斜决策树实现最佳可解释聚类算法

Optimal Interpretable Clustering Using Oblique Decision Trees
Magzhan Gabidolla and Miguel Á. Carreira-Perpiñán.
In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22).

1 背景

随着机器学习算法在各种领域中得到越来越广泛的应用，对预测模型内部工作方式的解释也变的愈发重要。随着模型的复杂性也不断增加，使用缺乏透明度的黑盒子AI模型（如神经网络和随机或提升森林）的现象也越来越普遍。此外，许多关键和敏感领域要求模型或算法决策在某种程度上可解释，以便可以信任或审计（针对公平性、错误等）。

现有的可解释模型绝大部分集中在分类和回归方面，旨在对在特定数据集上做出预测的依据进行解释。**现有的基于决策树的可解释聚类方法**（CART，C5.0及各种基于启发式方法优化分裂标准的贪心递归划分技术）**均只考虑了轴对称树，通过判断单个特征是否超过阈值进行分割**。作为更加灵活的模型，使用数个特征的线性组合作为分割依据的**斜决策树目前尚未在任何可解释性聚类方法中得到应用**。

2 解决的问题

本文通过稀疏斜决策树产生成本函数，由此重新对所有由成本函数决定聚类分配的方法进行重新定义。相较于其他已有方法，该方法具有以下优势：

- 更好的可解释性
    - k-means 方法无法说明聚类具体是由哪些特征决定的（全部或子集），并且难以对高纬聚类区域进行精确描述。
    - 轴对称决策树和斜决策树模型，均可以通过从根到叶子的路径来解释输入是根据哪些特征被分配到特定簇的。
- 更高的准确性

    - 轴对称决策树的每个决策节点以单个输入特征作为依据。一颗具有 L 个叶子节点的二叉树仅有 L-1 个决策节点，即整棵树**最多**使用了 L-1 个特征。在从根到叶子的路径来看，这个数量甚至更少（完整二叉树仅 $log_2 L$）。
    - 斜决策树使用数个特征的线性组合形成的超平面对高纬数据进行划分，具有更高的准确性。
- 更快的推断

  对于 D 维数据：k-means 的时间复杂度为 $O(DK)$，稠密决策树的开销仅为需要 $O(Dh)$，稀疏树则更少 —— 这在 $K \gg h$ 时显然节省了大量时间成本。

  本文在实践中使用树交替优化（TAO）算法进行剪枝和权重向量稀疏化，学习得到精确的稀疏斜决策树，且每个节点使用较少的特征作为判断依据，具有较高的聚类效率。
- 准确性与可解释性的平衡

  完全准确的聚类可以通过简单的生长一棵足够大的实现，但这会导致过拟合与可解释性的下降。本文提供了超参数 λ，用户可以通过探索性的方式进行调整，以找到一个在预测准确性和解释简洁性之间达成良好平衡的树。

3 方法

3.1 可解释性聚类问题的定义

对于具有参数 Θ 的决策树，我们认为其代表的聚类算法目标函数如下：

$$\min_{\Psi,\Theta} E(T(X;\Theta),\Psi) + \lambda\phi(\Theta),\ \lambda \geq 0 \tag{1}$$

其中：

- Ψ 包含了由聚类算法学习到的其他变量，如 k-means 的聚类中心点
- $T(\cdot;\Theta) : \mathbb{R}^D \to \{1,\dots,K\}$ 表示训练得到的决策树，实现输入数据到 K 个聚类结果 one-hot 编码结果的映射
- 正则化项 $\phi(\Theta)$ 与用户决定的超参数 λ 一同控制树的复杂程度

由于决策树 T 是一个不可微函数，此处使用辅助坐标法，引入赋值变量将（1）重写为一个约束性问题：

$$\begin{gathered} \min_{Z,\Psi,\Theta} E(Z,\Psi) + \lambda\phi(\Theta),\ \lambda \geq 0 \\ Z = T(X;\Theta),\ Z^T 1 = 1,\ Z \in \{0,1\}^{K\times N} \end{gathered} \tag{2}$$

3.2 聚类优化算法

这一步引入了惩罚函数，并将问题改写如下：

$$\begin{aligned} &\min_{Z,\Psi,\Theta} E(Z,\Psi)+\lambda\phi(\Theta)+\mu P(Z,T(X;\Theta)) \\ &Z^T 1=1,\ Z\in\{0,1\}^{K\times N} \end{aligned} \tag{3}$$

其中：

- $\mu \geq 0$ 是一个惩罚参数
- P 是一个满足 $P(z,z)=0,\ P(z,z')>0\ (z\neq z')$ 的惩罚函数

当 $\mu \to \infty$ 时，(2) (3) 将具有相同解；且随着 μ 的增大，(3) 将愈发难以求解。因此，优化将像二次惩罚等常见方法那样，从较小的 μ 开始。

对于固定的 μ，执行如图的算法进行优化：

input $\mathbf{X}_{D\times N}=\{\mathbf{x}_1,\cdots,\mathbf{x}_N\}$, $\lambda\geq 0$, $a>0$, $\mu_0>0$
initial tree structure and random $\boldsymbol{\Theta}$
$\mathbf{Z},\boldsymbol{\Psi} \leftarrow \arg\min E(\mathbf{Z},\boldsymbol{\Psi})$ s.t. $\mathbf{Z}^T\mathbf{1}=\mathbf{1}$, $\mathbf{Z}\in\{0,1\}^{K\times N}$ — Free clustering
$\boldsymbol{\Theta} \leftarrow \begin{cases} \arg\min P(\mathbf{Z},\mathbf{T}(\mathbf{X};\boldsymbol{\Theta})), & \lambda=0 \\ 0, & \lambda>0 \end{cases}$ — Direct tree fit
$\mu \leftarrow \mu_0$
repeat
  $\mathbf{Z},\boldsymbol{\Psi} \leftarrow \arg\min E(\mathbf{Z},\boldsymbol{\Psi})+\mu P(\mathbf{Z},\mathbf{T}(\mathbf{X};\boldsymbol{\Theta}))$ s.t. $\mathbf{Z}^T\mathbf{1}=\mathbf{1},\ \mathbf{Z}\in\{0,1\}^{K\times N}$ — Clustering step
  $\boldsymbol{\Theta} \leftarrow P(\mathbf{Z},\mathbf{T}(\mathbf{X};\boldsymbol{\Theta}))+\frac{\lambda}{\mu}\phi(\boldsymbol{\Theta})$ — Tree step
  $\mu \leftarrow \mu\cdot a$
until $\mathbf{Z}=\mathbf{T}(\mathbf{X};\boldsymbol{\Theta})$ and no parameter change
return tree $\mathbf{T}(\cdot;\boldsymbol{\Theta})$ and $\mathbf{Z},\boldsymbol{\Psi}$

3.2.1 惩罚函数选择

下面是两种较为简便的算法实现：

1. 当叶节点具有 one-hot 编码标签时，令 $P=01$ 损失函数
2. 当叶节点具有描述该节点下各类数据占比的直方图时，令 $P=$ 方差损失函数

3.2.2 TAO 算法

TAO算法实现了Tree Step 中的惩罚项迭代，其基本流程如下：

1. 从树的最底层开始，逐层往上推进。只考虑当前层内的所有子问题，并保持其它层的参数不变。

2. 对于每个子问题，进行一次优化，得到新的参数值。
3. 将更新后的参数值传递给下一层的相应子问题。
4. 不断重复步骤3和步骤4，直到达到整个树形结构的根节点。
5. 如果满足收敛条件，则结束算法，否则返回步骤1继续迭代。

在本文需要建立的斜决策树模型中，子问题根据树节点类型可被划分为两类：

- 决策节点：

  子问题是一个加权的0/1损失二分类问题，每个实例被分配至具有较低损失的子节点。

  简化后的问题为：

  $$\underset{w_i,w_{i0}}{min}\sum_{n\in R_i}\overline{L}(z_n,f_i(x;w_i,w_{i0}))+\lambda\|w_i\|_1$$

  这个问题对于斜决策树是NP-Hard的，此处使用 $l1$ 正则化逻辑回归，每个实例的权重为子节点间的损失差。

- 叶子节点：

  子问题是对在叶节点包含数据上训练分类器的原始损失进行优化：

  $$\underset{\theta_i}{min}\sum_{n\in R_i}P(z_n,\theta_i)$$

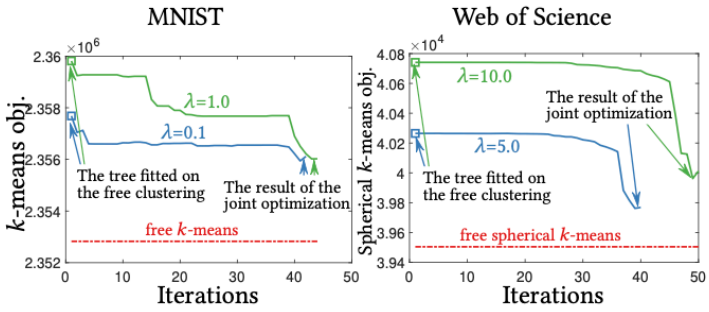
4 结果

在本文实验中对比的双方分别为：

- 通过递归使用 k=2 的 标准 / 球形 k-means 算法生成的完全决策树
- 使用本文算法生成的轴对称决策树与斜决策树

作者在灰度图数据集 MNIST & FashionMNIST 和统计矩数据集 Letter 上对两种方法进行了对比，得出的结论如下：

1. 使用本文算法生成的决策树（尤其是斜决策树）比直接将 TAO 树拟合到自由聚类上具有更好的性能，且与 k-means 的具有相近的分类质量
2. 生成的斜决策树是稀疏、较浅且易于解释的



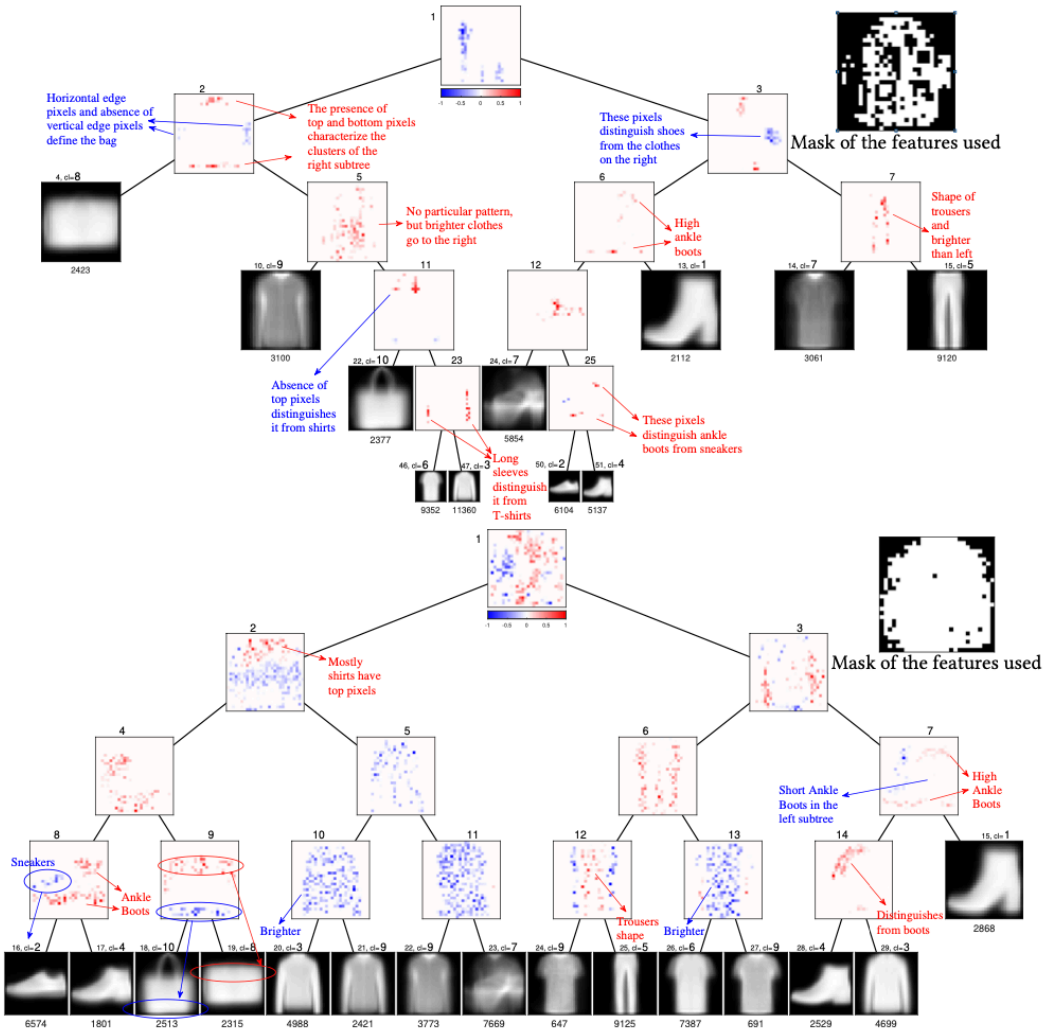
- 该图的纵轴为目标函数值：
  - 红线为使用 标准 / 球形 k-means 算法 时的目标函数值
  - 蓝绿线为 λ 取不同值时，目标函数随 TAO 算法迭代的变化曲线；起点为直接将 TAO 树拟合至自由聚类时的目标函数取值

  显然，目标函数随着 TAO 迭代次数增加而逐渐下降，最终与 k-means 聚类效果仅有2%以内的差距

	Method	cost (%)	#parameters	#features/node	Δ	#leaves
MNIST (60k,784,10)	IMM	14.34	28	1	9	10
	Ex-Greedy	12.48	28	1	8	10
	CART	11.54	28	1	4	10
	TAO	7.90	199	23	4	9
	CART	1.87	3070	1	16	1024
	ExKMC	1.81	3070	1	29	1024
	TAO	1.50	753	66	5	12
	TAO	0.94	1372	96	4	15
FashMNIST (60k,784,10)	IMM	28.34	28	1	9	10
	CART	23.33	28	1	4	10
	Ex-Greedy	17.28	28	1	7	10
	TAO	5.22	452	43	5	11
	ExKMC	2.49	3070	1	59	1024
	CART	1.75	3070	1	17	1024
	TAO	1.74	825	80	5	11
	TAO	0.44	2081	146	4	15
Letter (20k,16,26)	IMM	35.02	76	1	25	26
	CART	30.61	76	1	10	26
	ExGreedy	27.78	76	1	21	26
	TAO	9.94	523	15	5	32
	TAO	4.06	516	8	6	52
	CART	2.89	3070	1	25	1024
	ExKMC	2.91	3070	1	39	1024
	TAO	2.75	858	12	6	64

- `cost` 字段是该方法相较 k-means 方法增加的训练代价百分比
- `parameter` 为输入参数数量，`feature` 为所欲决策节点使用的平均特征数量
- Δ 为树的深度

  显然，TAO 相较于其他方法的成本较小，并能构建更浅、更稀疏的决策树。同时也是唯一能在单个决策节点上使用多个特征进行判断的聚类方法。



- 该图展示了在 FashionMNIST 数据集上，使用不同稀疏度参数得到的斜决策树
  - 上图参数为 $\lambda = 100$，$\Delta = 5$，相较 k-means 的失真度为 5.22%
  - 下图参数为 $\lambda = 10$，$\Delta = 4$，较 k-means 的失真度仅为 0.44%
- 决策节点被可视化为 28*28 的图像，红蓝两色像素点分别为将输入划分至 右/左 子树的依据
- 叶子节点下方可视化了达到该节点图像的平均值

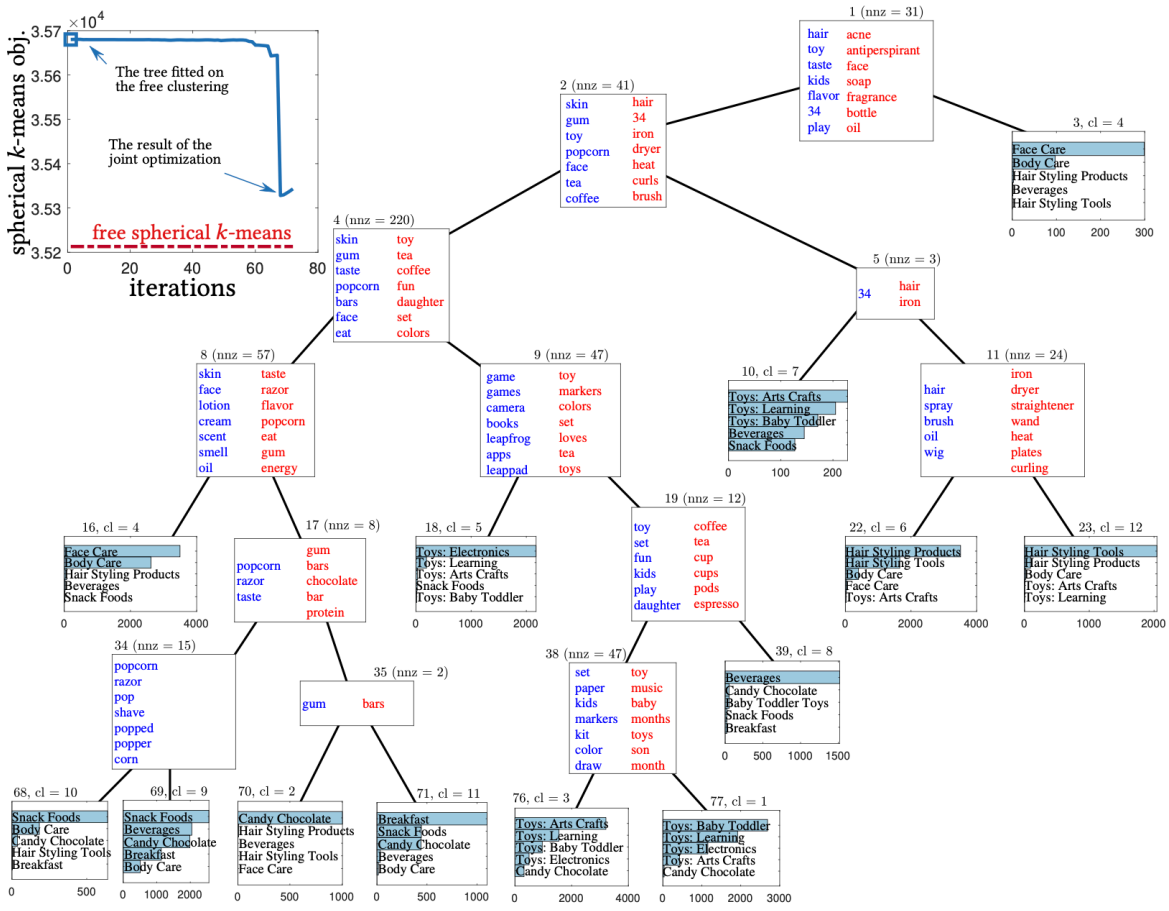
5 未来工作及个人看法

原文作者并未对 futrue work 进行展望，以下均为个人看法

- 优点

  本文提出的模型开创性的使用倾斜二叉树作为决策树，同时使用 TAO 算法实现减枝，在降低分类开销的同时提升了准确性，为可解释聚类模型（决策树）提供了更优的选项。

- 不足与展望
  - 在单个决策节点上使用若干个特征的线性组合作为决策依据事实上导致了可解释性的下降



图为该方法在 Amazon 评论数据集上的应用结果，决策节点中的红蓝色分别表示了该节点判据中权重最高/最低的7个特征，nnz 为当前决策节点中非零权重的数量。

显然，该方法无法直观的呈现各特征所占权重的具体比例，及其形成的超平面是如何对输入数据进行划分的。

在未来的工作中，可以进一步探究如何有效对超平面进行可视化解释。

  - 参数调优较为困难

在 k-means 方法中，我对聚类中心数量进行调整优化，其调整原因及产生的结果都是十分直观的。

本文描述的方法中提供了超参数 λ 让用户进行“探索性”的调整，以平衡生成决策树的准确性与可解释性。但 λ 本身的取值范围过于广泛（任何大 ≥ 0 的数，而 $k \in \mathbb{N}^+$），且其对最终生成的斜决策树的影响是不确定的，这似的调优过程较为困难。

在未来的工作中，可以进一步探究超参数 λ 与最终生成的斜决策树之间的具体关系。