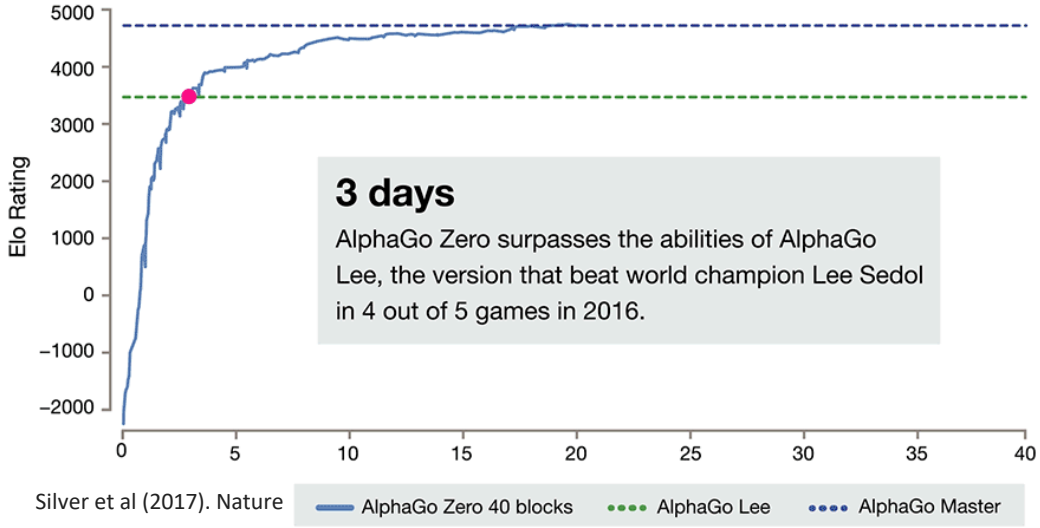
When Global Optimization Meets Machine Learning: Challenges and Opportunities

Yankai Cao

University of British Columbia

THE UNIVERSITY OF BRITISH COLUMBIA

Machine Learning vs Human Learning



Silver et al (2017). Nature

Why are humans more data-efficient?
4.9 million games vs 3 thousand

Hypothesis:

- Transfer learning
- Reasoning + Hybrid Modelling
- Higher solution accuracy?

→ Can global optimization improve the performance of ML?

Prevailing Mindsets in Machine Learning Community

The most obvious drawback of the learning procedure is that the error-surface may contain local minima so that gradient descent is not guaranteed to find a global minimum. However, experience with many tasks shows that the network very rarely gets stuck in poor local minima that are significantly worse than the global minimum.

— David Rumelhart, **Geoffery Hinton**, Ronald Williams
Learning Representations by Back-propagating Errors, Nature (1986)

In the late 1990s, neural nets and backpropagation were largely forsaken by the machine-learning community and ignored by the computer-vision and speech-recognition communities. It was widely thought that learning useful, multistage, feature extractors with little prior knowledge was infeasible. In particular, it was commonly thought that simple gradient descent would get trapped in poor local minima …

— Yann LeCun, Yoshua Bengio, **Geoffery Hinton**
Deep Learning, Nature (2015)

Prevailing Mindsets in Machine Learning Community – Part 2

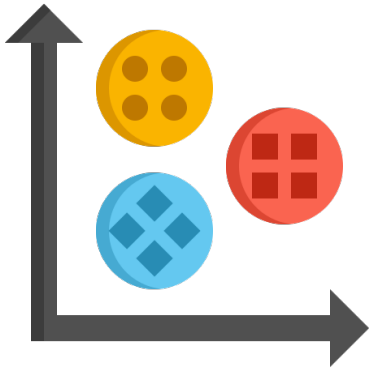
Belief 1 (1990s - now): Solving machine learning problems to global optimum is impossible using state-of-the-art solvers, especially for large datasets.

- It is widely believed that deterministic global algorithms are too slow (Hamm et al., 2007).
- In 1990s and 2000s, stochastic global optimization methods were very popular, including Evolving Neural Networks (Yao et al., 1999, Stanley et al., 2002), Genetic Algorithms (Miller et al., 1989, Leung et al., 2003), and Simulated Annealing (Goffe et al.,1994).

Belief 2 (2010s - now): Solving ML problem to local optimum is good enough.

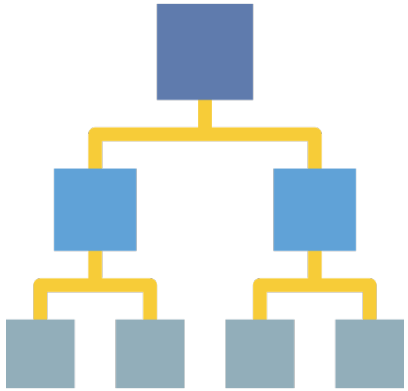
- With strong simplification assumptions on the distribution of data and network weight parameters, as the number of hidden units increases, all local minima become increasingly close to being global minima (Choromanska et al. 2015).
- If a neural net is strongly over-parameterized so that I can memorize any dataset, then all critical points become global minima (Livni et al. 2014, Nguyen & Hein 2017).
- For neural nets with one hidden layer and a convolutional structure with no overlap, and a ReLU activation function, if inputs are independent gaussian variables, gradient descent converges to the global optimum (Brutzkus & Globerson, 2017)

We Focus on the Global Optimization of



Centroid-based Clustering

[Hua et al. ICML 2021spotlight]
[Shi et al. ICML 2022]
[Ren et al. NeurIPS 2022]



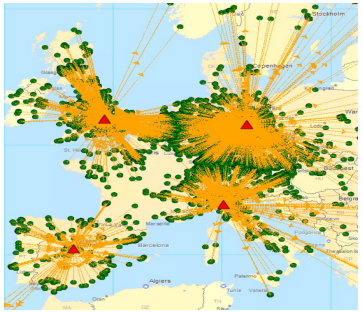
Decision Tree

[Hua et al. NeurIPS 2022]
2022 INFORMS Data Mining Best Paper

Applications with High Stake

Bail Decision

Medical Diagnosis



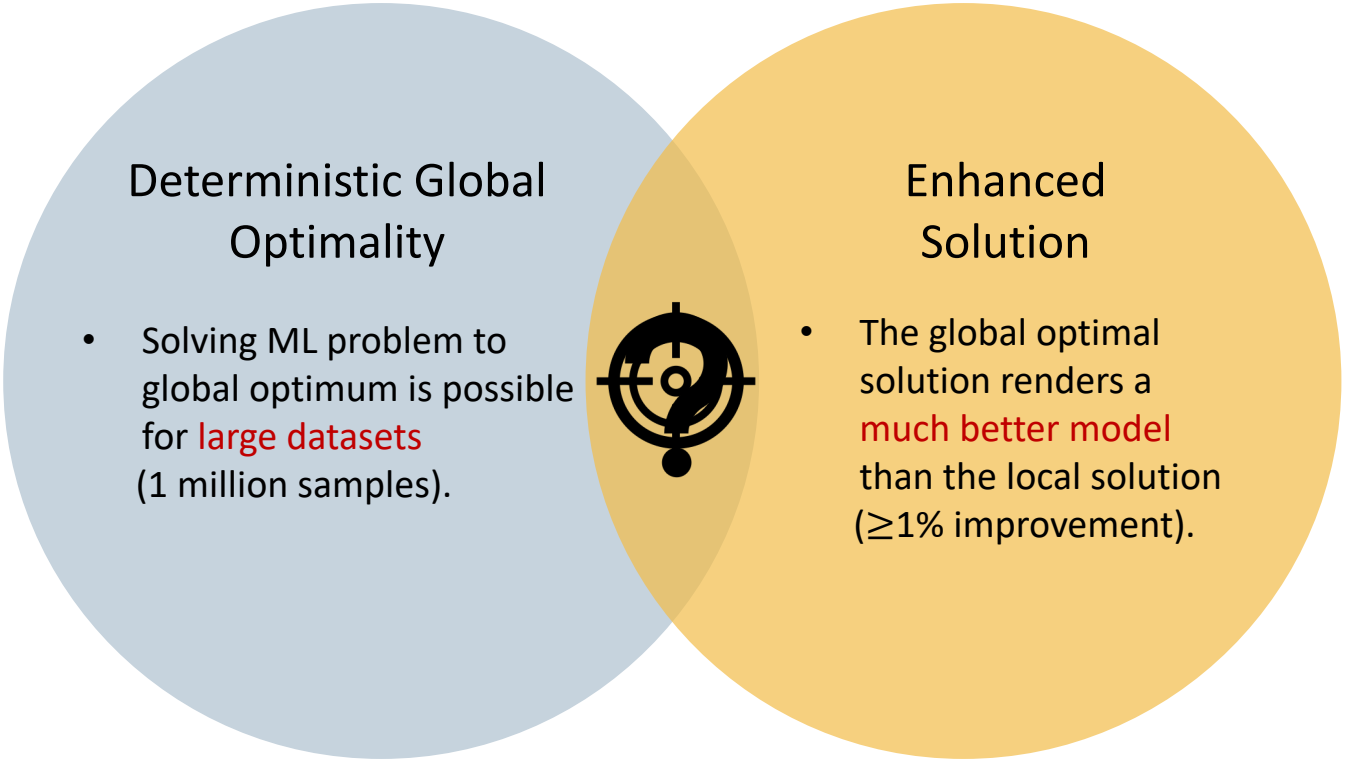
Facility Location Problem

Interpretable

Intuitive

? **Heuristic Algorithm -> Suboptimal Solutions**

Our Hypothesis



Outline

- ML as Stochastic Mixed Integer Problems
  - Reduced Space Branch and Bound Algorithm
- Centroid-based Clustering
- Optimal Decision Tree

Machine Learning Problems are MI(N)LP Problems

Given:

- $\{x_1, \dots, x_s, \dots, x_n\} \in \mathbb{R}^P$,
- $\{y_1, \dots, y_s, \dots, y_n\} \in \{0,1\}^K$,
- $\mathcal{S} = \{1, \dots, s, \dots, n\}$
- Model representation h

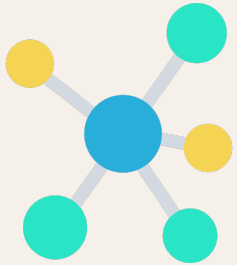
Output:

- Optimal parameters $m \in M$

Branch and Bound (BB) Framework

Supervised Learning

$$\mathcal{L}(M) \coloneqq \min_{m \in M, \hat{y}_s} \sum_{s \in \mathcal{S}} E(\hat{y}_s, y_s) + \gamma R(m)$$
$$s.t. \quad \hat{y}_s = h(m, x_s)$$
$$s \in \mathcal{S}$$



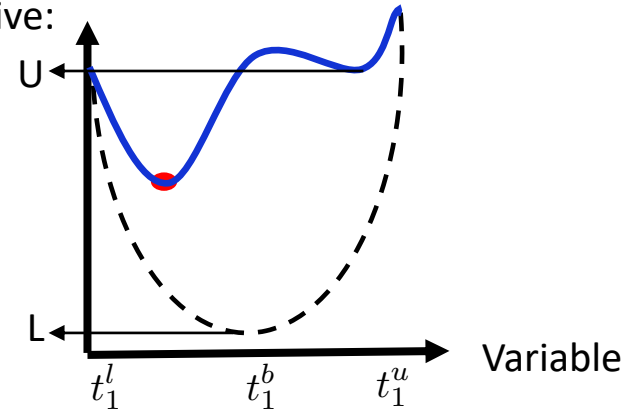
Unsupervised Learning

$$\mathcal{L}(M) \coloneqq \min_{m \in M, \hat{x}_s} \sum_{s \in \mathcal{S}} E(\hat{x}_s, x_s) + \gamma R(m)$$
$$s.t. \quad \hat{x}_s = h(m, x_s)$$
$$s \in \mathcal{S}$$

Branch and Bound Framework

$$\mathcal{L}(M) \coloneqq \min_{m \in M, \hat{y}_s} \sum_{s \in \mathcal{S}} E(\hat{y}_s, y_s) + \gamma R(m)$$
$$s.t. \quad \hat{y}_s = h\,(m, x_s)$$
$$s \in \mathcal{S}$$

Objective:



Root node

Branch and Bound Framework

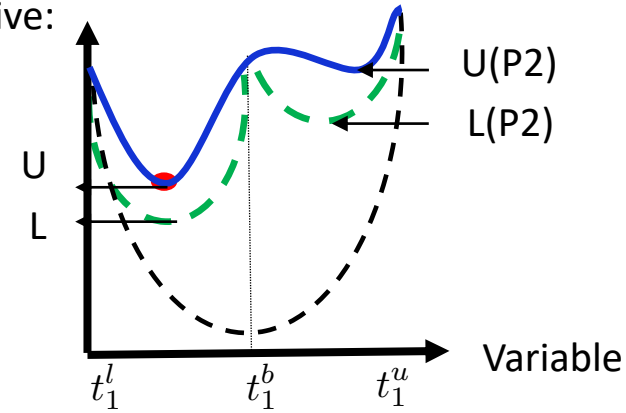
$$\mathcal{L}(M) \coloneqq \min_{m \in M, \hat{y}_s} \sum_{s \in \mathcal{S}} E(\hat{y}_s, y_s) + \gamma R(m)$$
$$s.t. \quad \hat{y}_s = h(m, x_s)$$
$$s \in \mathcal{S}$$

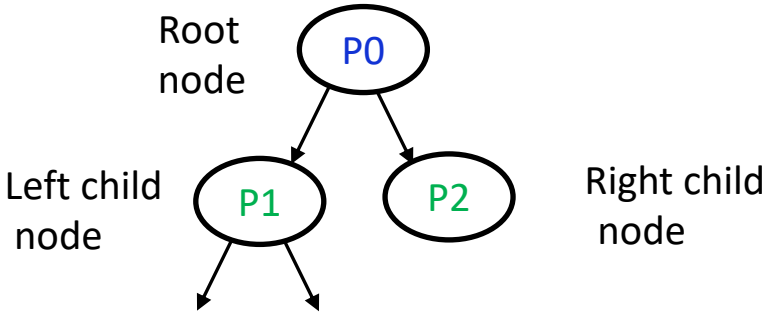
Limitation: Need to branch on the full variable space

For Decision Tree: # Integer $2^D(n + P + K + 1)$
e.g. depth $D = 2$ and n =1 million samples
-> 4 million integer variables

Key: Need to exploit the structure

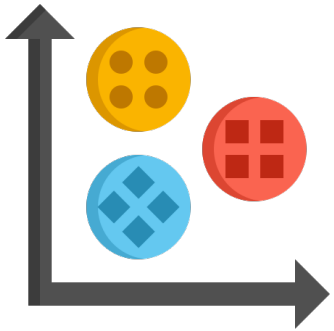
Objective:





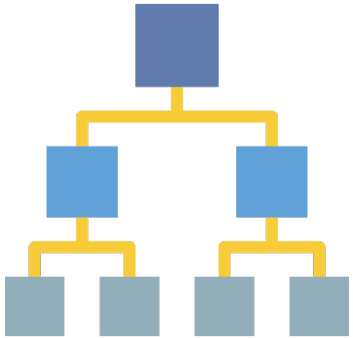
Machine Learning Problems are Two-stage Stochastic Programming Problems

First Stage Variables — Model Parameter



Centroid-based Clustering

Second Stage Variables — Sample Evaluation



Decision Tree

Machine Learning Problems are Two-stage Stochastic Programming Problems

First Stage Variables | Model Parameter

Second Stage Variables | Sample Evaluation

$$\mathcal{L}(M) \coloneqq \min_{m \in M, \hat{x}_s} \sum_{s \in \mathcal{S}} E(\hat{x}_s, x_s) + \gamma R(m)$$
$$s.t. \quad \hat{x}_s = h(m, x_s)$$
$$s \in \mathcal{S}$$

$$\mathcal{L}(M) \coloneqq \min_{m \in M} \sum_{s \in \mathcal{S}} Q_s(m)$$

Where:

$$Q_s(m) \coloneqq \min_{\hat{y}_s} E(\hat{y}_s, x_s) + \frac{\gamma}{|\mathcal{S}|} R(m)$$
$$s.t. \quad \hat{y}_s = h(m, x_s)$$

Various approaches proposed in the stochastic programming community can be utilized to exploit the problem structure (Karuppiah & Grossmann 2008, Li et al. 2011, Cao & Zavala 2019, Li & Grossman 2019, Ogbe & Li 2019, Kannan 2019).

Reduced Space Branch and Bound

Machine Learning Problems:

$$\mathcal{L}(M) \coloneqq \min_{m \in M} \sum_{s \in \mathcal{S}} Q_s(m)$$

Lower Bounding Problem:
(wait and see problem)

$$\beta(M) \coloneqq \min_{m_s \in M} \sum_{s \in \mathcal{S}} Q_s(m_s)$$

Upper Bounding Problem:
(fix $\widehat{m} \in M$)

$$\alpha(M) \coloneqq \sum_{s \in \mathcal{S}} Q_s(\widehat{m})$$

Assumption

Around solution m^*, $Q_s(m)$ is continuous with respect to m_c (i.e., continuous first-stage).

Theorem [Cao & Zavala, 2019]

With the above lower and upper bounds, the BB procedure can *converge* by branching only on first stage variables (i.e., model parameters)

For Decision Tree: e.g. depth $D = 2$ and $n =$1 million samples, with 3 features and 2 classes
branching variables: 4 million $\longrightarrow$ 23
Independent to the number of samples

Reduced Space Branch and Bound

Lower Bound Strategy

Upper Bound Strategy

Bound Tightening

Centroid-based Clustering

Decision Tree

Clustering Problems

Given: Dataset $X = \{x_1, \dots, x_S\} \in \mathbb{R}^{A \times S}$

Desired cluster number, K

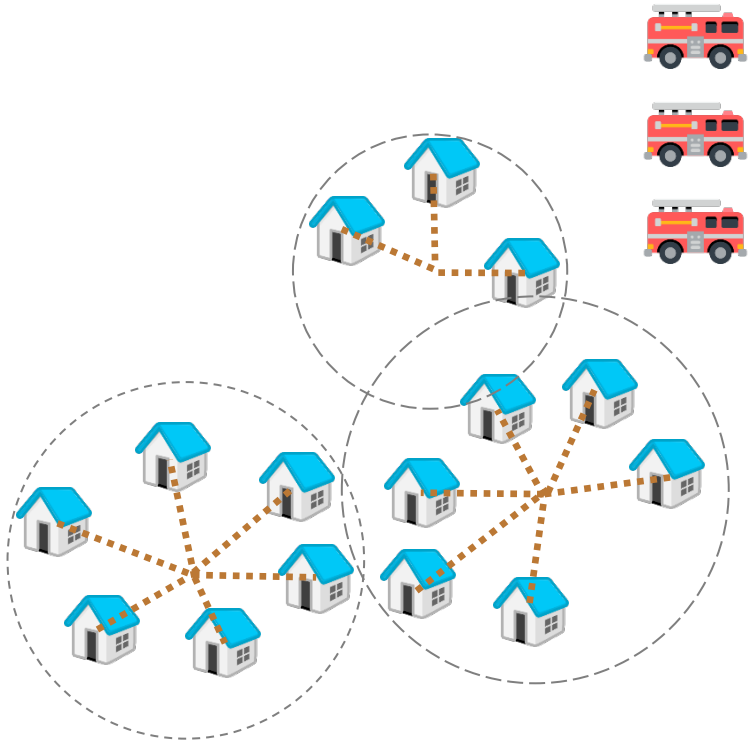
Output: m_k the center of cluster k

Object:

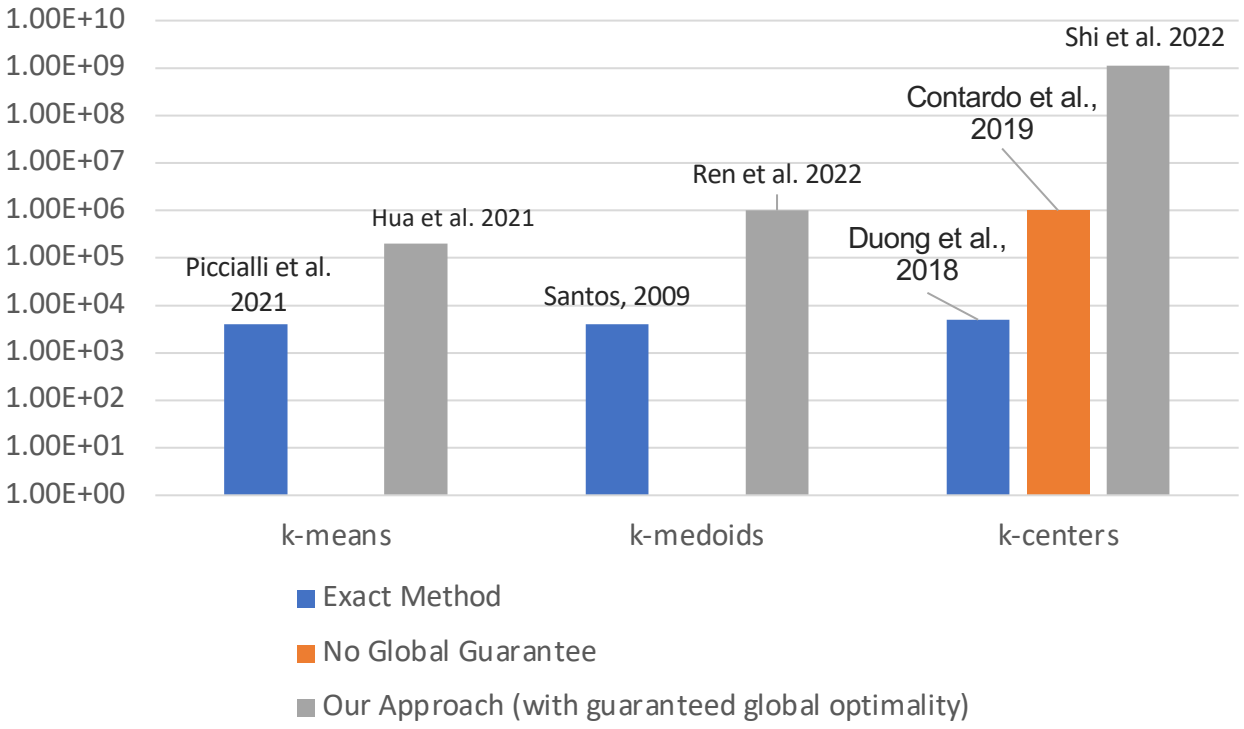
k-means : $\min_{m} \sum_{s \in \mathcal{S}} \min_{k \in \mathcal{K}} \|x_s - m_k\|_2^2$

k-medoids: $\min_{m \in X} \sum_{s \in \mathcal{S}} \min_{k \in \mathcal{K}} \|x_s - m_k\|_2^2$

k- centers: $\min_{m \in X} \max_{s \in \mathcal{S}} \min_{k \in \mathcal{K}} \|x_s - m_k\|_2^2$



Recent Progress on the Scalability of Data Size



MINLP Formulation

k-means : $\min_{m} \sum_{s \in \mathcal{S}} \min_{k \in \mathcal{K}} \|x_s - m_k\|_2^2$

$$b_{s,k} = \begin{cases} 1 & \text{if } x_s \text{ is in cluster } k. \\ 0 & \text{otherwise.} \end{cases}$$

$d_{s,k}$ is the distance between x_s and the cluster center m_k

$d_{s,*}$ is $\min_k d_{s,k}$

$$\min_{m,d,b} \sum_{s \in \mathcal{S}} d_{s,*}$$

$$s.t. \; -N(1 - b_{s,k}) \leq d_{s,*} - d_{s,k} \leq N(1 - b_{s,k})$$

$$d_{s,k} \geq \|x_s - m_k\|^2$$

$$\sum_{k \in \mathcal{K}} b_{s,k} = 1$$

$$b_{s,k} \in \{0,1\}$$

$$s \in \mathcal{S}, k \in \mathcal{K}$$

First Stage Variables

m: $K \times P$

(Center of clusters, continous)

Second Stage Variables

$b_{s,k}, d_{s,k}, d_{s,*}$

(Cluster assignment, mixed integer)

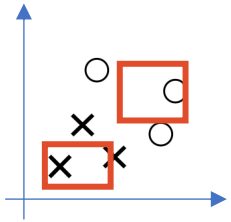
Basic Lower Bound

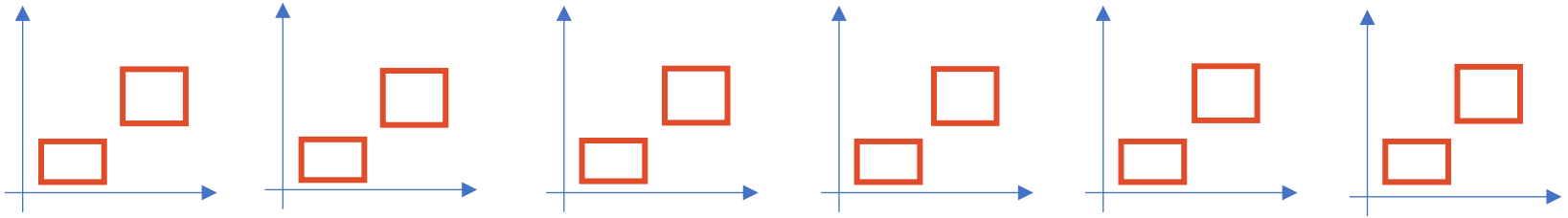
$$\min_{m \in M} \sum_{s \in \mathcal{S}} Q_s(m) \quad s.t. \; m_s = m_{s+1}$$

$$\Rightarrow \quad \beta^B(M) \coloneqq \min_{m_s \in M} \sum_{s \in \mathcal{S}} Q_s(m_s) \quad \Rightarrow \quad \beta_s(M) \coloneqq \min_{m_s \in M} Q_s(m_s)$$

non-anticipativity constraint

Decompose into $|\mathcal{S}| = n$ subproblems





Basic Lower Bound

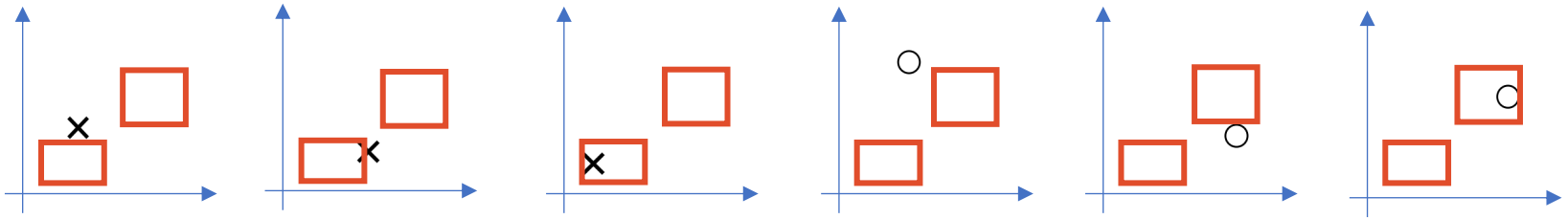
$$\beta_s(M) \coloneqq \min_{m_s \in M} Q_s(m_s) \quad \Rightarrow \quad \beta_s(M) \coloneqq \min_k \beta_{s,k}(M_k) = \min_k \min_{m_{s,k} \in M_k} \|x_s - m_{s,k}\|_2^2$$

$M_k \coloneqq \{m_{s,k} \mid \underline{m}_k \leq m_{s,k} \leq \overline{m}_k\}$

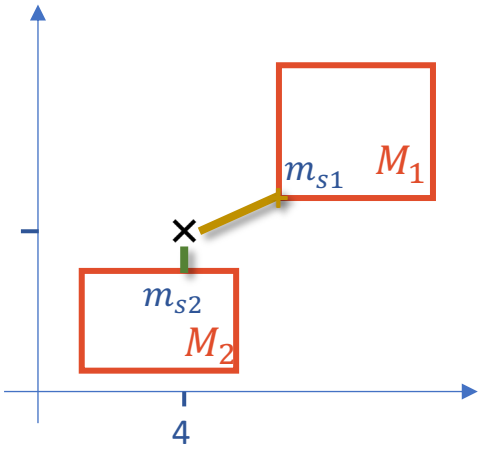
Decompose into $|\mathcal{S}| = n$ subproblems

$\beta_{s,k}$ has **closed form** solution:

$$m_{s,k,i} = med\{\underline{m}_{k,i}, x_{s,i}, \overline{m}_{k,i}\}, i \in \{1, \dots, P\}$$



Basic Lower Bound



$$M_k := \{m_{s,k} \mid \underline{m}_k \le m_{s,k} \le \overline{m}_k\}$$

$$\beta_s(M) := \min_k \beta_{s,k}(M_k) = \min_k \min_{m_{s,k} \in M_k} \|x_s - m_{s,k}\|_2^2$$

$\beta_{s,k}$ has **closed form** solution:

$$m_{s,k,i} = med\{\underline{m}_{k,i}, x_{s,i}, \overline{m}_{k,i}\}, i \in \{1, \dots, P\}$$

Calculation can be easily parallelized

No need to solve any optimization problem

Assuming 2 cluster, 2 features (K=2, P=2):

Subproblem s: $x_s = (4,5)$ $\beta_s(M) = 1$

- Cluster 1 center: $m_{s,1,1} = med\{4, 6, 10\} = 6$

  $m_{s,1,2} = med\{5, 6, 10\} = 6$

- Object value: $\beta_{s,1}(M_1) = \sqrt{(6-4)^2 + (6-5)^2} = \sqrt{5}$

Lagrangian Relaxation

Dualized the non-anticipativity constraints and added to the objective functions with Lagrange multipliers λ

λ can be optimized via sub-gradient method [Held and Karp, 1971]

Proposition:
$\beta^B(M) = \beta^{LD}(M,0) \leq \beta^{LD}(M) \leq \mathcal{L}(M)$

$$\beta^{LD}(M,\lambda) := \min_{m_s \in \mathrm{M}} \sum_{s \in \mathcal{S}} Q_s(m_s) + \sum_{s=1}^{|\mathcal{S}|-1} \lambda_s^T (m_s - m_{s+1})$$

$$\beta_s^{LD}(M,\lambda) := \min_{m_s \in \mathrm{M}} Q_s(m_s) + (\lambda_s - \lambda_{s+1})^T m_s$$

$\lambda_0 = \lambda_s = 0$

$$\beta^{LD}(M) := \max_{\lambda} \beta^{LD}(M,\lambda)$$

Lagrangian Relaxation provides a tighter bound than the basic lower bound, at the cost of significantly higher computational cost per node, with one exception.

Lagrangian Relaxation for K-medoid

$$\min_{b,y} \sum_{s\in\mathcal{S}}\sum_{j\in\mathcal{S}} d_{s,j}b_{s,j}$$

s.t. $\sum_{j\in\mathcal{S}} b_{s,j} = 1$

$$\sum_{j\in\mathcal{S}} y_j^k = 1$$

$$\sum_{k\in\mathcal{K}} y_j^k \le 1$$

$$b_{s,j} \le \sum_{k\in\mathcal{K}} y_j^k$$

$$b_{s,j}, y_j^k \in \{0,1\}$$

Lagrangian Relaxation

$$\beta_{LD}(M,\lambda) = \min_{b,y} \sum_{s\in\mathcal{S}}\left[\sum_{j\in\mathcal{S}}\left(d_{s,j}-\lambda_s\right)b_{s,j} + \lambda_s\right]$$

Closed-form solution (for a given λ):

$$\rho_j(\lambda) := \sum_{s\in\mathcal{S}} \min\left(0, d_{s,j}-\lambda_s\right),$$

$$\beta_{LD}(M,\lambda) = \min_{y} \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{S}} \rho_j(\lambda) y_j^k + \sum_{s\in\mathcal{S}} \lambda_s$$

Quality guarantee of LB:

$$Gap = \frac{z-\beta_{LD}}{z_r-\beta_{LD}} < \frac{1}{e}$$

Cornuejols, Fisher, and Nemhauser. Management Science (1977).

Sample Grouping

Decompose into
$G = |\mathcal{G}| \leq n$ subproblems

Non-anticipativity constraint on samples within groups are maintained

Proposition:
$\beta^B(M) \leq \beta^{SG}(M) \leq \mathcal{L}(M)$

$$\mathcal{L}(M) \coloneqq \min_{m \in M} \sum_{g \in \mathcal{G}} Q_g(m)$$

$$s.t. \boxed{m_g = m_{g+1}}$$

non-anticipativity constraint

$$\beta^{SG}(M) \coloneqq \min_{m_g \in M} \sum_{g \in \mathcal{G}} Q_g(m_g)$$

$$\beta_g^{SG}(M) \coloneqq \min_{m_g \in M} Q_g(m_g)$$

Upper Bounding Methods

Basic Upper bound: Fix m at a candidate solution $\hat{m} \in M$

$$\alpha(M) := \sum_{s \in \mathcal{S}} Q_s(\hat{m})$$

- Decomposable and trivial parallelism
- Solutions from lower bound subproblems

Heuristic methods on root node

- K-means: Lloyd Algorithm [Lloyd, 1982]
- K-center: Farthest First Traversal [Gonzalez, 1985]
- K-medoids: PAM [Kaufman and Rousseeuw, 1990]

Theorem [Hua et al. 2021]

With the basic lower and upper bounds, the BB procedure can *converge* by branching only on the centers $\hat{m}$.

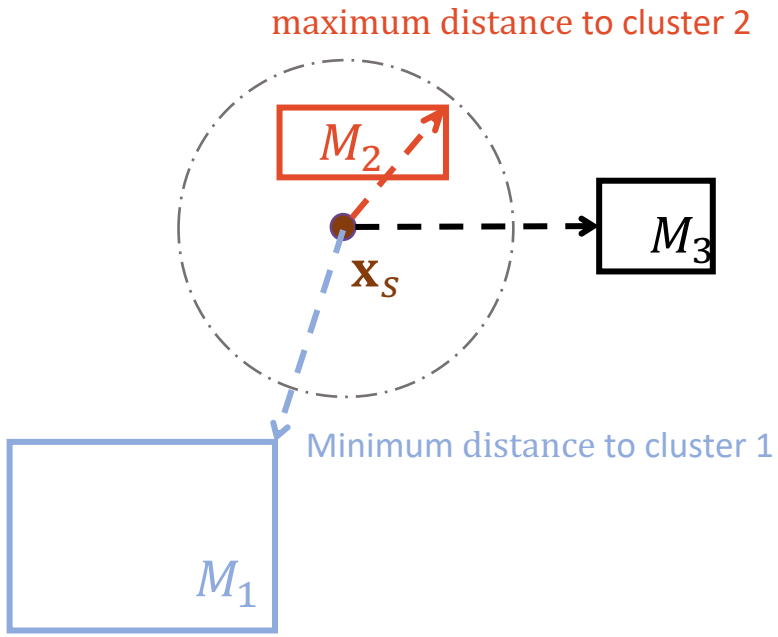
Proof: $Q_s(m) = \min_{k \in \mathcal{K}} \|x_s - m_k\|_2^{2.}$ is a continuous function on m.

Sample Determination

For each sample, if the following condition is satisfied, then the cluster of sample s can be determined and removed from the calculation.

$$\max_{m_{k^*} \in M_{k^*}} \|x_s - m_{k^*}\|^2 < \min_{m_k \in M_k} \|x_s - m_k\|^2, \forall\, k \in \mathcal{K} \backslash k^*$$

- Simplify Calculation
- Used for Bound Tightening
- Inherited to Children Nodes



Numerical Results for K-means [Hua et al. 2021]

k-means : $\min_{m} \sum_{s\in\mathcal{S}} \min_{k\in\mathcal{K}} \|x_s - m_k\|_2^2$

Heuristic Benchmark: Lloyd Algorithm [Lloyd, 1982]

Dataset	Sample	Dimension	Cores	Heuristic	UB	Nodes	Gap(%)	Time(s)
Iris	150	4	20	78.85	78.85	31	≤0.10	2,160
Glass	214	9	20	819.63	819.63	415	≤0.10	3,600
Seeds	310	7	20	587.32	587.32	89	≤0.10	3,600
Hemi	1,955	7	20	9.75E+06	9.75E+06	112	2.23	14,400
PR2392	2,392	2	20	2.97E+10	2.97E+10	247	≤0.10	3,600
Synthetic-1	42,000	2	20	5.01E+05	5.01E+05	20	3.16	14,400
Synthetic-2	210,000	2	200	2.43E+06	2.43E+06	6	2.55	14,400

Practical global optimality within reasonable time

Average improvement of global optimal value: <0.01% (10 datasets)

Solving ML problem to global optimum is possible

Solving to global optimal solution renders a much better model

Numerical Results for K-medoids [Ren et al. 2022]

k-medoids: $\min_{m \in X} \sum_{s \in S} \min_{k \in K} \|x_s - m_k\|_2^2$

Heuristic Benchmark: PAM [Kaufman et al., 1990]

Dataset	Sample	Dimension	Cores	Heuristic	UB	Nodes	Gap(%)	Time(s)
Iris	150	4	1	84.63	**83.91**	25	≤0.10	93
PR2392	2,392	2	1	2.13E+10	2.13E+10	37	≤0.10	123
UrbanGB_10	100,000	2	40	1.26E+05	**1.15E+05**	49	≤0.10	264
Rng_agr	199,843	7	1600	8.23E+14	8.23E+14	99	≤0.10	341
Spnet3D	434,874	3	1600	1.23E+08	2.28E+07	115	≤0.10	865
Retail-II	1,046,910	2	6000	2.31E+10	2.31E+10	214	≤0.10	2,515
USC1990	2,458,285	68	3000	6.91E+08	6.91E+08	25	6.33	14,400

Practical global optimality within reasonable time

Average improvement of global optimal value: 0.46% (28 datasets)

Solving ML problem to global optimum is possible

Solving to global optimal solution renders a much better model

Numerical Results for K-centers [Shi et al. 2022]

k- centers: $\min_{m \in X} \max_{s \in S} \min_{k \in K} \|x_s - m_k\|_2^2$

Heuristic Benchmark:
Farthest First Traversal [Gonzalez, 1985]

DATASET	SAMPLE	DIMENSION	CORES	HEURISTIC	UB	NODES	GAP(%)	TIME(S)
USC1990	2,458,285	68	1	2.04E+11	**1.68E+11**	1	≤0.10	277
HIGGS	11,000,000	29	400	368.35	**227.91**	12,576	30.2	14,400
BIGCROSS	11,620,300	56	400	1.43E+07	**9.38E+06**	10,071	≤0.10	6,444
TAXI	1,120,841,769	12	2000	3.09E+04	**1.62E+04**	1,063	≤0.10	5,705

Practical global optimality within reasonable time

Cost reduction: **25.8%** on average (38 datasets)

Solving ML problem to global optimum is possible

Solving to global optimal solution renders a much better model

Reduced Space Branch and Bound

Lower Bound Strategy

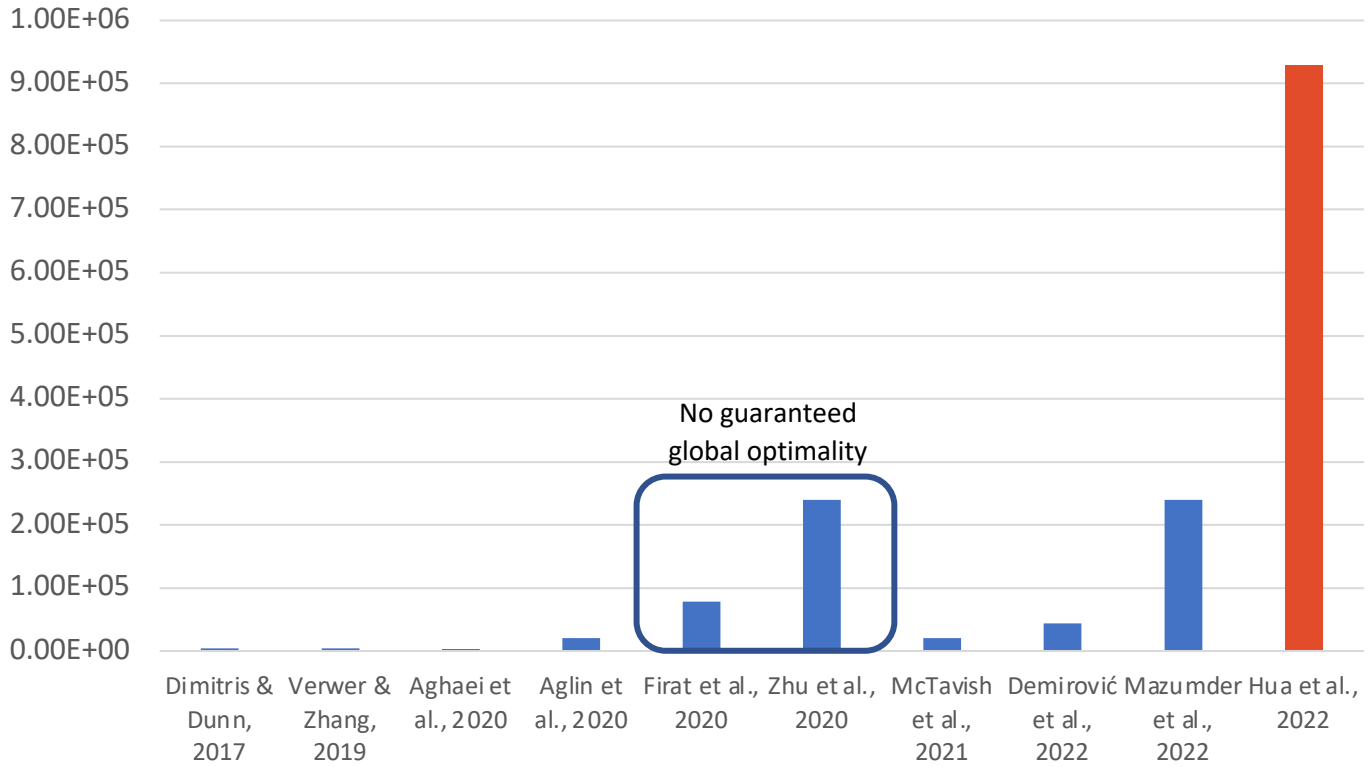
Upper Bound Strategy

Bound Tightening

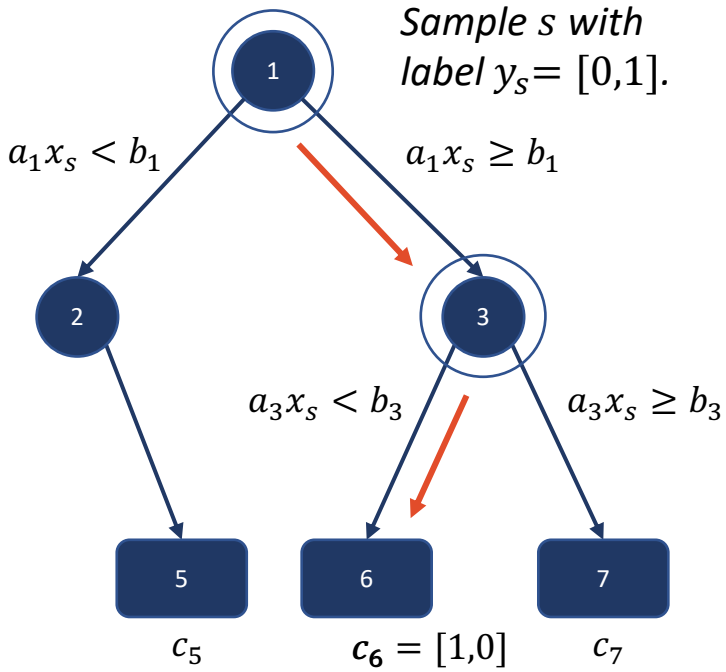
Centroid-based Clustering

Decision Tree

Decision Tree Training



Decision Tree Training



- $d_1 = d_3 = 1,\ d_2 = 0.$
- $z_{s6} = 1.$
- *Since* $c_{k6} \neq y_{sk}$, *then* $L_s = 1.$

a, b, c, d describe the tree structure:

- d_t where $t \in \mathcal{T}_D$ determines whether a decision node splits or not.
- variable $a_t = [a_{1t}, \cdots, a_{Pt}] \in \{0,1\}^P$ and $b_t \in [0,1]$ are used to track the split of decision nodes.
- The prediction of each leaf node is controlled by the class indicator $c_t \in \{0,1\}^K, \forall t \in \mathcal{T}_L$.

$z_{st} \in \{0,1\}, \forall t \in \mathcal{T}_L, s \in \mathcal{S}$ represents whether sample s falls into leaf t.

$L_s \in [0,1]$ represents the loss of sample s.

MILP Formulation

$$\min_{a,b,c,d,z,L} \sum_{s\in\mathcal{S}}\left(\frac{1}{\hat{L}}L_s + \frac{\lambda}{n}\sum_{t\in\mathcal{T}_D} d_t\right)$$

$$\text{s.t. } \frac{1}{2}\sum_{k\in\mathcal{K}}(y_{sk} + c_{kt} - 2y_{sk}c_{kt}) - L_s \le 1 - z_{st},\ \forall t \in \mathcal{T}_L$$

$$\sum_{k\in\mathcal{K}} c_{kt} = 1,\ \ \forall t \in \mathcal{T}_L$$

$$\sum_{t\in\mathcal{T}_L} z_{st} = 1$$

$$\mathbf{a}_m^T(\mathbf{x}_s + \epsilon - \epsilon_{min}) + \epsilon_{min} \le b_m + (1+\epsilon_{max})(1 - z_{st}),\ \forall m \in A_L(t), t \in \mathcal{T}_L$$

$$\mathbf{a}_m^T\mathbf{x}_s \ge b_m - (1 - z_{st}),\ \ \forall m \in A_R(t),\ t \in \mathcal{T}_L$$

$$\sum_{j=1}^{P} a_{jt} = d_t,\ \ \forall t \in \mathcal{T}_D, j \in \{1,\cdots,P\}$$

$$0 \le b_t \le d_t,\ \ \forall t \in \mathcal{T}_D$$

$$d_t \le d_{p(t)},\ \ \forall t \in \mathcal{T}_D$$

$$0 \le L_s \le 1$$

$$a_{jt}, d_t \in \{0,1\}, 0 \le b_t \le 1\ \ \forall t \in \mathcal{T}_D, j \in \{1,\cdots,P\}$$

$$z_{st}, c_{kt} \in \{0,1\},\ \ \forall t \in \mathcal{T}_L$$

$$s \in \mathcal{S}$$

- Minimize the misclassification error

MILP Formulation

$$\min_{a,b,c,d,z,L} \sum_{s\in\mathcal{S}}\left(\frac{1}{\hat{L}}L_s + \frac{\lambda}{n}\sum_{t\in\mathcal{T}_D} d_t\right)$$

$$\text{s.t. } \frac{1}{2}\sum_{k\in\mathcal{K}}(y_{sk} + c_{kt} - 2y_{sk}c_{kt}) - L_s \le 1 - z_{st},\ \forall t \in \mathcal{T}_L$$

$$\sum_{k\in\mathcal{K}} c_{kt} = 1,\ \ \forall t \in \mathcal{T}_L$$

$$\sum_{t\in\mathcal{T}_L} z_{st} = 1$$

$$\mathbf{a}_m^T(\mathbf{x}_s + \epsilon - \epsilon_{min}) + \epsilon_{min} \le b_m + (1+\epsilon_{max})(1 - z_{st}),\ \forall m \in A_L(t), t \in \mathcal{T}_L$$

$$\mathbf{a}_m^T\mathbf{x}_s \ge b_m - (1 - z_{st}),\ \ \forall m \in A_R(t),\ t \in \mathcal{T}_L$$

$$\sum_{j=1}^{P} a_{jt} = d_t,\ \ \forall t \in \mathcal{T}_D, j \in \{1,\cdots,P\}$$

$$0 \le b_t \le d_t,\ \ \forall t \in \mathcal{T}_D$$

$$d_t \le d_{p(t)},\ \ \forall t \in \mathcal{T}_D$$

$$0 \le L_s \le 1$$

$$a_{jt}, d_t \in \{0,1\}, 0 \le b_t \le 1\ \ \forall t \in \mathcal{T}_D, j \in \{1,\cdots,P\}$$

$$z_{st}, c_{kt} \in \{0,1\},\ \ \forall t \in \mathcal{T}_L$$

$$s \in \mathcal{S}$$

First Stage Variable

$m: a, b, c, d$

(Tree structure Variables, mixed integer)

Restrict each sample to be predicted follows a decision tree structure

Second Stage Variable

$z_{s,t}, L_s$

(Leaf assignment and loss, mixed integer)

Basic Lower Bound

$$\beta_S(M) \coloneqq \min_{m_S \in M} Q_S(m_S)$$

$$m \coloneqq (a, b, c, d)$$

Decompose into $|\mathcal{S}| = n$ subproblems

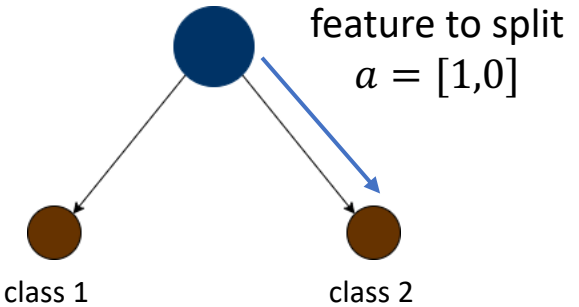
Searching for reachable leaves

Basic Lower Bound

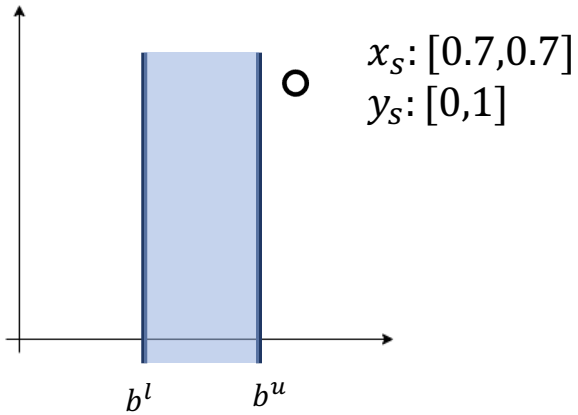
Given a branching node (in the branch and bound process) with bound M:

$$a^l = [0,0], \quad b^l = [0.3], \quad c^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^l = [1]$$

$$a^u = [1,1], \quad b^u = [0.6], \quad c^u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^u = [1]$$



$$\mathcal{T}_{z_s} = \{3\}, \beta_s = 0$$

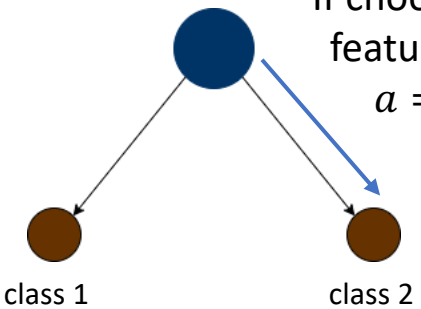


Basic Lower Bound

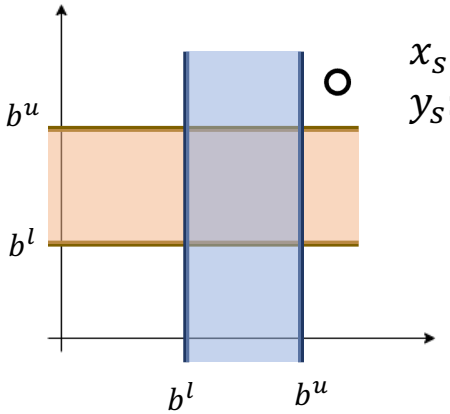
Given a branching node (in the branch and bound process) with bound M:

$$a^l = [0,0], \quad b^l = [0.3], \quad c^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^l = [1]$$

$$a^u = [1,1], \quad b^u = [0.6], \quad c^u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^u = [1]$$



If choose second feature to split $a = [0,1]$



x_s: [0.7,0.7]
y_s: [0,1]

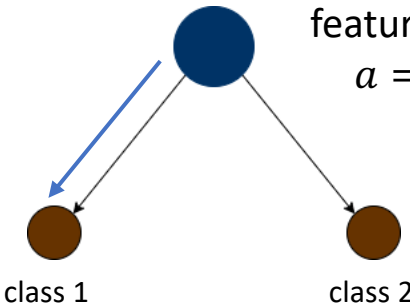
$$\mathcal{T}_{z_s} = \{3\}, \beta_s = 0$$

Basic Lower Bound

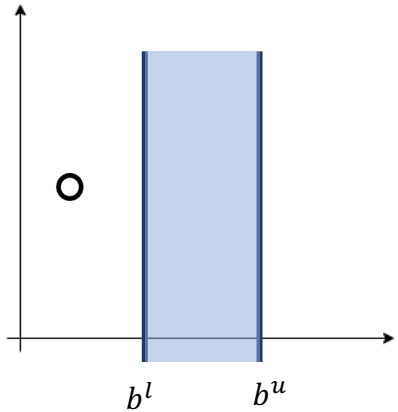
Given a branching node (in the branch and bound process) with bound M:

$$a^l = [0,0], \quad b^l = [0.3], \quad c^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^l = [1]$$

$$a^u = [1,1], \quad b^u = [0.6], \quad c^u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^u = [1]$$



x_s: $[0.2,0.5]$
y_s: $[0,1]$

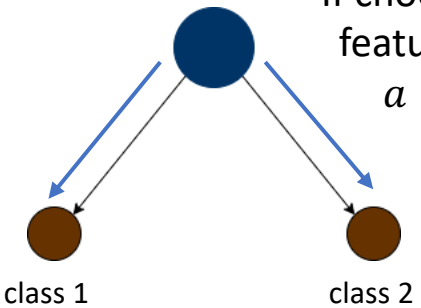


Basic Lower Bound

Given a branching node (in the branch and bound process) with bound M:

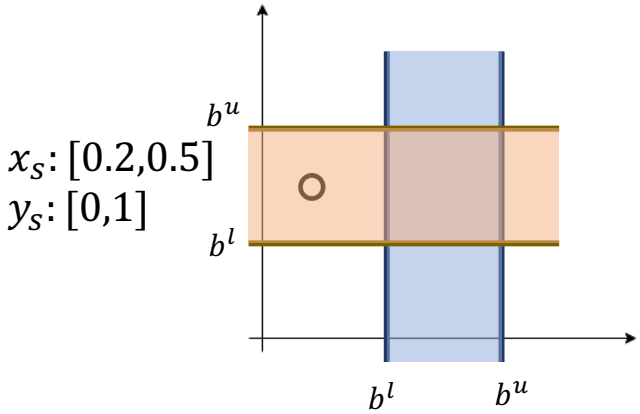
$$a^l = [0,0], \quad b^l = [0.3], \quad c^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^l = [1]$$

$$a^u = [1,1], \quad b^u = [0.6], \quad c^u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad d^u = [1]$$



If choose second feature to split $a = [0,1]$

$$\mathcal{T}_{z_s} = \{2,3\}, \beta_s = 0$$



Sample Determination

- Comparing the label y_{sk} with the range of c of each leaf node in $t \in \mathcal{T}_{z_s}$, the loss of some samples can be determined.

- Suppose k is the true label of sample s (i.e., $y_{sk} = 1$), then we can check the loss of sample s with the following equation:

$$L_s = \begin{cases} 1 & \text{if } \bigwedge_{t \in \mathcal{T}_{z_s}} c_{kt}^l = c_{kt}^u = 0 \\ 0 & \text{if } \bigwedge_{t \in \mathcal{T}_{z_s}} c_{kt}^l = c_{kt}^u = 1 \\ undtm. & \text{otherwise} \end{cases}$$

Numerical Results (D=2, nc=1000) [Hua et al. 2022]

Benchmark:
CART [Breiman et al, 1984],
OCT[Bertimas and Dunn, 2019]

Practical global optimality within reasonable time

Data-set	$\|\mathcal{S}\|$	P	K	Method	UB	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
pendigits	7,494	16	10	CART	32993.0	38.9	39.1	-	-
				OCT	32993.0	38.9	39.1	100	14400
				RS-OCT	**32426.1**	**40.0**	**39.9**	**<1%**	**2093.5**
Avila	10,430	10	12	CART	9454.3	50.4	51.0	-	-
				OCT	9196.4	51.7	52.5	100	14400
				RS-OCT	**8787.5**	**53.8**	**53.8**	**19.6**	14400
EEG	14,980	14	2	CART	7815.8	61.7	61.2	-	-
				OCT	7748.7	61.7	61.0	100	14400
				RS-OCT	**6752.7**	**66.9**	**65.5**	**50.8**	14400
HTRU	17,898	8	2	CART	327.0	97.8	97.7	-	-
				OCT	320.5	97.8	97.7	100	14400
				RS-OCT	**300.7**	**98.0**	**97.8**	**42.8**	14400
shuttle	43,500	9	7	CART	2567.5	93.8	93.8	-	-
				OCT	2567.5	93.8	93.8	100	14400
				RS-OCT	**1908.1**	**95.4**	**95.5**	**<1%**	**586.1**
Skin-Segmen-tation	245,057	3	2	CART	23409.5	89.9	89.8	-	-
				OCT	23409.5	89.9	89.8	100	14400
				RS-OCT	**16953.7**	**92.7**	**92.7**	**<1%**	**6698.9**
HT-Sensor	928,991	11	3	CART	782046.1	58.1	58.1	-	-
				OCT	782046.1	58.1	58.1	100	14400
				RS-OCT	**753075.8**	**59.7**	**59.7**	**56.2**	14400

Solving ML problem to global optimum is possible

Numerical Results (D=2, nc=1000) [Hua et al. 2022]

Benchmark:
CART [Breiman et al, 1984],
OCT[Bertimas and Dunn, 2019]

Practical global optimality within reasonable time

Cost reduction:
12.5% and **11.7%**
on average

Data-set	$\|\mathcal{S}\|$	P	K	Method	UB	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
pendigits	7,494	16	10	CART	32993.0	38.9	39.1	-	-
				OCT	32993.0	38.9	39.1	100	14400
				RS-OCT	**32426.1**	**40.0**	**39.9**	<1%	**2093.5**
Avila	10,430	10	12	CART	9454.3	50.4	51.0	-	-
				OCT	9196.4	51.7	52.5	100	14400
				RS-OCT	**8787.5**	**53.8**	**53.8**	**19.6**	14400
EEG	14,980	14	2	CART	7815.8	61.7	61.2	-	-
				OCT	7748.7	61.7	61.0	100	14400
				RS-OCT	**6752.7**	**66.9**	**65.5**	**50.8**	14400
HTRU	17,898	8	2	CART	327.0	97.8	97.7	-	-
				OCT	320.5	97.8	97.7	100	14400
				RS-OCT	**300.7**	**98.0**	**97.8**	**42.8**	14400
shuttle	43,500	9	7	CART	2567.5	93.8	93.8	-	-
				OCT	2567.5	93.8	93.8	100	14400
				RS-OCT	**1908.1**	**95.4**	**95.5**	<1%	**586.1**
Skin-Segmen-tation	245,057	3	2	CART	23409.5	89.9	89.8	-	-
				OCT	23409.5	89.9	89.8	100	14400
				RS-OCT	**16953.7**	**92.7**	**92.7**	<1%	**6698.9**
HT-Sensor	928,991	11	3	CART	782046.1	58.1	58.1	-	-
				OCT	782046.1	58.1	58.1	100	14400
				RS-OCT	**753075.8**	**59.7**	**59.7**	**56.2**	14400

Solving ML problem to global optimum is possible

Solving to global optimal solution renders a much better model

Numerical Results (D=2, nc=1000) [Hua et al. 2022]

Benchmark:
CART [Breiman et al, 1984],
OCT[Bertimas and Dunn, 2019]

Training accuracy:
3.7% and **3.6%** on average

Testing accuracy:
3.3% and **2.8%** on average

Data-set	$\|\mathcal{S}\|$	P	K	Method	UB	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
pendigits	7,494	16	10	CART	32993.0	38.9	39.1	-	-
				OCT	32993.0	38.9	39.1	100	14400
				RS-OCT	**32426.1**	**40.0**	**39.9**	**<1%**	**2093.5**
Avila	10,430	10	12	CART	9454.3	50.4	51.0	-	-
				OCT	9196.4	51.7	52.5	100	14400
				RS-OCT	**8787.5**	**53.8**	**53.8**	**19.6**	14400
EEG	14,980	14	2	CART	7815.8	61.7	61.2	-	-
				OCT	7748.7	61.7	61.0	100	14400
				RS-OCT	**6752.7**	**66.9**	**65.5**	**50.8**	14400
HTRU	17,898	8	2	CART	327.0	97.8	97.7	-	-
				OCT	320.5	97.8	97.7	100	14400
				RS-OCT	**300.7**	**98.0**	**97.8**	**42.8**	14400
shuttle	43,500	9	7	CART	2567.5	93.8	93.8	-	-
				OCT	2567.5	93.8	93.8	100	14400
				RS-OCT	**1908.1**	**95.4**	**95.5**	**<1%**	**586.1**
Skin-Segmen-tation	245,057	3	2	CART	23409.5	89.9	89.8	-	-
				OCT	23409.5	89.9	89.8	100	14400
				RS-OCT	**16953.7**	**92.7**	**92.7**	**<1%**	**6698.9**
HT-Sensor	928,991	11	3	CART	782046.1	58.1	58.1	-	-
				OCT	782046.1	58.1	58.1	100	14400
				RS-OCT	**753075.8**	**59.7**	**59.7**	**56.2**	14400

Solving ML problem to global optimum is possible

Solving to global optimal solution renders a much better model

Numerical Results of DE-MH (D=2)

Benchmark:
RS-OCT [Hua et al. 2022]

Training accuracy:
DE-MH (GPU) is 0.10% smaller than RS-OCT on average

Testing accuracy:
DE-MH (GPU) is 0.04% larger than RS-OCT on average

For datasets with ≤1% gap,
0.25% and 0.39% smaller on training and testing accuracy

Dataset	Sample	Dimension	Class	Method	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
Pen-Digits	7,494	16	10	RS-OCT	**40.0**	**39.9**	≤1%	2,094
				DE-MH (CPU)	**40.0**	39.8	-	72
				DE-MH (GPU)	**40.0**	39.8	-	13
Avila	10,430	10	12	RS-OCT	**53.8**	53.8	19.6	14,400
				DE-MH (CPU)	53.7	**53.9**	-	79
				DE-MH (GPU)	53.7	53.7	-	10
EEG	14,980	14	2	RS-OCT	66.9	65.5	50.8	14,400
				DE-MH (CPU)	66.5	65.8	-	101
				DE-MH (GPU)	**67.0**	**66.6**	-	10
HTRU	17,898	8	2	RS-OCT	**98.0**	**97.8**	42.8	14,400
				DE-MH (CPU)	97.9	97.7	-	94
				DE-MH (GPU)	97.9	97.6	-	10
Shuttle	43,500	9	7	RS-OCT	**95.4**	**95.5**	≤1%	586
				DE-MH (CPU)	94.6	94.4	-	276
				DE-MH (GPU)	94.7	94.5	-	11
Skin-Segmentation	245,057	3	2	RS-OCT	**92.7**	**92.7**	≤1%	6,699
				DE-MH (CPU)	**92.7**	**92.7**	-	973
				DE-MH (GPU)	**92.7**	**92.7**	-	12
HT-Sensor	928,991	11	3	RS-OCT	59.7	59.7	56.2	14,400
				DE-MH (CPU)	**59.8**	**59.8**	-	6,074
				DE-MH (GPU)	**59.8**	**59.8**	-	32

DE-MH (GPU) can maintain **comparable levels of accuracy** to the global method RS-OCT

Numerical Results of DE-MH (D=2)

Benchmark:
RS-OCT [Hua et al. 2022]

Running Time:
DE-MH (GPU) offers a **675X** improvement in speed over RS-OCT with 1000 cores

Dataset	Sample	Dimension	Class	Method	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
Pen-Digits	7,494	16	10	RS-OCT	**40.0**	**39.9**	≤1%	2,094
				DE-MH (CPU)	**40.0**	39.8	-	72
				DE-MH (GPU)	**40.0**	39.8	-	13
Avila	10,430	10	12	RS-OCT	**53.8**	53.8	19.6	14,400
				DE-MH (CPU)	53.7	**53.9**	-	79
				DE-MH (GPU)	53.7	53.7	-	10
EEG	14,980	14	2	RS-OCT	66.9	65.5	50.8	14,400
				DE-MH (CPU)	66.5	65.8	-	101
				DE-MH (GPU)	**67.0**	**66.6**	-	10
HTRU	17,898	8	2	RS-OCT	**98.0**	**97.8**	42.8	14,400
				DE-MH (CPU)	97.9	97.7	-	94
				DE-MH (GPU)	97.9	97.6	-	10
Shuttle	43,500	9	7	RS-OCT	**95.4**	**95.5**	≤1%	586
				DE-MH (CPU)	94.6	94.4	-	276
				DE-MH (GPU)	94.7	94.5	-	11
Skin-Segmentation	245,057	3	2	RS-OCT	**92.7**	**92.7**	≤1%	6,699
				DE-MH (CPU)	**92.7**	**92.7**	-	973
				DE-MH (GPU)	**92.7**	**92.7**	-	12
HT-Sensor	928,991	11	3	RS-OCT	59.7	59.7	56.2	14,400
				DE-MH (CPU)	**59.8**	**59.8**	-	6,074
				DE-MH (GPU)	**59.8**	**59.8**	-	32

The metaheuristic method obtains **comparable levels of accuracy** to the global method with **significantly reduced running time**.

Numerical Results of DE-MH (D=3)

Benchmark:
CART [Breiman et al, 1984],
RS-OCT [Hua et al. 2022]

DE-MH v.s. CART
3.36% and 3.26% larger on training and testing accuracy

DE-MH v.s. RS-OCT
2.52% and 2.75% larger on training and testing accuracy
(RS-OCT fails to achieve reasonable gaps)

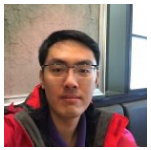
Dataset	Sample	Dimension	Class	Method	Training Accuracy(%)	Testing Accuracy(%)	Gap (%)	Time (s)
Pen-Digits	7,494	16	10	CART	62.5	63.0	-	0.01
				RS-OCT	63.9	63.7	46.7	14,400
				DE-MH (GPU)	**66.9**	**66.8**	-	32
Avila	10,430	10	12	CART	54.4	54.6	-	0.02
				RS-OCT	55.0	55.3	91.1	14,400
				DE-MH (GPU)	**57.2**	**57.4**	-	28
EEG	14,980	14	2	CART	66.6	65.2	-	0.02
				RS-OCT	67.9	65.7	100	14,400
				DE-MH (GPU)	**70.4**	**69.2**	-	27
HTRU	17,898	8	2	CART	97.9	97.7	-	0.04
				RS-OCT	97.9	97.7	99.9	14,400
				DE-MH (GPU)	**98.0**	**97.8**	-	21
Shuttle	43,500	9	7	CART	99.7	99.6	-	0.03
				RS-OCT	99.7	**99.8**	84.9	14,400
				DE-MH (GPU)	**99.8**	**99.8**	-	30
Skin-Segmen-tation	245,057	3	2	CART	96.5	96.5	-	0.07
				RS-OCT	96.6	96.5	80.3	14,400
				DE-MH (GPU)	**96.7**	**96.8**	-	29
HT-Sensor	928,991	11	3	CART	64.5	64.5	-	2.71
				RS-OCT	64.6	64.5	100	14,400
				DE-MH (GPU)	**67.7**	**67.7**	-	65

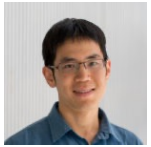
Conclusion and Highlights

- A reduced-space branch and bound training algorithm.
  - Guaranteed convergence by branching only on model parameter variables.
- Decomposable lower and upper bound strategies.
  - Trivial Parallelism with closed-form subproblem solutions.
- Achieve global optimum solutions for large-scale ML problems.
  - One billion samples within 2 hours (0.1%, k-center problem, 1000 cores).
- Global Optimization vs Heuristics

Acknowledge

Postdoc


Kaixun Hua


Kevin Kung


Yan Yu


Vijay Pediredla

Phd


Yixiu Wang


Jiayang Ren


Chaojie Ji


Qiangqiang Mao


Ghazaleh Mozafari


Mohammad Aliahmadi


Liang Cao


Jingyi Wang


Ahmed Al-Shafei

Master


Morimasa Okamoto


Mingfei Shi


Jeff MacDonald

Thank You!