol{\mu}_j \leftarrow \sum_{\mathbf{x} \in \mathcal{X}_j} \mathbf{x}$
4:     for $i \in [1, n]$ do
5:         $l \leftarrow \mathrm{sum}((\boldsymbol{\Sigma}_j - \mathrm{cov}(\mathcal{X}_j))^2)$
6:         $\boldsymbol{\Sigma}_j \leftarrow \boldsymbol{\Sigma}_j - \alpha \cdot \partial l / \partial \boldsymbol{\Sigma}_j$
7:     end for
8: end for
9: Fine-tune the whole parameters of $M$ for a fixed number of epochs
```

where $\mathcal{Y}$ is the set of labels. The hinge loss gives a zero if $u_{jk} + 1 < u_{jy}$. In other words, it maximizes the score u_{jy} for the target class y, but stops the training if it reaches a reasonably good performance. Unlike the cross entropy loss [8], the hinge loss improves the robustness by allowing GSDT to focus on learning the leaf nodes whose predictions are inaccurate.

Path regularization. We propose to add *path regularization* to encourage GSDT to utilize more leaf nodes instead of a few dominant ones. The regularizer measures the negative entropy of the arrival probability vector $\mathbf{r}(\mathcal{B})$ as

$$l_{\mathrm{lr}}(\mathcal{B}) = \sum_{j \in \mathcal{N}_d} r_j(\mathcal{B}) \log r_j(\mathcal{B}) \quad \text{where} \quad \mathbf{r}(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x} \in \mathcal{B}} \mathbf{r}(\mathbf{x}), \tag{10}$$

where $\mathbf{r}(\mathbf{x})$ is the vector representation of arrival probabilities of $\mathbf{x}$, and $\mathcal{B}$ is a training batch. The regularizer $l_{\mathrm{lr}}(\mathcal{B})$ forces GSDT to distribute the examples in each batch equally to all leaves to minimize the negative entropy. We add $l_{\mathrm{lr}}(\mathcal{B})$ to the overall objective function of GSDT with a regularization strength λ.

Post-optimization of leaf nodes. GSDT uses gradient-based optimization to minimize the objective function, instead of the EM algorithm commonly used with the Gaussian mixture models. To accommodate for possible weaknesses in the gradient-based approach, we apply additional post-optimization at each leaf as described in Algorithm 1. This step allows the leaf Gaussians to be closer to the examples that they represent, with respect to both mean and covariance. All parameters of GSDT are fine-tuned according to the change of leaf nodes.

4 Theoretical Analysis

We compare our GSDT with previous tree models, especially soft decision trees (SDT) that adopt the binary structure with linear decisions, with respect to the multi-branched structure and the nonlinearity of decisions.

Multiple branches. Our Gaussian decisions make it possible to adopt multiple branches at each node without affecting the interpretability of the decision tree structure. This is because the learned distribution $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ of each node j is itself interpretable regardless of the other children in the same branch. Specifically, $\boldsymbol{\mu}_j$ summarizes the examples that pass through node j as an interpretable prototype, while $\boldsymbol{\Sigma}_j$ takes into account the different effect of each feature in the split; small σ_{jk} represents that the k-th feature is dominant in determining the score, as a small change of x_k can change the score greatly.

The main advantage of such multi-branched structures is that one can reduce the depth of a tree while maintaining a similar number of leaf nodes, improving the interpretability of individual decisions; a tree depth directly tells the number of decisions that need to be interpreted to explain each prediction. However, such generalization to multi-branched structures is not straightforward in linear tree models such as SDTs. Consider a linear decision function f_{ij} that is represented as a multinomial logistic classifier:

$$f_{ij}(\mathbf{x}) = \frac{\exp(\mathbf{x}^\top \mathbf{w}_{ij} + b_{ij})}{\sum_k \exp(\mathbf{x}^\top \mathbf{w}_{ik} + b_{ik})}, \tag{11}$$

where $\mathbf{w}_{ij}$ and b_{ij} are the weight and bias for path (i, j), respectively. This is a direct extension of SDTs into the multi-branched structure.

The main limitation of this approach is that the weight $\mathbf{w}_{ij}$ for each child j should be interpreted in relation to the other weights, unlike the binary version where a single weight $\mathbf{w}_i$ is a complete explanation for node i. In other words, a positive weight $\mathbf{w}_{ij} > \mathbf{0}$ does not guarantee the positive correlation between $\mathbf{x}$ and $f_{ij}(\mathbf{x})$, since the other weights in the branch can have more strong weights; what matters is the relative size compared to the other children in that branch. Thus, one needs to examine all weights in that branch for interpreting a decision, which significantly drops the interpretability of the model.

Number of parameters. GSDT has a similar number of parameters to a binary SDT assuming the same number of leaf nodes, when the rank k of low-rank Gaussian covariances is fixed as a small constant. An SDT has $O(n(m+y))$ parameters, where n is the number of leaf nodes, m is the number of features, and y is the number of classes, respectively. The number of parameters in GSDT is given formally as Lemma 1.

Lemma 1. *The number of parameters of* GSDT *is $O(n(mk+y))$, where m, n, and y are the numbers of features, leaf nodes, and classes, respectively, assuming that every node has the same number of children, and k is the rank of low-rank perturbations of Gaussian covariances.*

Proof. Let d be the depth of GSDT. Each internal node has $n^{1/d}$ children, and the overall number of branches in the tree is $\sum_{i=1}^{d} n^{i/d} = (n-1)/(1-n^{-1/d})$, which is $O(n)$. Since each branch involves $m(k+1)$ parameters in the mean and covariance, the number of parameters in all internal decisions is $O(nmk)$.

This shows that GSDT efficiently models the decision process by hierarchical Gaussian mixtures with a few additional parameters from SDTs. The rank k is set to 1 or 2 in our experiments, since small k is sufficient to model the relations between features for learning non-diagonal covariance matrices.

Table 1: Classification accuracy for feature-based classification. The best performances are in bold, and the second-best ones are underlined. GSDT shows the highest accuracies in five datasets compared with eight strong baselines.

Model	**Brain**	**Breast**	**Breast-wis**	**Diabetes**	**Heart**	**Hepatitis**
LR	63.4 ± 0.0	65.5 ± 0.0	97.1 ± 0.0	76.0 ± 0.0	**86.9 ± 0.0**	77.4 ± 0.0
SVM-lin	61.0 ± 0.0	62.1 ± 0.0	97.1 ± 0.0	**76.6 ± 0.0**	83.6 ± 0.0	77.4 ± 0.0
SVM-rbf	58.5 ± 0.0	70.7 ± 0.0	97.1 ± 0.0	76.0 ± 0.0	**86.9 ± 0.0**	77.4 ± 0.0
DT	70.5 ± 0.7	68.8 ± 1.6	96.0 ± 0.9	69.7 ± 1.6	67.2 ± 1.6	70.0 ± 6.9
SDT	66.8 ± 5.0	73.3 ± 5.2	97.9 ± 0.0	76.0 ± 0.7	80.7 ± 2.7	67.3 ± 4.7
EDiT	58.5 ± 0.0	75.0 ± 2.6	97.1 ± 0.2	74.6 ± 1.5	85.2 ± 2.3	77.8 ± 3.8
MLP	73.4 ± 1.7	73.3 ± 2.3	98.6 ± 0.2	75.0 ± 0.8	80.5 ± 1.5	64.2 ± 3.0
RF	68.0 ± 2.3	76.6 ± 0.8	98.1 ± 0.3	73.4 ± 0.7	84.8 ± 0.8	70.3 ± 2.4
GSDT	**73.5 ± 1.5**	**77.2 ± 1.7**	**98.8 ± 0.6**	76.0 ± 0.9	**86.9 ± 1.2**	**78.2 ± 3.1**

5 Experiments

We compare GSDT with baseline models for feature-based classification on six datasets. We also demonstrate the interpretability of GSDT in a case study and compare the different approaches for modeling Gaussian covariances.

5.1 Experimental Settings

Datasets. We use six public feature-based datasets that are generated from the bio and medical domains, where interpretability is a crucial factor. Brain-tumor[3] is used to find a brain tumor from the information of a patient. Breast-cancer[4] and Breast-cancer-wisconsin[5] are used to predict breast cancers from clinical cases. Diabetes[6] is used to predict the status of a patient from diabetes. Heart-disease[7] is used to find the presence of heart disease in a patient. Hepatitis[8] is used to predict whether a patient lives or dies from the hepatitis disease.

Baselines. We compare GSDT with baseline models that have been used widely for classification tasks. Our main competitors are models that provide direct interpretability. Logistic regression (LR), support vector machines (SVM), and decision trees (DT) make interpretable decisions but have low accuracy in overall [1]. We implement two kinds of SVMs with the linear and RBF kernels, respectively. Soft decision trees (SDT) [11] and EDiT [21] improve decision trees by adopting soft decisions at internal branches, but weaken the interpretability.

[3] https://www.kaggle.com/pranavraikokte/braintumorfeaturesextracted
[4] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer
[5] https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original)
[6] https://www.kaggle.com/uciml/pima-indians-diabetes-database
[7] https://archive.ics.uci.edu/ml/datasets/Heart+Disease
[8] https://archive.ics.uci.edu/ml/datasets/Hepatitis

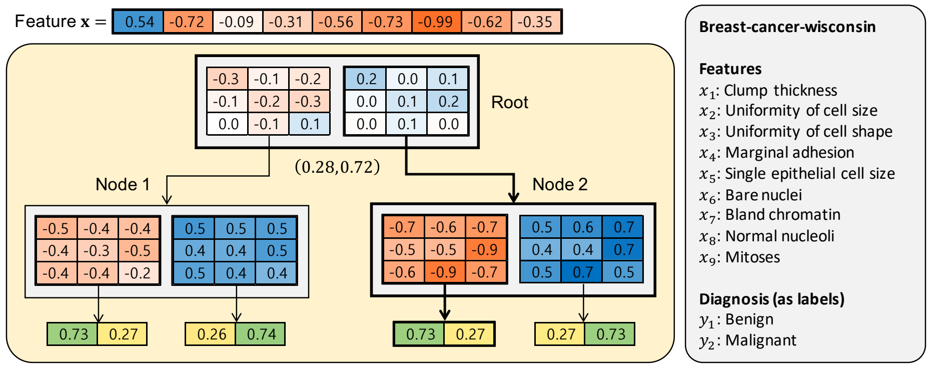


Fig. 2: The structure of simple GSDT trained for the Breast-cancer-wisconsin dataset having nine features and two labels. Each node is a Gaussian mixture that contains two Gaussian distributions with separate mean and covariance, but we represent only the mean vectors for simplicity. GSDT passes the example **x** to the third leaf through two Gaussian mixtures, classifying it as *benign*.

We also consider popular black box models that are not interpretable such as random forests (RF) and multilayer perceptrons (MLP) for completeness.

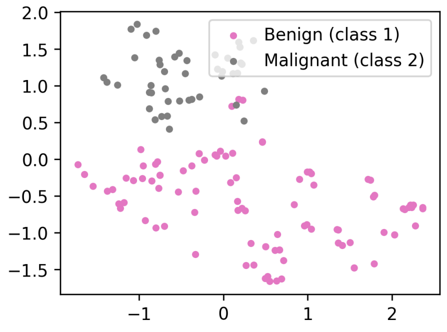
Implementation. We split each dataset randomly into training and testing by the 8:2 ratio. We run eight experiments for each model and report the average and standard deviation of classification accuracy on the test data. Some models have zero standard deviations as they are learned to find the global optima.

We use Scikit-learn implementations [20] of most baselines except SDTs and EDiT that we have implemented by PyTorch along with GSDT. We set the tree depth of SDTs and EDiT to 8 as in their original papers. On the other hand, we set the tree depth and the number of children of GSDT to 2 and 6, respectively. We set the strength λ of path regularization to 0.001 and the number n of post-optimization updates to 10. We set the rank k of Gaussian covariances to 1 or 2 based on the datasets. We use the Adam optimizer [14] for training.

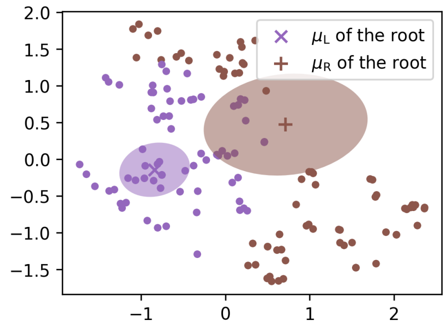
5.2 Classification Accuracy

Table 1 compares the accuracies of GSDT and the baselines on the six datasets. GSDT achieves at least the second-best accuracy in all datasets, outperforming all baselines and even the black box models by the average accuracy: the accuracy of GSDT is 4.1 and 5.5 points higher than that of the RF and MLP, respectively. This shows the effectiveness of GSDT for feature-based classification, which can avoid overfitting by our regularized training while having enough representation power for learning complex decision rules even with a few tree layers.

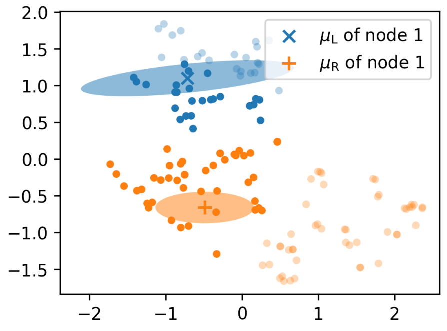
We compare the SDT-based models from the result: SDT, EDiT, and GSDT. The accuracy of EDiT is similar to that of SDTs, as it focuses on improving the interpretability of SDTs rather than its representation power; an SDT is better at Brain-tumor, while EDiT is better at Heart-disease and Hepatitis. The core



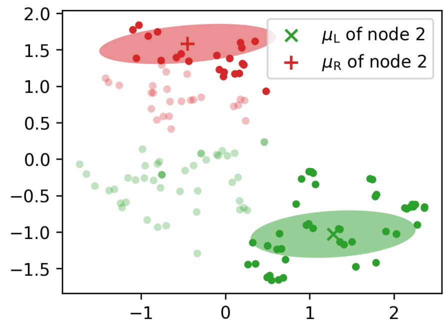
(a) Classes of test examples.



(b) Decision by the root.



(c) Decision by node 1.



(d) Decision by node 2.

Fig. 3: Gaussian distributions learned by GSDT for the Breast-cancer-wisconsin dataset. The test examples are divided first by the root and then by the internal nodes, based on the likelihoods of Gaussian distributions. The ellipses represent the covariance matrices, and the blurry points represent examples that reach at each node only at the training time, not at the inference time.

structure of EDiT is the same as SDTs, and thus it shares the same limitations that we aim to address in this work. GSDT achieves the highest accuracy among the three models, effectively improving the performance of SDTs.

5.3 Interpretability

Figure 2 shows the learned structure of GSDT of depth two, having two children at each branch, trained for the Breast-cancer-wisconsin dataset. The dataset has nine features that represent the cell characteristics of tissue images extracted from breast cancer patients to classify them into benign or malignant. The root node softly passes $\mathbf{x}$ to the right child by the probability of 72%, and the arrived node passes $\mathbf{x}$ again to the left leaf node, classifying it as benign. Since examples take a single path during inference, which is represented by a Gaussian node, it is straightforward to interpret both the structure and decisions.

Figure 3 illustrates the learned distributions and decisions of GSDT as 2D scatter plots. The same Breast-cancer-wisconsin dataset is used, but we run the t-SNE algorithm [16] before the training for clear visualization. Figure 3a shows test examples that are categorized into two classes. The root node first divides the examples into two clusters based on the Gaussian likelihoods in Figure 3b. The distribution of the right child of the root has the largest covariance in the figures, reflecting the uncertainty of the decision. In Figures 3c and 3d, each of the internal nodes splits examples to the leaf nodes for the final prediction.

Each distribution reflects the property of a decision in its mean vector and covariance matrix. Specifically, the mean vector works as an interpretable prototype that summarizes the examples that pass through that node, which is itself interpretable regardless of the other nodes. The blue and red distributions (and the orange and green distributions) have similar roles at the different branches, which is to classify the examples as benign (and malignant).

5.4 Ablation Study

We compare various options of modeling the Gaussian covariance in Figure 1 by generating a synthetic dataset consisting of two-dimensional features. GSDT with the identity covariance works well in Figure 1a, but the centers of Gaussians are different from those of true data because the covariance cannot be changed during training; the distributions have moved due to the bias of data. The diagonal covariance in Figure 1b works also well, but the covariance matrices cannot reflect the long shape of true clusters. Our choice of the covariance, which is to combine the diagonal entries with the low-rank perturbations, reflect accurately the property of original data. Moreover, the diagonal covariance with low-rank perturbations is able to capture both positive and negative correlations as seen by the yellow and green clusters, respectively, without limitations.

6 Conclusion

In this work, we have proposed Gaussian Soft Decision Trees (GSDT), a novel tree-structured classifier that models the internal nodes as Gaussian mixtures. Each edge in GSDT represents a multivariate Gaussian distribution parameterized by the learnable mean and covariance, which summarizes the examples that pass through it as an interpretable prototype. This makes it possible for GSDT to adopt a multi-branched structure where each edge is learned and interpreted independently of the other edges in the branch. Our experiments on six feature-based datasets show that GSDT achieves at least 4.8% higher average accuracy than models based on soft decision tree (SDT), while having a depth 4× smaller than that of SDTs. We also visualize the learned structure and decision process of GSDT to demonstrate its interpretability on an actual feature-based dataset of the biomedical domain as a case study.

Acknowledgments. Publication of this article has been funded by the Basic Science Research Program through the National Research Foundation of Korea (2018R1A1A3A0407953, 2018R1A5A1060031).

References

1. Bishop, C.M.: Pattern recognition and machine learning. springer (2006)
2. Chang, Y., Hsieh, C., Chang, K., Ringgaard, M., Lin, C.: Training and testing low-degree polynomial data mappings via linear SVM. J. Mach. Learn. Res. **11**, 1471–1490 (2010)
3. Dogan, Ü., Glasmachers, T., Igel, C.: A unified view on multi-class support vector classification. J. Mach. Learn. Res. **17**, 45:1–45:32 (2016)
4. Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., Garcia, R.: Incorporating second-order functional knowledge for better option pricing. In: NIPS. pp. 472–478. MIT Press (2000)
5. Eckart, B., Kim, K., Kautz, J.: HGMR: hierarchical gaussian mixtures for adaptive 3d registration. In: ECCV. vol. 11219, pp. 730–746. Springer (2018)
6. Frosst, N., Hinton, G.E.: Distilling a neural network into a soft decision tree. In: AI*IA. CEUR Workshop Proceedings, vol. 2071. CEUR-WS.org (2017)
7. Harville, D.A.: Matrix algebra from a statistician's perspective (1998)
8. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. CoRR **abs/1503.02531** (2015)
9. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. The annals of statistics pp. 1171–1220 (2008)
10. Irsoy, O., Alpaydin, E.: Autoencoder trees. In: ACML. vol. 45, pp. 378–390 (2015)
11. Irsoy, O., Yildiz, O.T., Alpaydin, E.: Soft decision trees. In: ICPR (2012)
12. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Neural Comput. **6**(2), 181–214 (1994)
13. Kim, B., Khanna, R., Koyejo, O.: Examples are not enough, learn to criticize! criticism for interpretability. In: NIPS (2016)
14. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
15. Liu, M., Chang, E., Dai, B.q.: Hierarchical gaussian mixture model for speaker verification. In: Seventh International Conference on Spoken Language Processing (2002)
16. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
17. Magdon-Ismail, M., Purnell, J.T.: Approximating the covariance matrix of gmms with low-rank perturbations. Int. J. Data Min. Model. Manag. **4**(2), 107–122 (2012)
18. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence **267** (2018)
19. Olech, L.P., Paradowski, M.: Hierarchical gaussian mixture model with objects attached to terminal and non-terminal dendrogram nodes. In: CORES (2015)
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
21. Yoo, J., Sael, L.: Edit: Interpreting ensemble models via compact soft decision trees. In: ICDM. pp. 1438–1443 (2019)
22. Zhu, J., Hastie, T.: Kernel logistic regression and the import vector machine. In: NIPS. pp. 1081–1088 (2001)