Interpretable Clustering via Soft Clustering Trees

Eldan Cohen(✉)

University of Toronto, Toronto, Canada
ecohen@mie.utoronto.ca

Abstract. Clustering is a popular unsupervised learning task that consists of finding a partition of the data points that groups similar points together. Despite its popularity, most state-of-the-art algorithms do not provide any explanation of the obtained partition, making it hard to interpret. In recent years, several works have considered using decision trees to construct clusters that are inherently interpretable. However, these approaches do not scale to large datasets, do not account for uncertainty in results, and do not support advanced clustering objectives such as spectral clustering. In this work, we present soft clustering trees, an interpretable clustering approach that is based on soft decision trees that provide probabilistic cluster membership. We model soft clustering trees as continuous optimization problem that is amenable to efficient optimization techniques. Our approach is designed to output highly sparse decision trees to increase interpretability and to support tree-based *spectral* clustering. Extensive experiments show that our approach can produce clustering trees of significantly higher quality compared to the state-of-the-art and scale to large datasets.

1 Introduction

Clustering, an unsupervised learning task, typically consists of partitioning an unlabelled dataset into K groups of similar data points. Since most popular clustering algorithms do not provide any explanation or interpretation for the obtained partition, a post-hoc analysis is often required to characterize the groups. In recent years, different approaches for *interpretable* clustering aim to provide clustering together with explanations of the obtained groups. One of the most prominent directions for interpretable clustering is based on using decision trees to construct clusters [2,16,28,30]. However, existing approaches for clustering trees are not scalable to large datasets, do not account for uncertainty in results, and do not support advanced clustering objectives such as Spectral Clustering [44] and Kernel-PCA clustering [37].

Soft decision trees are decision trees where at each node a data point is directed left with some probability p and right with probability $1-p$. Soft decision trees have been used for classification and regression in a range of previous works [4,5,7,23,48]. Unlike *hard* decision trees that are typically optimized

A. A. Cire (Ed.): CPAIOR 2023, LNCS 13884, pp. 281–298, 2023.
https://doi.org/10.1007/978-3-031-33271-5_19

using a specialized heuristic procedure, soft decision trees can be optimized using gradient-based continuous optimization techniques. However, soft decision trees have not been applied to clustering.

In this work we present soft clustering trees, the first approach for interpretable clustering via soft decision trees that provide probabilistic output on cluster membership. Our approach is scalable and supports advanced clustering objectives such as Spectral Clustering and Kernel-PCA clustering. Specifically, we make the following contributions:

1. We present a novel approach for interpretable clustering based on soft decision trees that provide probabilistic output on cluster membership.
2. We present a continuous optimization model for soft clustering trees that is designed to produce *fully sparse* trees and is amenable to efficient second-order continuious optimization algorithms as well as scalable, SGD-based optimization algorithms.
3. We extend our soft clustering trees model to support *spectral* and *Kernel-PCA* clustering objectives, while still using interpretable decision trees in the original feature space to construct clusters.
4. We run extensive experiments and show that: (1) our spectral clustering and Kernel-PCA clustering variants can significantly outperform the state-of-the-art clustering trees algorithm on small and medium datasets; (2) our scalable approach for training soft clustering trees can produce high-quality clustering trees for large datasets.

2 Soft Clustering Trees

2.1 Soft Decision Trees

A *tree* $\mathcal{T}$ is a tuple $(\mathcal{T}_B, \mathcal{T}_L, \delta, p, l, r)$ where $\mathcal{T}_B$ is the set of branching nodes and $\mathcal{T}_L$ is the set of leaf nodes. $\delta \in \mathcal{T}_B$ is the root node, $p : (\mathcal{T}_B \cup \mathcal{T}_L - \{\delta\}) \rightarrow \mathcal{T}_B$ is the parent function, and $l, r : \mathcal{T}_B \rightarrow (\mathcal{T}_B \cup \mathcal{T}_L)$ are the left and right child functions, respectively.

The *depth* of a node in the tree $t \in \mathcal{T}_B \cup \mathcal{T}_L$ is recursively defined as $depth(t) = depth(p(t)) + 1$ with $depth(\delta) = 0$. The depth of a tree $\mathcal{T}$ is defined as the maximum depth among its leaf nodes, $depth(\mathcal{T}) = \max_{t \in \mathcal{T}_L} depth(t)$. A tree is considered *complete* if all leaves have the same depth, $\forall t_1, t_2 \in \mathcal{T}_L : depth(t_1) = depth(t_2)$.

A *decision tree* maps each branching node $t \in \mathcal{T}_B$ with a feature $f_t \in F$ and a threshold value μ_t such that each data point $x_i \in \mathbb{R}^{|F|}$ is directed left if $x_i^{f_t} \leq \mu_t$. An oblique decision tree maps each branching node to an oblique cut $a_{\cdot t}^T x_i - \mu_t$ such that x_i is directed left if $a_{\cdot t}^T x_i \leq \mu_t$. In contrast, a *soft decision tree* is associated with a matrix $a \in \mathbb{R}^{|F| \times |\mathcal{T}_B|}$ and a vector $\mu \in \mathbb{R}^{|\mathcal{T}_B|}$ such that the *probability* of point x_i to be directed left at branching node $t \in \mathcal{T}_B$ is

$$P_{it} = Sigmoid(\Gamma_t \cdot (a_{\cdot t}^T x_i - \mu_t)), \tag{1}$$

where $a_{\cdot t}$ is the column vector representing the coefficients of all features in branching node t, μ_t is the threshold value, and Γ_t controls the softness of the split at node t such that higher values leads to more deterministic decisions [4,27]. Therefore, P_{it} can be considered as a soft (probabilistic) version of the oblique cut $a_{\cdot t}^T x_i \leq \mu_t$. Note that the complement, $1 - P_{it}$, is the probability that point x_i is directed right at node $t \in \mathcal{T}_B$.

Finally, the probability that a data point x_i ends up at a leaf node $t \in \mathcal{T}_L$ is defined as

$$Q_{it} = \prod_{t' \in A_L(t)} P_{it'} \prod_{t' \in A_R(t)} (1 - P_{it'}), \tag{2}$$

where $A_L(t)$ (resp. $A_R(t)$) denotes the set of all ancestors of a leaf node $t \in \mathcal{T}_L$ such that t is a descendant of their left (resp. right) branch.[1]

2.2 Soft Clustering Trees

Let $X = \{x_i\}_{i=1}^n$ be a set of n data points with x_i being a finite-sized feature vector, $x_i \in \mathbb{R}^{|F|}$, and K be the number of clusters ($K<|X|$). To extend soft decision trees to perform clustering of X into K clusters, we consider the following objective function inspired by fuzzy clustering,

$$\min \sum_{i \in |X|} \sum_{k \in 1..K} w_{ik}^m \cdot \|x_i - z_k\|^2, \tag{3}$$

where w_{ik} (defined below) is the probability that data point x_i is in cluster $k \in 1..K$, z_k is the centroid of cluster k, $\|x_i - z_k\| = \sqrt{\sum_{f \in F} (x_i^f - z_k^f)^2}$ is the Euclidean distance between point x_i and the centroid z_k, and $m \geq 1.0$ is a hyperparameter that controls the fuzziness of the clustering. Equation (3) is similar to the objective of Fuzzy C-Means (FCM) [3], however in our case w is defined based on our soft decision tree rather than being an unconstrained variable.

To define w_{ik}, we first define $c_{\cdot t}$ to represent the distribution over cluster labels, i.e., c_{kt} is the probability that data points reaching leaf node $t \in \mathcal{T}_L$ are in cluster k. Then, we define w_{ik}, the probability that point x_i is in cluster k as:

$$w_{ik} = \sum_{t \in \mathcal{T}_L} Q_{it} c_{kt}. \tag{4}$$

In Fig. 1, we present an example for a soft clustering tree of depth 2 for the Iris dataset.

2.3 Sparsity in Soft Clustering Trees

While the soft clustering trees in Sect. 2.2 provide inherent tree-based interpretation, oblique cuts can be hard to interpret as they may utilize many, or even all

[1] If $A_L(t) = \varnothing$ or $A_R(t) = \varnothing$ then the corresponding products in Eq. (2) are equal to 1.0.

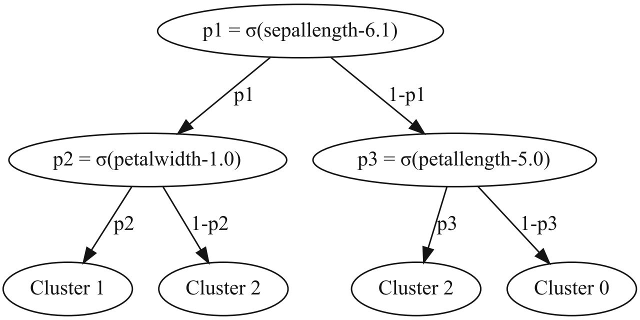

Fig. 1. Example soft clustering tree for the Iris dataset. We use σ to denote the Sigmoid function.

the features (i.e., cuts may have non-zero coefficients for many, or all, features). To obtain more interpretable trees, we would like to keep the number of non-zero coefficients in each branching node to a small number, ideally having only one non-zero coefficient similar to standard (non-oblique) decision trees. Previous works [4,21,27] on classification and regression have considered penalizing the ℓ_1-norm of the coefficient matrix a. However, this often results in some branching nodes having no non-zero coefficients while others having many non-zero coefficients, remaining difficult to interpret.

Instead of using oblique cuts, we consider the more restricted class of normalized cuts for branching nodes, i.e., all coefficients are non-negative ($\forall f \in F, t \in \mathcal{T}_B : a_{ft} > 0$) and the sum of coefficients in each branching node is equal to one ($\forall t \in \mathcal{T}_B : \sum_{f \in F} a_{ft} = 1$). This can be seen as a continuous relaxation of the typical univariate splits in standard decision trees by replacing the domain of each coefficient from the discrete set $\{0, 1\}$ to the continuous range $[0, 1]$ while keeping the sum of coefficients equal to one. Then, we introduce the following regularization term for each branching node $t \in \mathcal{T}_B$,

$$\phi_t = -\sum_{f \in F} a_{ft}^2. \tag{5}$$

The minimal value for each ϕ_t term is -1 which indicates a *fully sparse* cut, i.e., exactly one coefficient in the cut is equal to one and the rest are equal to zero.

2.4 Learning Sparse Soft Clustering Trees Using Continuous Optimization

We formulate the problem of learning soft clustering trees as a constrained continuous optimization problem. Given a dataset X and the number of clusters K, we search for an assignment of the variables a_{ft}, c_{kt}, μ_t, z_k, and Γ_t that minimizes our regularized clustering cost function.[2]

[2] Following [27], we keep Γ_t as variables rather than hyper-parameters.

In our continuous optimization model, we assume w.l.o.g that X is normalized in the range $[0, 1]$. We therefore bound the μ_t and z_k variables in the range $[0, 1]$. Further, to improve the optimization performance we consider $\mathbf{x}_i \in \mathbf{X}$ to be a transformation of the dataset $x_i \in X$ such that *each feature* is normalized within the range $[0, 1]$. In particular, we employed the quantile transformation following [27]. We redefine the cut in branching node $t \in \mathcal{T}_B$ to be $a_{\cdot t}^T \mathbf{x}_i - \mu_t$, while keeping the clustering objective based on the original $x \in X$. As we focus on fully sparse trees, each cut can be converted back to its original values using the inverse transformation.

Constrained Continuous Optimization Model. Equation (6) presents the complete constrained optimization model for sparse soft clustering trees. Equation (6a) is the objective function that consists of the clustering cost and the sparsity regularization weighted by hyper-parameter Λ. Equations (6b)–(6e) are based on the definitions discussed in Sects. 2.1, 2.2 and 2.3. The constraints in Eq. (6f) and Eq. (6g) guarantee that label probabilities in leaf nodes and feature coefficients in branch nodes sum to one, respectively. Finally, Eqs. (6h)–(6l) define the bounds for each of the continuous decision variables.

$$\min \sum_{i \in |X|} \sum_{k \in 1..K} w_{ik}^m \cdot \|x_i - z_k\|^2 + \Lambda \sum_{t \in \mathcal{T}_B} \phi_t \tag{6a}$$

$$s.t.:$$

$$P_{it} := Sigmoid(\Gamma_t \cdot (a_{\cdot t}^T \mathbf{x}_i - \mu_t)) \quad \forall \mathbf{x}_i \in \mathbf{X}, t \in \mathcal{T}_B \tag{6b}$$

$$Q_{it} := \prod_{t' \in A_L(t)} P_{it'} \prod_{t' \in A_R(t)} (1 - P_{it'}) \quad \forall x_i \in X, t \in \mathcal{T}_L \tag{6c}$$

$$w_{ik} := \sum_{t \in \mathcal{T}_L} Q_{it} c_{kt} \quad \forall x_i \in X, k \in 1..K \tag{6d}$$

$$\phi_t := -\sum_{f \in F} a_{ft}^2 \quad \forall t \in \mathcal{T}_B \tag{6e}$$

$$\sum_{k \in 1..K} c_{kt} = 1 \quad \forall t \in \mathcal{T}_L \tag{6f}$$

$$\sum_{f \in F} a_{ft} = 1 \quad \forall t \in \mathcal{T}_B \tag{6g}$$

$$0 \le a_{ft} \le 1 \quad \forall t \in \mathcal{T}_B, f \in F \tag{6h}$$

$$0 \le c_{kt} \le 1 \quad \forall t \in \mathcal{T}_L, k \in 1..K \tag{6i}$$

$$0 \le \mu_t \le 1 \quad \forall t \in \mathcal{T}_B \tag{6j}$$

$$0 \le z_k \le 1 \quad \forall k \in 1..K \tag{6k}$$

$$-1 \le \Gamma_t \le -128 \quad \forall t \in \mathcal{T}_B \tag{6l}$$

Constrained continuous optimization algorithms such as interior point optimization can be used to solve the optimization model in Eq. (6) for small and medium datasets, however they tend to be less efficient compared to unconstrained continuous optimization algorithms and are not amenable to scalable, mini-batch, stochastic gradient descent optimizers.

Unconstrained Optimization Model. We can reformulate the model in Eq. (6) to be an unconstrained optimization model by making the following changes.

Regularized Softmax Splits. To eliminate the constraints in Eqs. (6g)–(6h), we redefine a_{ft} based on Softmax normalization of the unnormalized variables $\hat{a}_{ft} \in \mathbb{R}^{|F|\times|\mathcal{T}_B|}$,

$$a_{ft} = \frac{\exp(\hat{a}_{ft})}{\sum_{f'\in F}\exp(\hat{a}_{f't}))}. \tag{7}$$

Similar to our constrained model, we use the regularization terms ϕ_t to guarantee sparse cuts. Due to the nature of Softmax, coefficients cannot be exactly zero, but could get very close to zero. However, since all features are scaled to the same range of [0, 1], features with near-zero coefficients will have negligible impact on the branching behavior and can be eliminated from the resultant clustering tree.[3] Although recent works have proposed several sparse variants of Softmax [10,29], we found that they can have negative impact on the optimization and are not needed in our case due to the feature-wise normalization.

Leaf Class Labels. To eliminate the constraints in Eq. (6f) and Eq. (6i), we redefine c_{kt} based on Softmax normalization of the unnormalized variables $\hat{c}_{kt} \in \mathbb{R}^{K\times|\mathcal{T}_L|}$,

$$c_{kt} = \frac{\exp(\hat{c}_{kt})}{\sum_{k'\in 1..K}\exp(\hat{c}_{k't}))}. \tag{8}$$

The bounds on the remaining variables, Eqs. (6j)–(6l), were used to improve the constrained model, but are not required and can be removed in our unconstrained model.

2.5 Interpretable Spectral and Kernel-PCA Clustering

One of the benefits of our formulation is that the feature representation used for constructing the decision tree and the feature representation used for the clustering do not have to be the same. Specifically, we can replace the objective in Eq. (3) with a more general objective,

$$\min \sum_{i\in|\bar{X}|}\sum_{k\in 1..K} w_{ik}^{m}\cdot\|\bar{x}_i - \bar{z}_k\|^2 \tag{9}$$

where $\bar{X} = \{\bar{x}_i\}_{i=1}^{n}$ is a (possibly) different representation of the dataset X based on feature set $\bar{F}$, $\bar{x}_i \in \mathbb{R}^{|\bar{F}|}$. We note that Eq. (3) is a special case of Eq. (9) with $\bar{F} = F$ and, consequently, $\bar{X} = X$. However, $\bar{X}$ can also be based on a different feature representation such as a spectral embedding or a PCA transformation of X. The decision tree is still constructed from the interpretable feature set X, i.e., branching node cuts based on $\mathbf{x} \in \mathbf{X}$, however the objective would be to optimize the clustering cost based on, for example, the spectral embedding or PCA transformation of the original dataset. In our experiment we consider two different objectives:

[3] For this purpose, we arbitrarily select 10^{-4} as the threshold for zeroing coefficients in the resultant clustering tree.

- Spectral clustering [44] where $\bar{X}$ is computed by applying spectral decomposition to the graph Laplacian of the k-nearest neighbors graph using the Laplacian Eigenmaps algorithm [1].
- Kernel-PCA (KPCA) clustering where $\bar{X}$ is computed using a non-linear dimensionality reduction through the use of kernels [37]. In our experiments, we use the radial basis function (RBF) kernel.

To our knowledge, this is the first approach for interpretable spectral and KPCA clustering that is based on decision trees in the original feature space.

2.6 Scalable Training of Soft Clustering Trees

Our unconstrained formulation in Sect. 2.4 is amenable for scalable training using mini-batch stochastic gradient descent algorithms to support interpretable clustering of large datasets using soft decision trees. However, consistent with work on soft classification trees [15], we found that training of soft clustering trees using first-order SGD optimizers can get stuck in poor solutions in which one or more of the branching nodes directs almost all the data points into one of the subtrees. We therefore introduce the following regularization term that encourages branching nodes to make equal use of both left and right branches, following [15],

$$\pi = - \sum_{t \in \mathcal{T}_B} \theta_t [0.5 \cdot \log(\alpha_t) + 0.5 \cdot \log(1 - \alpha_t)], \tag{10}$$

with α_t (resp. 1-α_t) being the fraction of probability mass directed to the left (resp. right) branch of branching node $t \in \mathcal{T}_B$ out of the probability mass directed to node t,

$$\alpha_t = \frac{\sum_{x_i \in X} Q_{i,l(t)}}{\sum_{x_i \in X} Q_{i,t}}, \tag{11}$$

and θ_t ensures the strength of the penalty decays exponentially with the depth of node t, $\theta_t = 2^{-depth(t)}$.

Training. The final objective function for our scalable training of soft clustering trees is

$$\min \sum_{i \in |X|} \sum_{k \in 1..K} w_{ik}^m \cdot \|x_i - z_k\|^2 + \Lambda \sum_{t \in \mathcal{T}_B} \phi_t + \omega\pi,$$

where ω is a hyper-parameter that controls the weight associated with the regularization term. To efficiently train clustering trees using mini-batch stochastic gradient descent algorithms, we start by training with no sparsity regularization, $\Lambda = 0$, for a fixed number of training steps. Then, we anneal Λ by increasing it every training step until we obtain a fully sparse tree.

3 Experiments

In this section, we perform extensive experiments with 18 datasets to evaluate the performance of soft clustering trees.

3.1 Implementation Details

Single-Batch Training of Soft Clustering Trees using Second-Order Optimizers. For our experiments with small and medium datasets, we implemented our constrained and unconstrained continuous optimization models for single-batch training using second-order optimizers. Our constrained optimization model was implemented in Julia using the JuMP library [13] and solved using IPOPT [45], a primal-dual interior-point algorithm with a filter line-search method for nonlinear programming. As IPOPT converges to a local minimum on non-convex problems, we run the solver five times, starting from different random initializations, and select the lowest-cost solution.

Our unconstrained model was implemented in Python and solved using the Limited-memory BFGS with bounds (L-BFGS-B) solver and the Sequential Least Squares Programming (SLSQP) solver, both implemented in the Scipy library. We found the runtime for unconstrained optimization to be shorter compared to constrained optimization, however it requires more runs to converge to high-quality solutions. We therefore restart the solver 20 times using random initializations and select the lowest-cost solution.

Scalable Training of Soft Clustering Trees using SGD. For our experiments with large datasets, we implemented our scalable model for training soft clustering trees (Sect. 2.6) in Python using the PyTorch library [31]. We employ minibatch stochastic optimization scheme using the RMSProp optimizer [20]. To obtain fully sparse trees, we train our model according to the training scheme described in Sect. 2.6: We first train the model for 25,000 steps with no sparsity regularization and then slowly anneal Λ by increasing it each step by 10^{-3}. In our experiments, we set the total number of training steps to be 50,000 and we employ cyclical learning rate schedule [39] in the range $[0.0005, 0.005]$.

Inference. Our decision trees are *soft* and represent probabilistic cluster membership. In our experiments, we obtain a *hard* clustering for each data point by selecting the cluster label for which the membership probability is the highest.

3.2 Datasets

To evaluate our single-batch approach for learning soft clustering trees, we use a set of 13 small- and medium-size real and synthetic datasets (Table 1). Seven real datasets were obtained from the UCI repository [12]: Glass, Ionosphere, Iris, TAE, Vertebral, Wine, Zoo. Four synthetic datasets representing challenging clustering problems in 2D and 3D were obtained from FCPS [43]. Finally, we generated two instances of the well-known clustering problems moons and circles.

Table 1. Details of small- and medium-size datasets used in the experiments. Synthetic datasets are marked "(s)".

Dataset	$\|X\|$	$\|F\|$	K
Atom (s)	800	3	2
Chainlink (s)	1,000	3	2
Circles (s)	500	2	2
Glass	214	9	6
Ionosphere	351	34	2
Iris	150	4	3
Moons (s)	500	2	2
TAE	151	5	3
Target (s)	770	2	6
Vertebral	310	6	3
Wine	178	12	3
Wingnut (s)	1,016	2	2
Zoo	101	16	7

Table 2. Details of large datasets used in the experiments.

Dataset	$\|X\|$	$\|F\|$	K
Adult[†]	48,842	105	2
Avila	20,867	10	12
Covertype	581,012	54	7
Pendigits	10,992	16	10
Shuttle	58,000	9	7

[†] Categorical features converted to one-hot encoding.

To evaluate our approach for scalable training of soft clustering trees, we run experiments on five large datasets obtained from UCI [12]: Adult, Avila, Covertype, Pendigits, and Shuttle, as described in Table 2. All datasets were standardized by removing the mean and scaling each feature to unit variance.

3.3 Evaluation

Since all datasets in Sect. 3.2 have ground-truth labels, we evaluate the quality of the obtained clusterings using the following external evaluation metrics. Note that we do not use internal evaluation metrics, such as the mean Silhouette Coefficient [36], as they depend on the feature representation and therefore are not comparable across different feature representations (such as the Spectral Embedding and the Kernel-PCA).

Adjusted Rand Index (ARI). Rand Index [35] measures agreement between two partitions of the same dataset, P_1 and P_2. Each partition represents $\binom{n}{2}$ decisions over all pairs, assigning them to the same or different clusters. Let a be the number of pairs assigned to the same cluster in both P_1 and P_2. Let b be the number of pairs assigned to different clusters. Rand Index is defined as follows:

$$RI(P_1, P_2) = \frac{a+b}{\binom{n}{2}}.$$

The Adjusted Rand Index (ARI) [22] is a correction for RI, based on its expected value:

$$ARI = \frac{RI - \mathbb{E}(RI)}{Max(RI) - \mathbb{E}(RI)}.$$

ARI score of zero indicates the partition is not better than a random assignment, while a score of one indicates a perfect match. We compute the ARI between the obtained clustering and the ground-truth labels.

Normalized Mutual Information (NMI). Mutual information quantifies the statistical information shared between two distributions [40]. $MI(P_1, P_2)$ denotes the mutual information between partitions P_1 and P_2, and $H(P_i)$ denotes the entropy of partition P_i. Normalized mutual information (NMI) [40] is normalized using the mean of $H(P_1)$ and $H(P_2)$:

$$NMI(P_1, P_2) = \frac{MI(P_1, P_2)}{Mean(H(P_1), H(P_2))}.$$

Values close to zero indicate independent partitions, while values close to one indicate a significant agreement between $P1$ and $P2$. We compute the NMI between the obtained clustering and the ground-truth labels.

Unsupervised Clustering Accuracy (ACC). The unsupervised clustering accuracy [46] is defined as:

$$ACC = \max_{map \in M} \frac{\sum_{i=1}^{n} \mathbb{1}\{l_i = map(c_i))\}}{n},$$

where $l(x_i)$ and $c(x_i)$ are the ground-truth label and the assigned cluster label for data point x_i, respectively, and M is the set of all possible one-to-one mappings from clusters to ground-truth labels.

3.4 Results

First, we compare our basic constrained and unconstrained optimization models to ExKMC [16], the state-of-the-art approach for interpretable clustering using decision trees. For our unconstrained model, we used both L-BFGS and SLSQP solvers. Each problem is solved 20 times starting from different initializations and the median runtime for one run was 1.81 s for L-BFGS and 2.01 s for SLSQP. As all runs are independent, they can be parallelized over multiple cores. As we are comparing our approaches that are probabilistic to the deterministic and fully sparse ExKMC that aims to optimize the standard K-means cost, we tuned the hyper-parameters of our approach over a small set of possible values, $\Lambda \in \{10^0, 10^1, 10^2\}$ and $m \in \{1.05, 1.1\}$, and select the ones that yielded the *hard* clustering with the lowest K-means cost while being fully sparse (all results presented for our approaches are therefore based on fully sparse trees). For our constrained model, solved using IPOPT, runs required longer runtime (median of 14.45 s) and we therefore opted for only five random initializations and considered only one value for m that was found to work well ($m = 1.0$). We note that the results for our approaches are not directly comparable in terms of optimization performance due to the large set of possible choices available for each solver (how many runs vs. how long each run, how many available

Table 3. Experimental Results on Soft Clustering Trees for Small and Medium Datasets.

dataset	Max	Adjusted Rand Index (ARI)				Clustering Accuracy (ACC)			
	$\lvert\mathcal{T}_L\rvert$	BFGS	SLSQ	IPOP	ExKM	BFGS	SLSQ	IPOP	ExKM
atom	4	0.180	0.182	0.161	**0.186**	0.713	0.714	0.701	**0.716**
atom	8	0.149	**0.165**	0.161	0.159	0.694	**0.704**	0.701	0.700
atom	16	**0.189**	0.176	0.174	**0.189**	**0.718**	0.710	0.709	**0.718**
chainl	4	−0.001	−0.001	**0.183**	−0.001	0.500	0.500	**0.714**	0.504
chainl	8	−0.001	−0.001	**0.207**	−0.001	0.509	0.505	**0.728**	0.508
chainl	16	−0.001	−0.000	**0.107**	−0.001	0.505	0.514	**0.664**	0.509
circles	4	**−0.002**	**−0.002**	**−0.002**	**−0.002**	**0.502**	**0.502**	**0.502**	**0.502**
circles	8	**−0.002**	**−0.002**	**−0.002**	**−0.002**	**0.502**	**0.502**	**0.502**	**0.502**
circles	16	**−0.002**	**−0.002**	**−0.002**	**−0.002**	0.500	**0.502**	**0.502**	**0.502**
glass	8	**0.200**	0.188	0.168	0.148	**0.505**	0.458	0.481	0.425
glass	16	0.148	0.173	**0.230**	0.173	0.481	0.472	**0.519**	0.472
iono	4	0.158	0.145	0.149	**0.163**	0.701	0.692	0.695	**0.704**
iono	8	0.112	0.178	**0.183**	0.168	0.670	0.712	**0.715**	0.707
iono	16	**0.193**	0.178	0.168	0.168	**0.721**	0.712	0.707	0.707
iris	4	**0.759**	0.610	0.515	0.574	**0.907**	0.827	0.747	0.800
iris	8	**0.653**	0.620	0.574	0.601	**0.853**	0.833	0.800	0.820
iris	16	**0.642**	0.632	**0.642**	0.610	**0.847**	0.840	**0.847**	0.827
moons	4	**0.483**	**0.483**	**0.483**	0.456	**0.848**	**0.848**	**0.848**	0.838
moons	8	**0.472**	**0.472**	**0.472**	0.461	**0.844**	**0.844**	**0.844**	0.840
moons	16	0.472	0.472	0.472	**0.478**	0.844	0.844	0.844	**0.846**
tae	4	**0.064**	**0.064**	0.050	0.047	**0.510**	**0.510**	0.444	0.417
tae	8	0.047	0.047	0.047	**0.064**	0.417	0.417	0.417	**0.510**
tae	16	0.048	0.047	0.048	**0.064**	0.424	0.417	0.424	**0.510**
target	8	0.529	0.557	0.302	**0.636**	0.635	0.627	0.416	**0.638**
target	16	0.636	**0.637**	0.634	0.636	0.642	**0.652**	0.625	0.636
vert	4	0.163	**0.212**	0.175	0.165	0.465	**0.487**	0.471	0.452
vert	8	0.180	**0.221**	0.169	0.194	0.461	**0.516**	0.455	0.461
vert	16	**0.221**	0.210	0.219	0.196	0.506	**0.513**	0.490	0.461
wine	4	0.754	0.748	**0.848**	0.802	0.916	0.910	**0.949**	0.933
wine	8	0.732	0.757	0.741	**0.880**	0.904	0.916	0.910	**0.961**
wine	16	0.683	0.835	0.880	**0.897**	0.882	0.944	0.961	**0.966**
wingn	4	0.760	0.791	0.791	**0.930**	0.936	0.945	0.945	**0.982**
wingn	8	0.736	0.733	0.743	**0.764**	0.929	0.928	0.931	**0.937**
wingn	16	0.700	0.693	**0.730**	0.683	0.918	0.916	**0.927**	0.913
zoo	8	0.870	**0.871**	0.617	0.737	**0.871**	**0.871**	0.762	0.822
zoo	16	0.814	**0.815**	0.792	0.737	0.822	0.812	**0.832**	0.822
average		0.354	0.359	0.356	**0.360**	0.683	0.684	**0.687**	0.682

cores, how many hyper-parameters values to consider, etc.) We therefore simply demonstrate the performance of each approach with a reasonable set of choices.

For ExKMC, we run the algorithm from 100 different random initializations and choose the one with the lowest cost (experiments with additional runs did not lead to significant improvement). As ExKMC does not have a limit on the tree depth but on the maximum number of leaves, we compare the results for three different values of number of leaves, namely $4, 16, 32$. In our approach these values correspond to a maximum tree depth of $2, 3, 4$. For each dataset, we set K to be the number of ground-truth labels in the dataset. As the datasets Glass, Target, and Zoo have more than 4 clusters, we only run experiments for a maximum number of leaves of 16 and 32.

Table 3 shows the ARI and ACC scores obtained by each of the approaches for each of the datasets. It also reports the average scores across all datasets. Results on NMI exhibited similar trends and are omitted due to space. We can see that, in general, the different methods are relatively comparable. Each of the methods outperforms the other methods on some of the datasets, and we observe minor differences between the methods in the average scores. Specifically, ExKMC performed slightly better in terms of average ARI and IPOPT performs slightly better in terms of ACC as well as NMI (not presented).

In the next two sections we demonstrate the unique benefits of our approaches, namely that they can be extended to use Spectral and K-PCA objectives and that are amenable to scalable optimization procedures.

Spectral and K-PCA Clustering Trees. We present results for the extensions of our basic approach: our Kernel PCA model (*Ours-K*), and our Spectral Clustering model (*Ours-S*). For KPCA, we used 10 components. For the spectral embedding, we used k-nearest neighbors graph with $k = 10$ for all datasets and set the dimension of the projected subspace to be the number of clusters. Due to limited space, in this experiment we focus on our unconstrained model as it is the basis for our scalable model, and we present results for the L-BFGS solver. We compare our approaches to our basic model (*Ours*) and to ExKMC [16].

Table 4 shows the ARI and ACC scores obtained by each of the approaches for each of the datasets. Results on NMI exhibited similar patterns to ARI and are omited due to space. It also reports the average scores across all datasets. The best performing approach based on the average scores is *Ours-S* followed by *Ours-K*. Furthermore, we observe that in approximately 86% of the cases, for all evaluation metrics (including NMI), the top performing approach is one of our approaches. The results demonstrate the unique benefits of approaches like *Ours-S* in cases such as the datasets Atom, Chainlink, and Circles, where ExKMC and *Ours* find low-quality solutions compared to the high-quality solutions found by *Ours-S* due to the spectral embedding.

Results for Large Datasets. Next, we run experiments with our approach for scalable training of soft clustering trees (Sect. 2.6). As our approach is the first scalable approach for interpretable clustering based on decision trees, we

Table 4. Experimental Results on Soft Clustering Trees for Small and Medium Datasets. Our approaches are based on our unconstrained model solved by L-BFGS.

dataset	Max	Adjusted Rand Index (ARI)				Clustering Accuracy (ACC)			
	$\lvert\mathcal{T}_L\rvert$	Ours	Ours-K	Ours-S	ExKM	Ours	Ours-K	Ours-S	ExKM
atom	4	0.180	0.577	**0.779**	0.186	0.713	0.880	**0.941**	0.716
atom	8	0.149	0.865	**0.874**	0.159	0.694	0.965	**0.968**	0.700
atom	16	0.189	0.912	**0.970**	0.189	0.718	0.978	**0.993**	0.718
chain	4	−0.001	0.174	**0.861**	−0.001	0.500	0.709	**0.964**	0.504
chain	8	−0.001	0.178	**0.933**	−0.001	0.509	0.711	**0.983**	0.508
chain	16	−0.001	0.181	**0.941**	−0.001	0.505	0.713	**0.985**	0.509
circles	4	−0.002	−0.002	**0.369**	−0.002	0.502	0.504	**0.804**	0.502
circles	8	−0.002	−0.002	**0.639**	−0.002	0.502	0.502	**0.900**	0.502
circles	16	−0.002	−0.002	**1.000**	−0.002	0.500	0.504	**1.000**	0.502
glass	8	**0.200**	0.145	0.112	0.148	**0.505**	0.411	0.360	0.425
glass	16	0.148	**0.184**	0.133	0.173	**0.481**	0.439	0.379	0.472
iono	4	0.158	**0.203**	−0.028	0.163	0.701	**0.726**	0.538	0.704
iono	8	0.112	**0.208**	−0.034	0.168	0.670	**0.729**	0.504	0.707
iono	16	0.193	**0.224**	−0.034	0.168	0.721	**0.738**	0.501	0.707
iris	4	**0.759**	0.600	0.489	0.574	**0.907**	0.820	0.773	0.800
iris	8	0.653	**0.736**	0.394	0.601	0.853	**0.900**	0.700	0.820
iris	16	**0.642**	0.611	0.413	0.610	**0.847**	0.827	0.713	0.827
moons	4	0.483	0.512	**0.678**	0.456	0.848	0.858	**0.912**	0.838
moons	8	0.472	0.512	**0.853**	0.461	0.844	0.858	**0.962**	0.840
moons	16	0.472	0.512	**1.000**	0.478	0.844	0.858	**1.000**	0.846
tae	4	**0.064**	**0.064**	0.050	0.047	**0.510**	**0.510**	0.444	0.417
tae	8	0.047	**0.113**	0.047	0.064	0.417	**0.550**	0.417	0.510
tae	16	0.048	**0.113**	0.047	0.064	0.424	**0.550**	0.417	0.510
target	8	0.529	0.538	0.544	**0.636**	0.635	0.627	0.626	**0.638**
target	16	**0.636**	0.634	0.328	**0.636**	**0.642**	0.627	0.443	0.636
vert	4	0.163	0.169	**0.171**	0.165	**0.465**	0.458	0.461	0.452
vert	8	0.180	**0.254**	0.212	0.194	0.461	0.474	**0.500**	0.461
vert	16	0.221	**0.251**	0.219	0.196	0.506	**0.539**	0.474	0.461
wine	4	0.754	0.723	0.762	**0.802**	0.916	0.899	0.916	**0.933**
wine	8	0.732	0.642	0.754	**0.880**	0.904	0.871	0.916	**0.961**
wine	16	0.683	0.725	0.820	**0.897**	0.882	0.904	0.938	**0.966**
wingn	4	0.760	0.693	**1.000**	0.930	0.936	0.916	**1.000**	0.982
wingn	8	0.736	0.651	**1.000**	0.764	0.929	0.904	**1.000**	0.937
wingn	16	0.700	0.736	**0.984**	0.683	0.918	0.929	**0.996**	0.913
zoo	8	**0.870**	0.820	0.653	0.737	**0.871**	0.832	0.743	0.822
zoo	16	**0.814**	0.646	0.633	0.737	**0.822**	0.743	0.752	**0.822**
average		0.354	0.419	**0.544**	0.360	0.683	0.721	**0.748**	0.682

compare our approach to *non-interpretable* scalable clustering using Mini-Batch K-Means [38].

We run experiments for three tree depths: the minimum depth based on the number of ground-truth labels, as well as two levels deeper. We did not tune hyper-parameters for each dataset and instead fix $\omega = 0.1$ and $m = 1.05$ across datasets (hyper-parameter tuning per dataset may lead to further improvement). We run the training procedure five times, starting from different random initializations, using a batch size of 256. Similar to previous experiment, we select the one that yielded the hard clustering with the lowest K-means cost while being fully sparse. For Mini-Batch K-Means, we run the algorithm for 100 random initializations with a similar batch size of 256 and select the lowest cost solution.

Table 5 compares our approach for scalable training (*Ours*) to Mini-Batch K-Means (mKM) on the five large datasets. We note that the two methods are *not* directly comparable as Mini-Batch K-Means is not constrained to produce tree-based clusterings. The results show that for Adult, Covertype, and Shuttle, our approach can reach comparable results to mKM and even find higher-quality solutions according to some criteria. For Pendigits, we observe that as we increase the tree depth we are getting closer to mKM's performance however even a depth of 6 was not sufficient to reach the performance of mKM with a *fully-sparse* decision tree. For Avila, we interestingly find the best solution at the lowest tree depth. Overall, the results in Table 5 indicate that our scalable approach is able to produce high-quality, fully sparse clustering trees for large datasets.

Table 5. Experimental Results for Large Datasets.

X	Max	ARI		NMI		ACC	
	$\lvert\mathcal{T}_L\rvert$	Ours	mKM	Ours	mKM	Ours	mKM
Adult	2	**0.184**	0.183	0.134	**0.136**	**0.719**	0.718
Adult	4	**0.184**		0.134		**0.719**	
Adult	8	0.180		0.133		0.717	
Avila	16	**0.064**	0.052	0.117	**0.136**	0.291	**0.292**
Avila	32	0.016		0.053		0.218	
Avila	64	0.055		0.108		0.232	
Cover	8	0.037	0.056	0.143	**0.150**	0.291	0.319
Cover	16	**0.057**		**0.150**		**0.329**	
Cover	32	0.031		0.145		0.309	
Pend.	16	0.403	**0.539**	0.554	**0.685**	0.590	**0.675**
Pend.	32	0.437		0.595		0.590	
Pend.	64	0.485		0.624		0.638	
Shut.	8	0.181	0.214	0.366	0.378	0.412	0.421
Shut.	16	0.196		0.329		0.444	
Shut.	32	**0.348**		**0.475**		**0.631**	

4 Related Work

Soft decision trees have been a popular choice in tasks such as classification and regression, solved using either constrained or unconstrained continuous optimization algorithms [4,5,23,27]. Some works have explored using soft decision trees together with learned representations by formulating the problem as a deep neural network [15,19,41,47]. To our knowledge, our work is the first approach that use soft decision trees for interpretable clustering.

Recent work on neural oblivious classification and regression trees has considered sparse alternatives of Softmax, such as entmax [33], to produce sparse trees [34], however we found it difficult to produce fully sparse trees without hurting the optimization performance.

Previous work on interpretable clustering primarily focused on using decision trees [2,14,16,18,26,30,42]. Other approaches also include polytope machines [6,25], rectangular rules [8,32], and layerwise relevance propagation [24]. To our knowledge, our work is the first to consider soft decision trees, to support scalable training, and to be extended to tree-based spectral and KPCA clustering.

Several works on interpretable clustering via decision trees focus on a setting in which each cluster corresponds to exactly one leaf, similar to hierarchical clustering [2,18,30,49]. This approach significantly restricts the expressive power of the decision trees and their ability to accurately match the observed clusters in the dataset. Similar to the recent ExKMC [16], our approach allows more than K leaves to support more expressive trees.

A very recent work has focused on clustering using hard, oblique decision trees via alternating optimization [17]. While their implementation is not available, their experiments show limited improvement over ExKMC for fully sparse (axis-aligned) trees. Different from our work, they focus on hard decision trees and their approach is not amenable to scalable, mini-batch, stochastic gradient descent optimization.

5 Conclusion

We present a novel approach for interpretable clustering based on soft clustering trees. We formulate the problem as a continuous optimization problem that can be efficiently solved by second-order optimizers, such as L-BFGS, as well as scalable SGD optimization. We extend our approach to support spectral and KPCA clustering trees. We conduct extensive experiments using 18 datasets and show that our spectral and KPCA approaches significantly outperform the state-of-the-art approach on small and medium datasets and our scalable training using SGD produces high quality clustering trees for large datasets.

Our work can be extended in a number of ways. Investigating approaches for joint construction of soft clustering trees where clustering is based on learned representations would be an interesting extension of our work. Investigating strategies to incorporate fairness considerations [9] is an important direction for future work. Finally, incorporating domain-specific knowledge in the form of constraints [11] could lead to higher-quality, yet interpretable, solutions.

References

1. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)
2. Bertsimas, D., Orfanoudaki, A., Wiberg, H.: Interpretable clustering: an optimization approach. Mach. Learn. **110**(1), 89–138 (2021)
3. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy C-means clustering algorithm. Comput. Geosci. **10**(2–3), 191–203 (1984)
4. Blanquero, R., Carrizosa, E., Molero-Río, C., Morales, D.R.: Sparsity in optimal randomized classification trees. Eur. J. Oper. Res. **284**(1), 255–272 (2020)
5. Blanquero, R., Carrizosa, E., Molero-Río, C., Morales, D.R.: Optimal randomized classification trees. Comput. Oper. Res. **132**, 105281 (2021)
6. Carrizosa, E., Kurishchenko, K., Marín, A., Morales, D.R.: Interpreting clusters via prototype optimization. Omega **107**, 102543 (2022)
7. Carrizosa, E., Molero-Río, C., Romero Morales, D.: Mathematical optimization in classification and regression trees. TOP **29**(1), 5–33 (2021). https://doi.org/10.1007/s11750-021-00594-1
8. Chen, J., et al.: Interpretable clustering via discriminative rectangle mixture model. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 823–828. IEEE (2016)
9. Chhabra, A., Masalkovaitė, K., Mohapatra, P.: An overview of fairness in clustering. IEEE Access **9**, 130698–130720 (2021)
10. Correia, G.M., Niculae, V., Martins, A.F.: Adaptively sparse transformers. In: Proceedings of the EMNLP-IJCNLP (2019, to appear)
11. Dao, T.B.H., Vrain, C., Duong, K.C., Davidson, I.: A framework for actionable clustering using constraint programming. In: Proceedings of the Twenty-Second European Conference on Artificial Intelligence, pp. 453–461 (2016)
12. Dua, D., Graff, C.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
13. Dunning, I., Huchette, J., Lubin, M.: JuMP: a modeling language for mathematical optimization. SIAM Rev. **59**(2), 295–320 (2017). https://doi.org/10.1137/15M1020575
14. Fraiman, R., Ghattas, B., Svarc, M.: Interpretable clustering using unsupervised binary trees. Adv. Data Anal. Classif. **7**(2), 125–145 (2013)
15. Frosst, N., Hinton, G.: Distilling a neural network into a soft decision tree. arXiv preprint arXiv:1711.09784 (2017)
16. Frost, N., Moshkovitz, M., Rashtchian, C.: ExKMC: expanding explainable *k*-means clustering. arXiv preprint arXiv:2006.02399 (2020)
17. Gabidolla, M., Carreira-Perpiñán, M.Á.: Optimal interpretable clustering using oblique decision trees. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 400–410 (2022)
18. Gamlath, B., Jia, X., Polak, A., Svensson, O.: Nearly-tight and oblivious algorithms for explainable clustering. Adv. Neural. Inf. Process. Syst. **34**, 28929–28939 (2021)
19. Hazimeh, H., Ponomareva, N., Mol, P., Tan, Z., Mazumder, R.: The tree ensemble layer: Differentiability meets conditional computation. In: International Conference on Machine Learning, pp. 4138–4148. PMLR (2020)
20. Hinton, G., Srivastava, N., Swersky, K.: Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. Cited on **14**(8), 2 (2012)
21. Hou, Q., Zhang, N., Kirschen, D.S., Du, E., Cheng, Y., Kang, C.: Sparse oblique decision tree for power system security rules extraction and embedding. IEEE Trans. Power Syst. **36**(2), 1605–1615 (2020)

22. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**(1), 193–218 (1985)
23. Irsoy, O., Yildiz, O.T., Alpaydin, E.: Soft decision trees. In: Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), pp. 1819–1822. IEEE (2012)
24. Kauffmann, J., Esders, M., Ruff, L., Montavon, G., Samek, W., Müller, K.R.: From clustering to cluster explanations via neural networks. IEEE Trans. Neural Netw. Learn. Syst. (2022)
25. Lawless, C., Kalagnanam, J., Nguyen, L.M., Phan, D., Reddy, C.: Interpretable clustering via multi-polytope machines. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7309–7316 (2022)
26. Liu, B., Xia, Y., Yu, P.S.: Clustering via decision tree construction. In: Chu, W., Young Lin, T. (eds.) Foundations and Advances in Data Mining. Studies in Fuzziness and Soft Computing, vol. 180, pp. 97–124. Springer, Heidelberg (2005). https://doi.org/10.1007/11362197_5
27. Luo, H., Cheng, F., Yu, H., Yi, Y.: SDTR: soft decision tree regressor for tabular data. IEEE Access **9**, 55999–56011 (2021)
28. Makarychev, K., Shan, L.: Near-optimal algorithms for explainable k-medians and k-means. In: International Conference on Machine Learning, pp. 7358–7367. PMLR (2021)
29. Martins, A., Astudillo, R.: From softmax to sparsemax: a sparse model of attention and multi-label classification. In: International Conference on Machine Learning, pp. 1614–1623. PMLR (2016)
30. Moshkovitz, M., Dasgupta, S., Rashtchian, C., Frost, N.: Explainable k-means and k-medians clustering. In: International Conference on Machine Learning, pp. 7055–7065. PMLR (2020)
31. Paszke, A., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
32. Pelleg, D., Moore, A.: Mixtures of rectangles: interpretable soft clustering. In: ICML, vol. 2001, pp. 401–408 (2001)
33. Peters, B., Niculae, V., Martins, A.F.: Sparse sequence-to-sequence models. arXiv preprint arXiv:1905.05702 (2019)
34. Popov, S., Morozov, S., Babenko, A.: Neural oblivious decision ensembles for deep learning on tabular data. arXiv preprint arXiv:1909.06312 (2019)
35. Rand, W.M.: Objective criteria for the evaluation of clustering methods. J. Am. Stat. Assoc. **66**(336), 846–850 (1971)
36. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**, 53–65 (1987)
37. Schölkopf, B., Smola, A., Müller, K.-R.: Kernel principal component analysis. In: Gerstner, W., Germond, A., Hasler, M., Nicoud, J.-D. (eds.) ICANN 1997. LNCS, vol. 1327, pp. 583–588. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0020217
38. Sculley, D.: Web-scale k-means clustering. In: Proceedings of the 19th International Conference on World Wide Web, pp. 1177–1178 (2010)
39. Smith, L.N.: Cyclical learning rates for training neural networks. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 464–472. IEEE (2017)
40. Strehl, A., Ghosh, J.: Cluster ensembles–a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. **3**, 583–617 (2002)

41. Tanno, R., Arulkumaran, K., Alexander, D., Criminisi, A., Nori, A.: Adaptive neural trees. In: International Conference on Machine Learning, pp. 6166–6175. PMLR (2019)
42. Tavallali, P., Tavallali, P., Singhal, M.: K-means tree: an optimal clustering tree for unsupervised learning. J. Supercomput. **77**(5), 5239–5266 (2021)
43. Ultsch, A., Lötsch, J.: The fundamental clustering and projection suite (FCPS): a dataset collection to test the performance of clustering and data projection algorithms. Data **5**(1), 13 (2020)
44. Von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
45. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. **106**(1), 25–57 (2006)
46. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp. 478–487. PMLR (2016)
47. Yang, Y., Morillo, I.G., Hospedales, T.M.: Deep neural decision trees. In: ICML Workshop on Human Interpretability in Machine Learning (WHI) (2018)
48. Yoo, J., Sael, L.: EDiT: interpreting ensemble models via compact soft decision trees. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 1438–1443. IEEE (2019)
49. Zantedeschi, V., Kusner, M., Niculae, V.: Learning binary decision trees by argmin differentiation. In: International Conference on Machine Learning, pp. 12298–12309. PMLR (2021)