Combinatorial Optimization and Interpretable Machine Learning

Thibaut Vidal

SCALE-AI Chair in Data-Driven Supply Chains, MAGI, Polytechnique Montreal, Canada
Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Montreal, Canada
Department of Computer Science, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil

Image Credit: Hitchhikers Guide to the Galaxy

Seminar DAAO, GDR-RO | 06/21/2021

Recent applications for machine learning

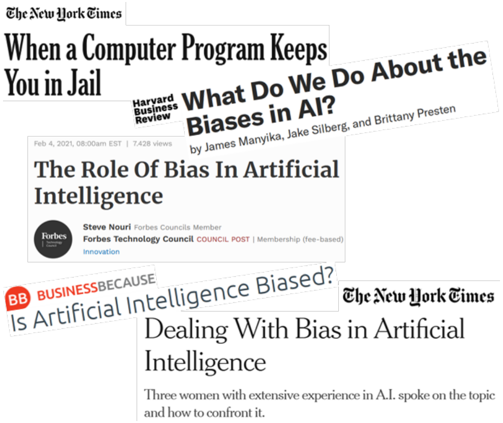
- Machine learning seeps into more and more industries and applications
  - Recommendation systems
  - Computer vision
  - Text editing and translation...

Great!

- remarkably, especially into high stakes decisions:
  - Recurrence predictions in medicine
  - Credit default risk evaluations
  - Bail decisions in criminal justice...

Wait! Should it be applied here in the first place?

and their loopholes...



The New York Times
When a Computer Program Keeps You in Jail

Harvard Business Review
What Do We Do About the Biases in AI?
by James Manyika, Jake Silberg, and Brittany Presten

Feb 4, 2021, 08:00am EST | 7,428 views
The Role Of Bias In Artificial Intelligence
Steve Nouri Forbes Councils Member
Forbes Technology Council COUNCIL POST | Membership (fee-based)
Innovation

BB BUSINESSBECAUSE
Is Artificial Intelligence Biased?

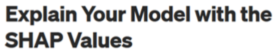
The New York Times
Dealing With Bias in Artificial Intelligence
Three women with extensive experience in A.I. spoke on the topic and how to confront it.

Humans can be biased...

- ... and so can be AI (algorithmic bias)
  - lack of fairness
  - lack of justice, i.e., presence of prejudices
- algorithmic bias can take different forms (e.g., gender bias, age discrimination, racial prejudice)
  - even if we exclude sensitive variables (!)
- we need to understand the root cause of algorithmic bias
  - there is rarely a way to "opt out" of the influence of these algorithms
  - this endangers especially marginalized groups of people

Recent trends

- the ML community's interest to understand the decision taking of ML algorithms is increasing
- explainability has gone a long way from early methods strongly focused on feature importance on top of a black-box...



Explain Your Model with the SHAP Values

- ...to modern studies and applications focusing on training transparent and interpretable models.

nature machine intelligence

Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead

A first step: developing a proper understanding of how AI algorithms work

Transparency, Explainability, and Interpretability

Explainability $\neq$ Interpretability

Interpretability

The decision process of an **interpretable model** can be understood **by design**, e.g., it is possible to track the individual decisions taken in a decision tree. This implies that interpretable models are also **transparent**.

Explainability

The internal process of an **explainable model** does not need to be transparent (it can be a black box), but an **additional algorithm** should be available to explain the decisions taken (e.g., saliency maps, feature relevance analysis).

Aims and Scope

Scope of this talk

- A **brief survey of methods for interpretable and explainable machine learning**, discussing some of their advantages and disadvantages
- Second, pinpoint specific tasks for which optimization represents a promising asset, connecting them with some of our recent works:
  - **1) Model Compression**
    Vidal, T., Schiffer, M. (2020). *Born-Again Tree Ensembles. ICML'20.* (`https://arxiv.org/abs/2003.11132`)
  - **2) Counterfactual Search**
    Parmentier, A., Vidal, T. *Optimal Counterfactual Explanations in Tree Ensembles. ICML'21.* (`https://arxiv.org/abs/2106.06631`)
  - **3) Training Sparse Models**
    Martins, P., Schiffer, M., Serra, T. Vidal, T. *Optimal Decision Diagrams for Classification.* (Submitted – ArXiV report soon available).

Before starting: some terminology

Notations and Basic Concepts

- Considering a data set $(\boldsymbol{X}, \boldsymbol{y}) = \{x_i, y_i\}_{i=1}^{n}$
  - with $x_i \in \mathbb{R}^p$ being a p-dimensional feature vector
  - and $y_i \in \mathbb{N}$ being its associated target
- Each sample i has been independently drawn from an unknown distribution $(\mathcal{X}, \mathcal{Y})$
- We aim to learn a function (i.e., classifier or predictor) $F : \mathcal{X} \rightarrow \mathcal{Y}$
  - that predicts y_i for each x_i drawn from $\mathcal{X}$

Brief Tour d'Horizon

Part I—Interpretable Models

Linear Regression

Linear regression

- Learn a linear relationship $y = \beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n + \epsilon$
  - e.g., by minimizing the squared deviation between the actual and estimates outcomes

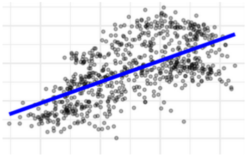
$$\hat{\boldsymbol{\beta}} = \arg\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y^{(i)} - \left(\beta_0 + \sum_{j=1}^{n} \beta_0 + \sum_{j=1}^{p} \beta_j x_j^{(i)} \right) \right)$$

- Easy to understand and interpretable but has several key assumptions and limitations
  - e.g., linearity, normality, homoscedasticity...
  - adequate for regression, but not much for classification.
- In cases with multicollinearity, the interpretation of weights can be counter-intuitive
  - two highly correlated features can receive very different weights (e.g., first feature can be used by the model as a main effect, and the second intervenes to “correct” the prediction).
- Extension towards Generalized Linear Models (GLMs) of the form $g(E_Y(y|x)) = \beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n$ permits to circumvent some of these issues.

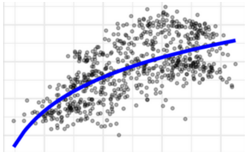
Generalized Linear Models & Generalized Additive Models

- Other ways to deal with non-linear relationships...

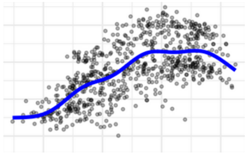
Linear model



Feature transformation



GAM



picture credit: [2]

Generalized additive models (GAMs)

- Idea: instead of a simple weighted sum, we consider a sum of arbitrary functions (one for each feature)

$$y = \beta_0 + f_1(x_1) + \ldots + f_n(x_n)$$

- we can use spline functions to approximate such nonlinearities

Support Vector Machines

Support Vector Machines

- Originally designed for classification. Training a SVM consists in best separating two classes of samples by a hyperplane defined by points satisfying $\boldsymbol{w}^T\boldsymbol{x} - b = 0$
  - A regularization term is added on the coefficients of $\boldsymbol{w}$, implicitly maximizing the separation margin between the points.
  - Points within the margin or on the wrong side are linearly penalized relatively to their distance.
- SVM training can be cast as a linear program:

$$\min_{\boldsymbol{w},b,\xi} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n}\xi_i \tag{1}$$

$$\text{s.t.} \quad y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad i \in \{1,\dots,n\} \tag{2}$$

$$\xi_i \geq 0, \quad i \in \{1,\dots,n\} \tag{3}$$

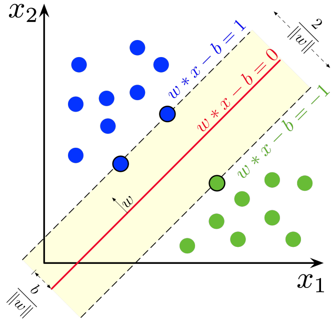
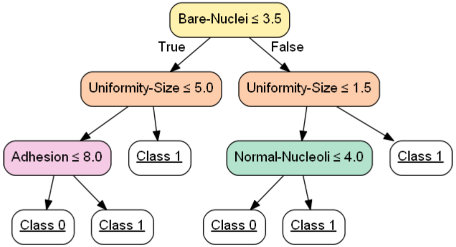


Figure: Credit to Larhmam, CC BY-SA 4.0, via Wikimedia Commons

Decision Trees

Decision Trees

- Decision-tree classifiers consist in a hierarchical tree structure in which each internal node represents a linear-inequality on a feature, and each leaf is associated to a class.
- Any sample starts at the root node, and descends along the tree in accordance with the status of the inequalities, until it reaches a class node.
- This process is usually viewed as **interpretable**, since the trajectory of the samples through the tree and the features which led to the final prediction are directly observable.
- Generalization into *tree ensembles* often improves performance, but drastically diminishes interpretability



Advantages and Disadvantages

The models reviewed in this section...

+ are transparent and interpretable by design
+ permit to comprehend some of the interaction between the features
+ provide natural visualizations

– regularly have a worse performance than sophisticated models (e.g., tree ensembles, neural networks)
– depend on key assumptions about the data
– may lack stability and robustness

Brief Tour d’Horizon

Part II—Explanation Methods

Sophisticated or Black-Box Models are sometimes needed

Ensembles, deep neural networks and cie...

- Sophisticated models are sometimes needed for advanced machine learning tasks and complex data sets.
- There are also situations where the owner of the model does not want transparency, i.e., refuses to reveal the internal decision process (e.g., for applications in security, or simply to avoid reverse-engineering)
- In this case, **explanation methods** are needed, which often consist of a second algorithmic layer on top of the original model.
  - Depending on the class of model and the knowledge of the model (black-box vs white-box), different explanation techniques can be employed.

Feature Importance

Permutation Feature Importance

- Perhaps the simplest explanation method consists in measuring the impact of each of the features of the model, i.e., determining which features most "contributed" to the classification.
- A simple empirical approach, called **permutation feature importance** consists in considering each feature, randomly *shuffling* it's values among the training samples (leading to a loss of information), and measuring the impact of this shuffle on prediction performance.
  - No transparency initially required, simulating the prediction function is often enough.
  - Features that significantly impact the classification performance are "important"
  - Useful to detect the possible use of protected features
  - Weak to correlations
  - Importance is not always meaningful as a individual measure

Shapley Value based approaches

SHAP – SHapley Additive exPlanations [1]

- Shapley values come from cooperative game theory, where they represent by how much each player contributes to the surplus of a coalition in a cooperative game
  - **Analogy**: assume that each feature is a player and that the prediction is the surplus
- SHAP computes the contribution of each feature for a specific instance
  - the Shapley value explanation in SHAP is a linear model

$$g(\boldsymbol{z}) = \phi_0 + \sum_{j=1}^{M} \phi_j z_j$$

    with g–explanation model, $\boldsymbol{z} \in \{0,1\}^M$–coalition vector (sometimes also called "simplified features"), $\phi_j \in \mathbb{R}$–feature attribution (i.e., Shapley values)
  - the model satisfies additivity and consistency properties, i.e., if the model changes and the marginal contribution of a feature increases or is equal, then the Shapley value increases or remains equal

Global Surrogate Models

Global Surrogate Models

- Train an interpretable model to *imitate* the predictor of a complex or opaque model.
- Simple approaches in this family consist in sampling training examples from the original data set (or even outside of it) and obtaining the corresponding prediction as given by the original model. This data set is then used to train an interpretable model.
    - However, unless the data set permits to infer the complete behavior of the original model, such an approach does not provide guarantees on the faithfulness of the surrogate model.
- For transparent classification models such as random forests, mathematical techniques ([6] – see next part of this talk) can be used to generate a surrogate decision tree model that has the same prediction function everywhere.

Local Surrogate models

LIME – Local Interpretable Model-agnostic Explanations [4]

- LIME: Focuses on training a surrogate models to explain individual predictions
- Select an instance which needs to be explained
- Create a data set by adding random perturbations to the point of interest, and obtain the predicted class from the original model
- Weight the new data set based on proximity to the original instance
- Train an interpretable surrogate model on this data, and use it for explanations and visualizations.

Counterfactual Explanations

Counterfactual Explanations [7]

- Counterfactual Explanations are constrastive arguments of the type:
  "To obtain this loan, you need \$40,000 of annual revenue instead of the current \$30,000".
- **Rationale**: Smallest set of actions needed to shift the outcome of a classification task. Gives *recourse* and can play a critical role in a negotiation process.
- Given an origin point $\hat{\boldsymbol{x}}$ and a desired prediction class c^*, locate a new data point $\boldsymbol{x} \in \mathcal{X}$ that solves the following problem:

$$\begin{aligned} \min \quad & D(\boldsymbol{x}, \hat{\boldsymbol{x}}) \\ \text{s.t.} \quad & F(\boldsymbol{x}) = c^* \\ & \boldsymbol{x} \in X \end{aligned}$$

- Note the close connections with *local explanations*, *adversarial search* and *robustness evaluation*.
- Solution strategy for this search problem will largely depend on the characteristics of $F(\boldsymbol{x})$ and the transparency of the model. It can consist of a simple gradient descent (for convex and differentiable predictors) or require much more advanced techniques in other cases [3].

Advantages and Disadvantages

Explanation Methods

+ allow to explain the outcome of a black box model without imposing any limitations to the model itself
  - ▶ we can preserve the good performance of black box models
+ usually model agnostic (LIME, SHAP, ...)
- Explainable AI techniques pose one issue: in case of an unexpected behavior or bias, two algorithms/models may need debugging instead of one [5].
- Explainable AI techniques allow to understand a specific outcome of a model but not to develop a general understanding of the models rules and decision process
  - ▶ In general, we are still looking at a black box and try to explain case by case what happens

Brief Tour d'Horizon

Part III—Promising research paths for combinatorial optimization and interpretable machine learning

[Next set of slides]

Bibliography I

[1] S. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.

[2] C. Molnar. *Interpretable Machine Learning*. 2019. `https://christophm.github.io/interpretable-ml-book/`.

[3] A. Parmentier and T. Vidal. Optimal counterfactual explanations in tree ensembles. In *38th International Conference on Machine Learning, PMLR 139*, 2021.

[4] M. Ribeiro, S. Singh, and C. Guestrin. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining – KDD '16*, pages 1135–1144, 2016.

[5] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[6] T. Vidal and M. Schiffer. Born-again tree ensembles. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 9743–9753, Virtual, 2020. PMLR.

[7] S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31:841, 2018.

1) Born-Again Tree Ensembles

Thibaut Vidal[1,2], Toni Pacheco[2], Maximilian Schiffer[3]

[1] CIRRELT & SCALE-AI Chair in Data-Driven Supply Chains, MAGI, Polytechnique Montreal, Canada
[2] Computer Science Department, Pontifical Catholic University of Rio de Janeiro
[3] TUM School of Management, Technical University of Munich

Image Credit: Hitchhikers Guide to the Galaxy

Reference: Vidal, T., & Schiffer, M. (2020). Born-Again Tree Ensembles. In Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119.

Decision Trees and Random Forests

Decision tree:

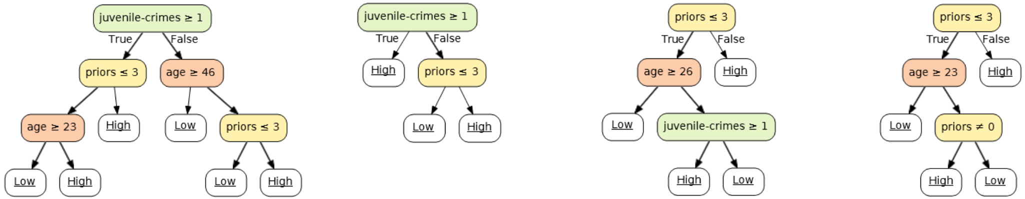
++ Simple and explainable

– – Possible overfit & typically lower accuracy on test data

Tree ensemble – Random forest:

++ Ensemble learning algorithm: **better generalization** on test data

– – **Lack of interpretability**



Related Literature

Thinning tree ensembles

Pruning some weak learners [16, 20, 22, 25]

Replacing the tree ensemble by a simpler classifier [1, 6, 17, 23]

Rule extraction via bayesian model selection [13]

Extracting a single tree from a tree ensemble by actively sampling training points [2, 3]

Thinning neural networks

Model compression and knowledge distillation [7, 14]: Using a "teacher" to train a compact "student' with similar knowledge.

Creating soft decision trees from a neural network [10], or decomposing the gradient in knowledge distillation [11].

Simplifying neural networks [8, 9, 21].

Optimal decision trees

Linear programming algorithms have been exploited to find linear combination splits [4].

Extensive study of global optimization methods, based on mixed-integer programming or dynamic programming, for the construction of optimal decision trees [5, 12, 15, 18, 24]

- Other, model-agnostic, explanation approaches such as LIME [19].
  - $\Rightarrow$ Aimed at providing a **local** explanation.
  - $\Rightarrow$ Works by training a simpler surrogate model (e.g., a linear classifier) around an instance that should be explained and analyze the weights.

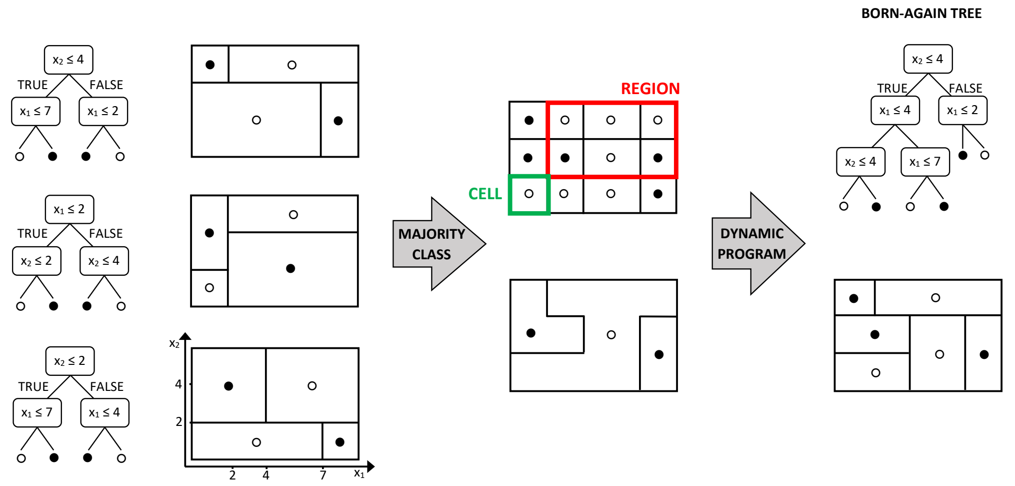
Born-Again Tree Ensembles

- A recent exact algorithm that transforms a tree ensemble into a born-again decision tree (BA tree) that is:
  - **Optimal** in size (number of leaves or depth), and
  - **Faithful** to the tree ensemble **in its entire feature space**.
- The BA tree is effectively **a different representation of the same decision function**.

A single —minimal-size— decision tree that faithfully reproduces the decision function of the random forest.

Methodology

Construction Process



Methodology

Problem 1: Born-Again Tree Ensemble

Given a tree ensemble $\mathcal{T}$, we search for a decision tree T of **minimal size** such that $F_T(\mathbf{x}) = F_{\mathcal{T}}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^p$.

Theorem 1

Problem 1 is NP-hard when optimizing depth, number of leaves, or any hierarchy of these two objectives.

Verifying that a given solution is feasible (faithful) is NP-hard.

Methodology

Dynamic Program 1

Let $\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}})$ be the depth of an optimal born-again decision tree for a region $(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}})$. Then:

$$\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) = \begin{cases} 0 \quad \text{if} \quad \textsc{id}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) \\ \min\limits_{1 \leq j \leq p} \left\{ \min\limits_{z_j^{\text{L}} \leq l < z_j^{\text{R}}} \left\{ 1 + \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}_{jl}^{\text{R}}), \Phi(\mathbf{z}_{jl}^{\text{L}}, \mathbf{z}^{\text{R}})\} \right\} \right\} \end{cases},$$

in which $\textsc{id}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}})$ takes value TRUE iff all cells $\mathbf{z}$ such that $\mathbf{z}^{\text{L}} \leq \mathbf{z} \leq \mathbf{z}^{\text{R}}$ are from the same class (i.e. base case).

Issue 1

Detecting base cases

Issue 2

Numerous recursive calls

Circumventing Issue 1

We tried several alternatives to efficiently check base cases. The best approach we found consisted in including the base case evaluation within the DP:

Dynamic Program 2

Let $\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}})$ be the depth of an optimal born-again decision tree for a region $(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}})$. Then:

$$\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) = \min_{1 \le j \le p} \left\{ \min_{z_j^{\text{L}} \le l < z_j^{\text{R}}} \{ \mathbb{1}_{jl}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) + \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}), \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}})\} \} \right\}$$

$$\text{where } \mathbb{1}_{jl}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) = \begin{cases} 0 & \text{if} \quad \Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}) = \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}}) = 0 \\ & \text{and} \;\; F_{\mathcal{T}}(\mathbf{z}^{\text{L}}) = F_{\mathcal{T}}(\mathbf{z}^{\text{R}}); \\ 1 & \text{otherwise.} \end{cases}$$

Circumventing Issue 2

We exploit two simple properties to reduce the number of recursive calls:

Property 2

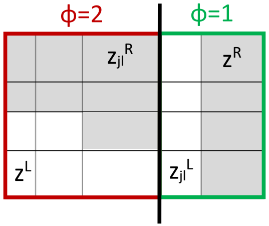
If $\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}) \geq \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}})$ then for all $l' > l$:

$$\begin{aligned}\mathbb{1}_{jl}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) &+ \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}), \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}})\} \\ &\leq \mathbb{1}_{jl'}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) + \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl'}), \Phi(\mathbf{z}^{\text{L}}_{jl'}, \mathbf{z}^{\text{R}})\}\end{aligned}$$

Property 3

If $\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}) \leq \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}})$ then for all $l' < l$:

$$\begin{aligned}\mathbb{1}_{jl}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) &+ \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl}), \Phi(\mathbf{z}^{\text{L}}_{jl}, \mathbf{z}^{\text{R}})\} \\ &\leq \mathbb{1}_{jl'}(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}) + \max\{\Phi(\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}_{jl'}), \Phi(\mathbf{z}^{\text{L}}_{jl'}, \mathbf{z}^{\text{R}})\}\end{aligned}$$



Allowing us to search for the best hyperplane level for each feature with a binary search.

Algorithm 1 Born-Again($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$)

```
1: if ($\mathbf{z}^{\text{L}} = \mathbf{z}^{\text{R}}$) return 0
2: if ($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$) exists in memory return Memory($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$)
3: $(\text{LB}, \text{UB}) \leftarrow (0, \infty)$
4: for $j = 1$ to $p$ and $\text{LB} < \text{UB}$ do
5:     (Low, Up) $\leftarrow (z_j^{\text{L}}, z_j^{\text{R}})$
6:     while Low < Up and $\text{LB} < \text{UB}$ do
7:         $l \leftarrow \lfloor(\text{Low} + \text{Up})/2\rfloor$
8:         $\Phi_1 \leftarrow$ Born-Again($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}} + \mathbf{e}_j(l - z_j^{\text{R}})$)
9:         $\Phi_2 \leftarrow$ Born-Again($\mathbf{z}^{\text{L}} + \mathbf{e}_j(l + 1 - z_j^{\text{L}}), \mathbf{z}^{\text{R}}$)
10:        if ($\Phi_1 = 0$) and ($\Phi_2 = 0$) then
11:            if $f(\mathbf{z}^{\text{L}}, \mathcal{T}) = f(\mathbf{z}^{\text{R}}, \mathcal{T})$ then Memorize(($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$), 0) and return 0
12:            else Memorize(($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$), 1) and return 1
13:        end if
14:        $\text{UB} \leftarrow \min\{\text{UB}, 1 + \max\{\Phi_1, \Phi_2\}\}$
15:        $\text{LB} \leftarrow \max\{\text{LB}, \max\{\Phi_1, \Phi_2\}\}$
16:        if ($\Phi_1 \geq \Phi_2$) then Up $\leftarrow l$
17:        if ($\Phi_1 \leq \Phi_2$) then Low $\leftarrow l + 1$
18:    end while
19: end for
20: Memorize(($\mathbf{z}^{\text{L}}, \mathbf{z}^{\text{R}}$), UB) and return UB
```

Experimental Analyses

Goals

Evaluate the scalability of the DP algorithm depending on:

- the size metric in use
- the number of trees in the ensemble
- the number of samples and features in the datasets

Study the structure and complexity of the born-again trees for different size metrics.
Measure the impact of an a-posteriori pruning strategy.

Experimental Analyses

Datasets

We used datasets from diverse applications, including medicine (BC, PD), criminal justice (COMPAS), and credit scoring (FICO).

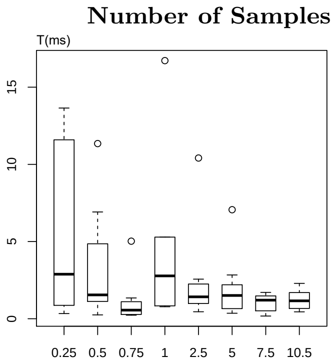
Data set	n	p	K	CD	Src.
BC – Breast-Cancer	683	9	2	65-35	UCI
CP – COMPAS	6907	12	2	54-46	HuEtAl
FI – FICO	10459	17	2	52-48	HuEtAl
HT – HTRU2	17898	8	2	91-9	UCI
PD – Pima-Diabetes	768	8	2	65-35	SmithEtAl
SE – Seeds	210	7	3	33-33-33	UCI

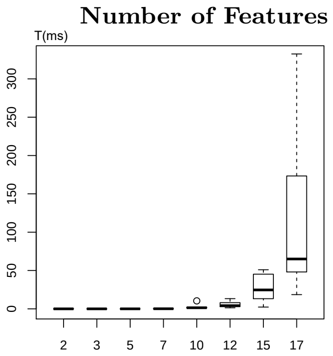
Data Preparation

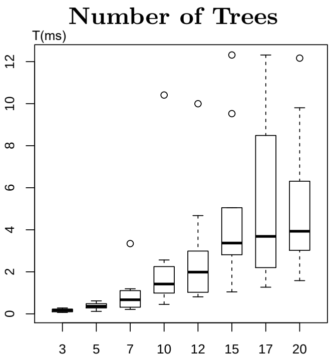
One-hot encoding for categorical variables.
Continuous variables binned into ten ordinal scales.
Generate training and test samples for all data sets by ten-fold cross validation. For each fold and each dataset, generate a random forest composed of 10 trees with a depth of 3.

Experimental Analyses

Scalability







Computational time(ms) of the DP as a function of the number of samples, features and trees.

Experimental Analyses

Simplicity

Depth and number of leaves of the born-again trees:

	D		L		DL	
Data set	Depth	# Leaves	Depth	# Leaves	Depth	# Leaves
BC	12.5	2279.4	18.0	890.1	12.5	1042.3
CP	8.9	119.9	8.9	37.1	8.9	37.1
FI	8.6	71.3	8.6	39.2	8.6	39.2
HT	6.0	20.2	6.3	11.9	6.0	12.0
PD	9.6	460.1	15.0	169.7	9.6	206.7
SE	10.2	450.9	13.8	214.6	10.2	261.0
Avg.	9.3	567.0	11.8	227.1	9.3	266.4

Analysis

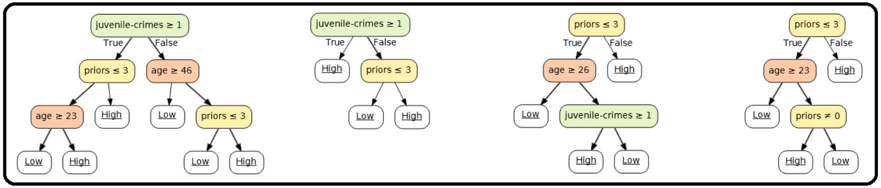
The decision function of a random forest is visibly complex
One main reason: *Incompatible feature combinations* are being represented, and the decision function of the RF is not necessarily uniform on these regions due to the other features.

Post-Pruning

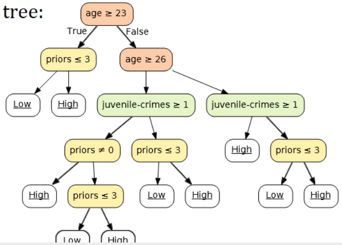
Eliminate inexpressive tree sub-regions. From bottom to top:

- Verify whether both sides of a split contain at least one sample, and eliminate every such *empty* split

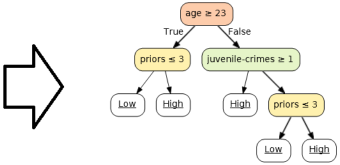
Initial tree ensemble with T=4 trees:



Born-again tree:



After pruning:



Experimental Analyses

Analysis

With post-pruning, faithfulness is no longer guaranteed per definition. We need to experimentally evaluate:

- Impact on simplicity
- Impact on accuracy

Depth and number of leaves:

	RF	BA-Tree		BA+P	
	Leaves	Depth	Leaves	Depth	Leaves
BC	61.1	12.5	2279.4	9.1	35.9
CP	46.7	8.9	119.9	7.0	31.2
FI	47.3	8.6	71.3	6.5	15.8
HT	42.6	6.0	20.2	5.1	13.2
PD	53.7	9.6	460.1	9.4	79.0
SE	55.7	10.2	450.9	7.5	21.5
Avg.	51.2	9.3	567.0	7.4	32.8

Accuracy and F1 score comparison:

	RF		BA-Tree		BA+P	
	Acc	F1	Acc	F1	Acc	F1
BC	0.953	0.949	0.953	0.949	0.946	0.941
CP	0.660	0.650	0.660	0.650	0.660	0.650
FI	0.697	0.690	0.697	0.690	0.697	0.690
HT	0.977	0.909	0.977	0.909	0.977	0.909
PD	0.746	0.692	0.746	0.692	0.750	0.700
SE	0.790	0.479	0.790	0.479	0.790	0.481
Avg.	0.804	0.728	0.804	0.728	0.803	0.729

Heuristic Solutions

The current DP approach can be applied to datasets with up to 20 features in our experiments. To solve larger cases we introduced a heuristic **that guarantees faithfulness, but relaxes optimality**.

- Instead of opening all recursions, it uses a greedy split criterion (information gain) considering $n_c = 100$ random cells within the region.
- If the n_c cells belong to the same class, it uses a resource-constrained shortest path bound to attempt to prove that all cells within this region belong to the same class.
- If this bound is insufficient, a MIP is used to prove uniformity or detect a violating cell.

This heuristic finds faithful BA-trees for large datasets (Ionosphere, Spambase, and Miniboone, the later with over 130,000 samples and 50 features) in less than 30 seconds.
The depth and number of leaves increases by 22.90% and 18.20% on average over the optimal solutions, but the heuristic solutions usually give good trade-offs.

Conclusions

- BA-trees provide a compact representations of the decision functions of random forests, as a single —minimal size— decision tree.
- Sheds a new light on **random forests visualization and interpretability**.
- Progressing towards interpretable models is an important step towards addressing bias and data mistakes in learning algorithms.
- Optimal classifiers can be fairly complex. Indeed, BA-trees reproduce the complete decision function for *all regions of the feature space.*
  - Pruning can solve this issue
  - Heuristics can be used for datasets which are too large to be solved to optimality

To be continued...

Controlled relaxation of faithfulness, Better optimization algorithms, Gradient boosting, Extension to Regression, Local Explainability, Case studies...

Bibliography I

[1] Bai, J., Y. Li, J. Li, Y. Jiang, S. Xia. 2019. Rectified decision trees: Towards interpretability, compression and empirical soundness. *arXiv preprint arXiv:1903.05965* .

[2] Bastani, O., C. Kim, H. Bastani. 2017. Interpretability via model extraction. *arXiv preprint arXiv:1706.09773* .

[3] Bastani, O., C. Kim, H. Bastani. 2017. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504* .

[4] Bennett, K. 1992. Decision tree construction via linear programming. *Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society Conference, Utica, Illinois*.

[5] Bertsimas, D., J. Dunn. 2017. Optimal classification trees. *Machine Learning* **106**(7) 1039–1082.

[6] Breiman, L., N. Shang. 1996. Born again trees. Tech. rep., University of California Berkeley.

[7] Buciluǎ, C., R. Caruana, A. Niculescu-Mizil. 2006. Model compression. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[8] Clark, K., M.-T. Luong, U. Khandelwal, C. D. Manning, Q. V. Le. 2019. Bam! born-again multi-task networks for natural language understanding. *arXiv preprint arXiv:1907.04829* .

[9] Frankle, J., M. Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* .

[10] Frosst, N., G. Hinton. 2017. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784* .

[11] Furlanello, Tommaso, Zachary C Lipton, Michael Tschannen, Laurent Itti, Anima Anandkumar. 2018. Born again neural networks. *arXiv preprint arXiv:1805.04770* .

[12] Günlük, O., J. Kalagnanam, M. Menickelly, K. Scheinberg. 2018. Optimal decision trees for categorical data via integer programming. *arXiv preprint arXiv:1612.03225* .

Bibliography II

[13] Hara, S., K. Hayashi. 2016. Making tree ensembles interpretable: A bayesian model selection approach. *arXiv preprint arXiv:1606.09066* .

[14] Hinton, G., O. Vinyals, J. Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

[15] Hu, X., C. Rudin, M. Seltzer. 2019. Optimal sparse decision trees. *Advances in Neural Information Processing Systems*.

[16] Margineantu, D., T. Dietterich. 1997. Pruning adaptive boosting. *Proceedings of the Fourteenth International Conference Machine Learning*.

[17] Meinshausen, N. 2010. Node harvest. *The Annals of Applied Statistics* 2049–2072.

[18] Nijssen, S., E. Fromont. 2007. Mining optimal decision trees from itemset lattices. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[19] Ribeiro, M.T., S. Singh, C. Guestrin. 2016. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining – KDD '16*. 1135–1144.

[20] Rokach, L. 2016. Decision forest: Twenty years of research. *Information Fusion* **27** 111–125.

[21] Serra, T., A. Kumar, S. Ramalingam. 2020. Lossless compression of deep neural networks. *CPAIOR 2020: Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. 417–430.

[22] Tamon, C., J. Xiang. 2000. On the boosting pruning problem. *Proceedings of the 11th European Conference on Machine Learning*.

[23] Tan, H. F., G. Hooker, M. T. Wells. 2016. Tree space prototypes: Another look at making tree ensembles interpretable. *arXiv preprint arXiv:1611.07115* .

Bibliography III

[24] Verwer, S., Y. Zhang. 2019. Learning optimal classification trees using a binary linear program formulation. *Proceedings of the AAAI Conference on Artificial Intelligence*.

[25] Zhang, Y., S. Burer, W. N. Street. 2006. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research* **7**(Jul) 1315–1338.

2) Optimal Counterfactual Explanations in Tree Ensembles

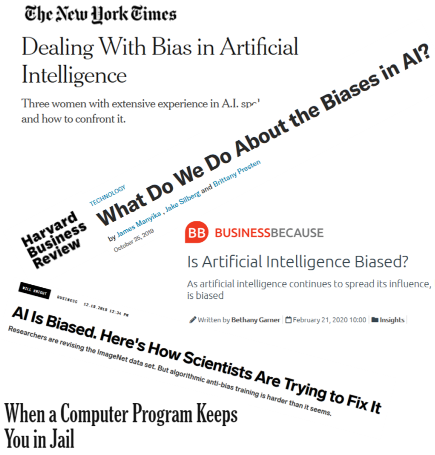
Axel Parmentier[1], Thibaut Vidal[2,3]

[1] Cermics, École des Ponts Paristech
[2] CIRRELT & SCALE-AI Chair in Data-Driven Supply Chains, MAGI, Polytechnique Montreal, Canada
[3] Department of Computer Science, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil

Image Credit: Hitchhikers Guide to the Galaxy

Sensitive applications of ML require Transparency and Explainability

- **Machine learning** applied to high stakes decisions:
  - Recurrence predictions in medicine
  - Credit default risk evaluations
  - Even in contexts where it should not be applied in current form (e.g., bail decisions in criminal justice...)
- **Critical decisions** $\Rightarrow$ Right to have explanations and recourse, i.e., "what can I do to change the outcome".
- **Counterfactual explanations:** contrastive arguments of the type: "To obtain this loan, you need \$40,000 of annual revenue instead of the current \$30,000".
  - Ideally, good counterfactual explanations provide the "smallest" set of changes of the features (or actions) needed to achieve the desired class,
  - Bound by additional constraints imposing plausibility and actionability.



The New York Times

Dealing With Bias in Artificial Intelligence

Three women with extensive experience in A.I. spe
and how to confront it.

Harvard Business Review

TECHNOLOGY

What Do We Do About the Biases in AI?

by James Manyika, Jake Silberg and Brittany Presten

October 25, 2019

BB BUSINESSBECAUSE

Is Artificial Intelligence Biased?

As artificial intelligence continues to spread its influence, is biased

Written by Bethany Garner | February 21, 2020 10:00 | Insights

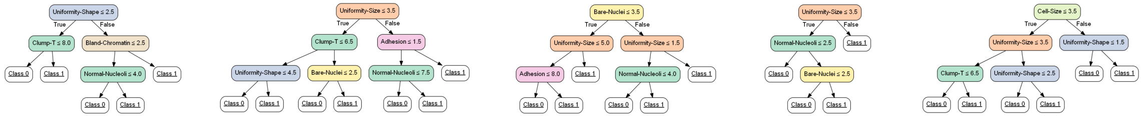
WILL KNIGHT BUSINESS 12.19.2019 12:34 PM

AI Is Biased. Here's How Scientists Are Trying to Fix It

Researchers are revising the ImageNet data set. But algorithmic anti-bias training is harder than it seems.

When a Computer Program Keeps You in Jail

Explanations in Tree Ensembles



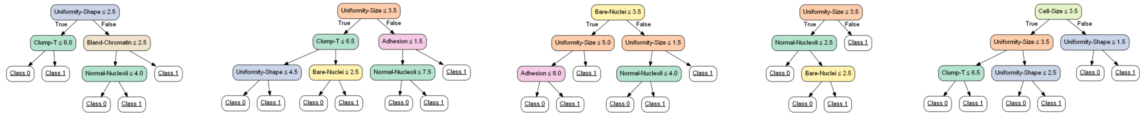
Counterfactual Search

Given an origin point $\hat{\mathbf{x}}$ and a desired prediction class c^*, searching for a plausible and actionable counterfactual explanation consists in locating a new data point $\mathbf{x} \in \mathcal{X}$ that solves the following problem:

$$\begin{aligned} \min \quad & f_{\hat{\mathbf{x}}}(\mathbf{x}) \\ \text{s.t.} \quad & F_{\mathcal{T}}(\mathbf{x}) = c^* \\ & \mathbf{x} \in X^{\mathrm{P}} \cap X^{\mathrm{A}} \end{aligned}$$

- Finding counterfactual explanations in **tree ensembles** is notably difficult:
  - Function $F_{\mathcal{T}}(\mathbf{x})$ has a number of pieces that grows as the *product* of the number of leaves of the trees [7].
  - Changing any feature impacts the trajectory in all trees $\Rightarrow$ searching for the right combination of leaves
  - Non-convex, non differentiable decision function, NP-hard optimization problem.

HEURISTIC vs OPTIMAL Explanations



- Current **HEURISTIC** explanation algorithms, e.g., Feature Tweaking (FT – [5]) regularly produce suboptimal solutions
  - Largely overshooting the actions (up to 31.7× in our experiments) needed to achieve the desired outcome.
  - Unstable solution quality, widely varying between different subjects and subject groups...
  - ***Is this transparent and fair?***

- **OPTIMAL** counterfactual search through mixed-integer linear programming (MILP) provides explanations grounded on a mathematical definition, independently of the search algorithm
  - The flexibility of the modeling framework permit to seamlessly include a wide diversity of metrics, objectives and constraints
  - As seen in this study, is possible to achieve optimal results within seconds

OCEAN – Optimal Counterfactual Explanations

- MILP = solution of a problem represented as a set of linear equations, in which some variables are restricted to the integer domain, under an objective measuring how difficult it is to act on the different features.
- Solved to optimality with a branch-and-cut solver (Gurobi)
- Our model has several desirable characteristics that permit an efficient solution:
  - **logarithmic number of integer variables** (not linear as in [1, 2]);
  - **tighter linear relaxation** than previous models $\Rightarrow$ improves branch-and-cut performance
- The approach is very flexible:
  - applicable to **heterogeneous data** with numerical, ordinal, categorical and binary features;
  - large **variety of objectives**: l_0, l_1 and l_2 norm and extensions thereof;
  - additional **actionability** and **plausibility** constraints.

Sample Flows

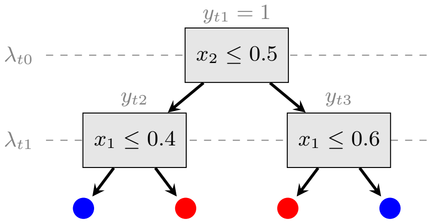
The $\boldsymbol{\lambda}$ variables represent the branch decisions for each tree t at each layer d. The $\mathbf{y}$ variables represent the flows of the counterfactual example through each tree.

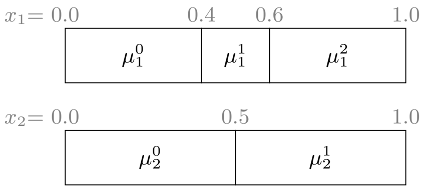
$$\begin{aligned}
& y_{t1} = 1 && t \in \mathcal{T} \\
& y_{tv} = y_{tl(v)} + y_{tr(v)} && t \in \mathcal{T}, v \in \mathcal{V}_t^I \\
& \sum_{v \in \mathcal{V}_{td}^I} y_{tl(v)} \leq \lambda_{td} && t \in \mathcal{T}, d \in \mathcal{D}_t \\
& y_{tv} \in [0,1] && t \in \mathcal{T}, v \in \mathcal{V}_t^I \cup \mathcal{V}_t^I \\
& \lambda_{td} \in \{0,1\} && t \in \mathcal{T}, d \in \mathcal{D}_t .
\end{aligned}$$

Numerical Features

The $\boldsymbol{\mu}$ variables represent the levels of numerical features as ordered simplices, also connecting them with the variables representing the branch choices.

$$\begin{aligned}
& \mu_i^{j-1} \geq \mu_i^j && j \in \{1, \dots, k_i\} \\
& \mu_i^j \leq 1 - y_{tl(v)} && j \in \{1, \dots, k_i\}, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
& \mu_i^{j-1} \geq y_{tr(v)} && j \in \{1, \dots, k_i\}, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
& \mu_i^j \geq \epsilon y_{tr(v)} && j \in \{1, \dots, k_i\}, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
& \mu_i^j \in [0,1] && j \in \{0, \dots, k_i\}
\end{aligned}$$





OCEAN – Optimal Counterfactual Explanations

Categorical Features

The $\boldsymbol{\nu}$ variables represent the possible categories:

$$\begin{aligned}
&\nu_i^j \leq 1 - y_{tl(v)} && j \in C_i, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
&\nu_i^j \geq y_{tr(v)} && j \in C_i, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
&\nu_i^j \in \{0,1\} && j \in C_i \\
&\sum_{j \in C_i} \nu_i^j = 1
\end{aligned}$$

Ordinal Features

The $\boldsymbol{\omega}$ variables represent the relevant levels for the ordinal features (only those appearing in some splitting hyperplanes):

$$\begin{aligned}
&\omega_i^{j-1} \geq \omega_i^j && j \in \{2, \dots, k_i - 1\} \\
&\omega_i^j \leq 1 - y_{tl(v)} && j \in \{1, \dots, k_i - 1\}, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
&\omega_i^j \geq y_{tr(v)} && j \in \{1, \dots, k_i - 1\}, t \in \mathcal{T}, v \in \mathcal{V}_{tij}^I \\
&\omega_i^j \in \{0,1\} && j \in \{1, \dots, k_i - 1\}
\end{aligned}$$

Binary Features

Simply done through binary $\mathbf{x}$ variables:

$$\begin{aligned}
&x_i \leq 1 - y_{tl(v)} && t \in \mathcal{T}, v \in \mathcal{V}_{ti}^I \\
&x_i \geq y_{tr(v)} && t \in \mathcal{T}, v \in \mathcal{V}_{ti}^I \\
&x_i \in \{0,1\}
\end{aligned}$$

The domain of the variables representing the features can be relaxed to the continuous interval $[0,1]$ while retaining integrality of the linear-relaxation solutions (with the simplex algorithm).

OCEAN – Optimal Counterfactual Explanations

For **binary**, **categorical**, or **ordinal** features, we can freely set a weight for each discrete choice in the objective. For **continuous** numerical features, we can proceed as follows:

Objective for numerical features

$$l_0 : \begin{cases} f_0^{\mathrm{N}}(\boldsymbol{\mu}) = \sum_{i \in I_{\mathrm{N}}} (c_i^- z_i^- + c_i^+ z_i^+) \\ z_i^- \geq 1 - \mu_i^{j-1}, z_i^+ \geq \mu_i^j & i \in I_{\mathrm{N}}, j = \hat{j}_i \\ z_i^- \in \{0,1\}, z_i^+ \in \{0,1\} & i \in I_{\mathrm{N}} \end{cases}$$

$$l_1 : \begin{cases} f_1^{\mathrm{N}}(\boldsymbol{\mu}) = \sum_{j=0}^{k_i} (\phi_i^{j+1} - \phi_i^j)\mu_i^j \\ \text{with parameter } \phi_i^j = c_i^- \max(\hat{x}_i - x_i^j, 0) \\ \qquad\qquad + c_i^+ \max(x_i^j - \hat{x}_i, 0) \end{cases}$$

Achieving the desired counterfactual class c^* though majority vote can be expressed as:

$$z_c = \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}_t^L} w_t p_{tvc} y_{tv} \qquad c \in \mathcal{C}$$

$$z_{c^*} > z_c \qquad c \in \mathcal{C}, c \neq c^*$$

OCEAN – Optimal Counterfactual Explanations

Many additional constraints related to actionability and plausibility can be seamlessly integrated into the model:

Domain Knowledge	**Constraints**
Fixed features	$x_i = \hat{x}_i,\ \mu_i = \hat{\mu}_i,\ \nu_i = \hat{\nu}_i$
Monotonic features	$x_i \geq \hat{x}_i,\ \mu_i \geq \hat{\mu}_i,\ \nu_i \geq \hat{\nu}_i$
Known **linear relations** between features (i.e., joint actionability – Venkatasubramanian and Alfano 6)	$A(x_i - \hat{x}_i) \leq \mathbf{b}$
Known **logical implications** between features,	
Example for binary features $(x_1 = \text{True}) \Rightarrow (x_2 = \text{True})$	$x_2 \geq x_1$
Example for categorical features $x_1 \in \{\text{Cat1}, \text{Cat2}\} \Rightarrow x_2 \in \{\text{Cat3}, \text{Cat4}\}$	$\nu_2^3 + \nu_2^4 \geq \nu_1^1 + \nu_1^2$
Resource constraints (e.g., time) as modeled by additional functions $g_i(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\mu})$	$g_i(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\mu}) \leq b_i$

Experimental Setup

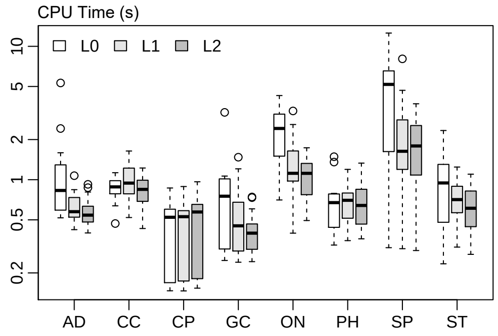
- We used heterogeneous datasets coming from a wide range of applications, with up to 45222 samples and 57 features.
  - Divided each data set into 80% training and 20% test
  - For each data set, trained a random forest (RF) with 100 trees and maximum depth of 5, and selected 20 origin samples with negative outcome for the counterfactual explanations
  - Saved/Serialized all the RF and samples for a fair comparison between different counterfactual explanation methods

Data set	n	p	p_{N}	p_{B}	p_{C}	Src.
AD: Adult	45222	11	5	2	4	UCI
CC: Credit Card Default	29623	14	11	3	0	UCI
CP: COMPAS	5278	5	2	3	0	ProPublica
GC: German Credit	1000	9	5	1	3	UCI
ON: Online News	39644	47	43	2	2	UCI
PH: Data Phishing	11055	30	8	22	0	UCI
SP: Spambase	4601	57	57	0	0	UCI
ST: Students Performance	395	30	13	13	4	UCI

- Data, code and scripts available at `https://github.com/vidalt/OCEAN`.

Computational Experiments – Performance and Optimality

Measuring the time needed to find optimal counterfactual explanations with OCEAN for different objectives and data sets



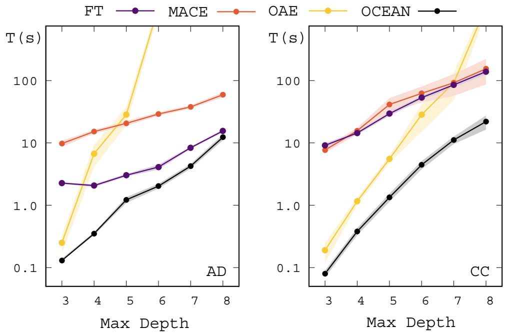
Comparing CPU time and solution quality with other approaches for RF explanations: FT [5], MACE [3], OAE [1, 2]. We use the l_1 objective (which is common to all methods) and the same serialized RFs.

Data	**FT**		**MACE**		**OAE**		**OCEAN**	
	T(s)	R	T(s)	R	T(s)	R	T(s)	R
AD	3.03	15.9	20.60	1.1	28.37	1.0	1.22	1.0
CC	29.44	10.2	41.25	1.2	5.52	1.0	1.34	1.0
CP	22.68	4.5	15.82	1.0	0.38	1.0	0.52	1.0
GC	16.26	4.8	19.03	1.0	5.08	1.0	1.16	1.0
ON	10.05	31.7	>900	—	>900	—	2.97	1.0
PH	10.95	1.4	>900	—	0.94	1.0	0.52	1.0
SP	NA	—	>900	—	>900	—	2.73	1.0
ST	NA	—	>900	—	69.64	1.0	1.10	1.0

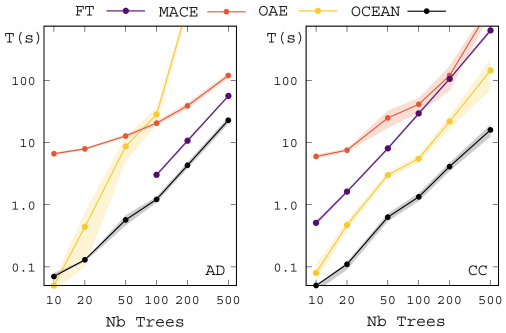
R = Ratio between l_1 distance found & optimum
NA = No counterfactual explanation found
>900 = Time limit exceeded

Computational Experiments – Scalability

Comparative analysis of CPU time as a function of the maximum depth of the trees. Number of trees fixed to 100:

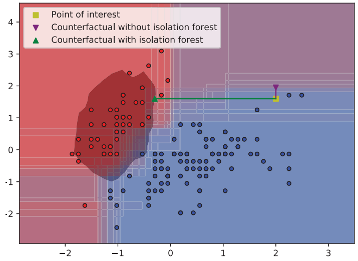


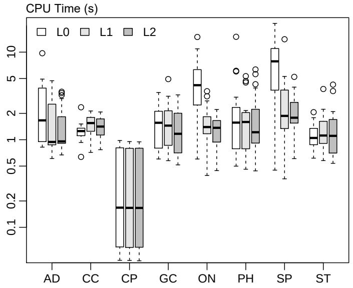
Comparative analysis of CPU time as a function of the number of trees in the ensemble. Maximum depth fixed to 5:



Computational Experiments – Isolation Forests for Plausibility

- Isolation forests [4] are trained to return an outlier score for any sample, inversely proportional to its average path depth within a set of randomized trees.
  - Constraining this average depth to be greater than a threshold δ controls the plausibility of the counterfactual explanation.
  - This is done within the same MILP formulation, with an additional set of constraints representing the IF. We select δ to capture 10% of the training data as an outlier $\Rightarrow$ counterfactual explanation typical of the 90% most common samples for the target class.
  - Computational time remains tractable even with the addition of the IF





Conclusions & Perspectives

- Optimal counterfactual explanations are achievable for most tabular datasets of practical interest
- The flexibility of an appaoch based on MILP gives much-needed flexibility to integrate additional objectives, penalty terms, and constraints related to actionability and plausibility
- Models are still evolving, and likely to need customization for each application at hand
- Further developments could focus on improving performance, but without losing sight of formulation ease and extendability

References

References

[1] Cui, Z., W. Chen, W. He, Y. Chen. 2015. Optimal action extraction for random forests and boosted trees. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 179–188.

[2] Kanamori, K., T. Takagi, K. Kobayashi, H. Arimura. 2020. DACE: Distribution-aware counterfactual explanation by mixed-integer linear optimization. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20* 2855–2862.

[3] Karimi, A.-H., G. Barthe, B. Balle, I. Valera. 2020. Model-agnostic counterfactual explanations for consequential decisions. Silvia Chiappa, Roberto Calandra, eds., *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research*, vol. 108. PMLR, 895–905.

[4] Liu, F.T., K.M. Ting, Z.-H. Zhou. 2008. Isolation forest. *2008 Eighth IEEE International Conference on Data Mining*. 413–422.

[5] Tolomei, Gabriele, Fabrizio Silvestri, Andrew Haines, Mounia Lalmas. 2017. Interpretable predictions of tree-based ensembles via actionable feature tweaking. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, 465–474.

[6] Venkatasubramanian, Suresh, Mark Alfano. 2020. The philosophical basis of algorithmic recourse. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* 284–293.

[7] Vidal, T., M. Schiffer. 2020. Born-again tree ensembles. Hal Daumé III, Aarti Singh, eds., *Proceedings of the 37th International Conference on Machine Learning*, vol. 119. PMLR, Virtual, 9743–9753.

3) Optimal Decision Diagrams for Classification

Pedro Martins[1], Maximilian Schiffer[2], Thiago Serra[3], Thibaut Vidal[4,1]

[1] Department of Computer Science, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil
[2] TUM School of Management, Technical University of Munich, Germany
[3] Freeman College of Management, Bucknell University, USA
[4] CIRRELT & SCALE-AI Chair in Data-Driven Supply Chains, MAGI, Polytechnique Montreal, Canada

Image Credit: Hitchhikers Guide to the Galaxy

Decision Diagram for Classification – An Underexploited Opportunity?

- Decision diagrams (DD – also called decision graphs or decision streams) have a long history
  - in logic synthesis and formal circuit verification [5, 6]
  - in the optimization and artificial intelligence domains [1, 13].
  - In machine learning, regularly re-emerging as a possible classification model [10, 12, 13, 14] or a by-product of model compression algorithms [4, 7, 9].
  - A DD is represented as a rooted directed acyclic graph in which each internal node represents a splitting hyperplane, and each terminal node is uniquely associated to a class.

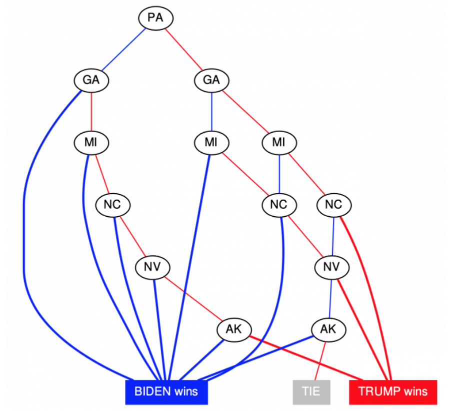


Figure: Credit to Stefan Szeider, TU Wien

- The topology of the graph is free (i.e., internal edge connections)
  $\Rightarrow$ **DD learning requires *jointly* determining the splitting hyperplanes and the internal connections.**

Decision Diagram for Classification – An Underexploited Opportunity?

- Decision diagrams have notable advantages over decision trees.
    - Their width is not bound to grow exponentially with their depth, allowing training deep but narrow decision diagrams without facing issues of data fragmentation [10, 14].
    - Additional degrees of freedom in their topology design permit to express a richer set of concepts and to achieve better model compression [4, 11].
- Despite this, decision diagrams have been more rarely used than decision trees
    - Learning them is inherently complex: a decision diagram topology cannot easily be optimized by construction and local optimization
- **Our goal: finally proposing efficient training algorithms based on MILP.**

Optimal Decision Diagram Training as a MILP

Flow variables:

- For each sample i and internal node $u \in \mathcal{V}^{\mathrm{C}}$, define a pair of flow variables $w^-_{iu} \in [0,1]$ and $w^+_{iu} \in [0,1]$.
  - A non-zero value in w^-_{iu} (respectively w^+_{iu}) means that sample i passes through node u on the negative side of the separating hyperplane (on the positive side, respectively).
- Define variables $z^-_{iuv} \in [0,1]$ (respectively $z^+_{iuv} \in [0,1]$) to characterize the flow going from the positive and negative sides of u to other nodes v.

$$w^+_{iv} + w^-_{iv} = \begin{cases} 1 & \text{if } v = 0 \\ \sum_{u \in \delta^-(v)} (z^+_{iuv} + z^-_{iuv}) & \text{otherwise} \end{cases} \qquad v \in \mathcal{V}^{\mathrm{I}}, i \in \{1,\dots,n\} \tag{1}$$

$$w^-_{iu} = \sum_{v \in \delta^+(u)} z^-_{iuv} \qquad u \in \mathcal{V}^{\mathrm{I}}, i \in \{1,\dots,n\} \tag{2}$$

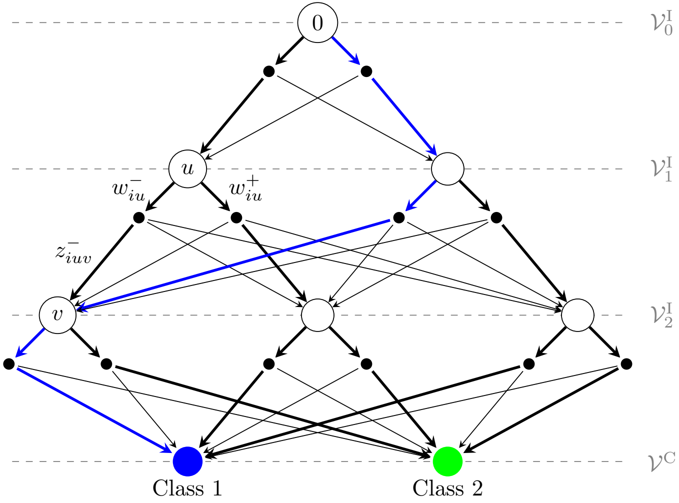
$$w^+_{iu} = \sum_{v \in \delta^+(u)} z^+_{iuv} \qquad u \in \mathcal{V}^{\mathrm{I}}, i \in \{1,\dots,n\} \tag{3}$$

Optimal Decision Diagram Training as a MILP

Thick edges represent a possible decision-graph topology (selected by the training algorithm)

Flow variables w_{iu}^-, w_{iu}^+ and z_{iuv}^- indicate the trajectory of sample i. The following conditions always hold: $(w_{iu}^- = 1) \Rightarrow (\mathbf{a}_u^\mathsf{T} x_i < b_u)$ $(w_{iu}^+ = 1) \Rightarrow (\mathbf{a}_u^\mathsf{T} x_i \geq b_u)$

The blue path corresponds to the possible trajectory of a sample classified as Class 1



0
$\mathcal{V}_0^\mathrm{I}$
u
w_{iu}^-
w_{iu}^+
$\mathcal{V}_1^\mathrm{I}$
z_{iuv}^-
v
$\mathcal{V}_2^\mathrm{I}$
$\mathcal{V}^\mathrm{C}$
Class 1
Class 2

Also additional "long arcs" connecting the internal nodes to the leaves.

Optimal Decision Diagram Training as a MILP

- Integrality of the flow variables is not guaranteed, due to the constraints coming from the hyperplanes.
- To obtain integer sample flows, using an additional binary variable $\lambda_{il} \in \{0, 1\}$ for each sample $i \in \{1, \dots, n\}$ and level $l \in \{0, \dots, D-1\}$:

$$\sum_{u \in \mathcal{V}_l^{\mathrm{I}}} w_{iu}^{-} \leq 1 - \lambda_{il} \qquad l \in \{0, \dots, D-1\},\ i \in \{1, \dots, n\} \tag{4}$$

$$\sum_{u \in \mathcal{V}_l^{\mathrm{I}}} w_{iu}^{+} \leq \lambda_{il} \qquad l \in \{0, \dots, D-1\},\ i \in \{1, \dots, n\} \tag{5}$$

- Sample i can only go to the negative (respectively positive) side of any node u of level $\mathcal{V}_l^{\mathrm{I}}$ if $\lambda_{il} = 0$ (respectively $\lambda_{il} = 1$).

Optimal Decision Diagram Training as a MILP

Decision diagram topology:

- Connect the flow variables to the binary design variables that characterize the topology of the diagram.
- Define binary variable $d_u \in \{0,1\}$ for each $u \in \mathcal{V}$ that takes value 1 if this node is used in the classification.
- For the negative and positive sides of each node $u \in \mathcal{V}^{\mathrm{I}}$, we create binary design variables $y^-_{uv} \in \{0,1\}$ and $y^+_{uv} \in \{0,1\}$ taking value 1 if and only if u links towards v on the negative and positive sides, respectively.

$$d_u = \sum_{v \in \delta^+(u)} y^+_{uv} = \sum_{v \in \delta^+(u)} y^-_{uv} \qquad u \in \mathcal{V}^{\mathrm{I}} \tag{6}$$

$$y^+_{uv} + y^-_{uv} \leq d_v \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(u) \tag{7}$$

$$z^+_{iuv} \leq y^+_{uv}, \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(u), i \in \{1,\dots,n\} \tag{8}$$

$$z^-_{iuv} \leq y^-_{uv} \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(u), i \in \{1,\dots,n\} \tag{9}$$

Optimal Decision Diagram Training as a MILP

Reducing symmetry.

- We impose that the arcs (u, v) and (u, w) such that $y^-_{uv} = 1$ and $y^+_{uw} = 1$ satisfy $v < w$ for each internal node $u \in \mathcal{V}^{\mathrm{I}}$. This corresponds to the logical constraint $(y^-_{uv} = 1) \Rightarrow (y^+_{uw} = 0\ \forall w \leq v)$, formulated as

$$y^-_{uv} + \sum_{w \in \delta^+(u), w \leq v} y^+_{uw} \leq 1 \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(u). \tag{10}$$

- Let $p(u) \in \{1, \ldots, |\mathcal{V}^{\mathrm{I}}_l|\}$ be the position of each internal node u at depth l within its layer. We also impose:

$$y^-_{uv} = 0 \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(v) \cap \mathcal{V}^{\mathrm{I}}, p(v) \geq 2p(u) \tag{11}$$

$$y^+_{uv} = 0 \qquad u \in \mathcal{V}^{\mathrm{I}}, v \in \delta^+(v) \cap \mathcal{V}^{\mathrm{I}}, p(v) > 2p(u) \tag{12}$$

$$d_u \geq d_v \qquad p(v) = p(u) + 1, u \in \mathcal{V}^{\mathrm{I}}_l, v \in \mathcal{V}^{\mathrm{I}}_l, l \in \{1, \ldots, D-1\} \tag{13}$$

Optimal Decision Diagram Training as a MILP

Linear separator variables and consistency with the sample flows:

- Associate to each internal node $v \in \mathcal{V}^{\mathrm{I}}$ a vector of variables $\mathbf{a}_v \in [-1,1]^d$ and a variable $b_v \in [-1,1]$ to characterize the splitting hyperplane.
- Add *indicator constraints* to connect the path of the samples with their relative positions from the hyperplane:

$$(w^-_{iv} = 1) \Rightarrow (\mathbf{a}_v^\mathsf{T} \mathbf{x}^i + \varepsilon \leq b_v) \qquad i \in \{1,\ldots,n\}, v \in \mathcal{V} \tag{14}$$

$$(w^+_{iv} = 1) \Rightarrow (\mathbf{a}_v^\mathsf{T} \mathbf{x}^i \geq b_v) \qquad i \in \{1,\ldots,n\}, v \in \mathcal{V} \tag{15}$$

Optimal Decision Diagram Training as a MILP

Objective function:

- In a similar fashion as in [2], optimizing accuracy with an additional regularization term that favors simple decision diagrams with few internal nodes.
- Using variables $w_{iv} \in [0,1]$ for each sample $i \in \{1,\dots,n\}$ and leaf $v \in \mathcal{V}^{\mathrm{C}}$ expressing the amount of flow of i reaching terminal node v with class c_v. These variables must satisfy the following constraints.

$$w_{iv} = \sum_{u \in \delta^-(v)} (z^+_{iuv} + z^-_{iuv}) \qquad v \in \mathcal{V}^{\mathrm{C}}, i \in \{1,\dots,n\} \tag{16}$$

- Objective can be stated as

$$\min \frac{1}{n} \sum_{i=1}^{n} \sum_{v \in \mathcal{V}^{\mathrm{C}}} \phi_{iv} w_{iv} + \frac{\alpha}{|\mathcal{V}^{\mathrm{I}}| - 1} \sum_{v \in \mathcal{V}^{\mathrm{I}} - \{0\}} d_v, \tag{17}$$

where ϕ_{iv} represents the mismatch penalty when assigning sample i to terminal node v (typically defined as 0 if $c^i = c_v$ and 1 otherwise), and α is a regularization parameter.

General Solution Approach

Step 1: Initial construction and improvement:

- Top-down construction approach which shares some common traits with CART [3].
  - For each layer $l \in \{0, \dots, D-2\}$ and each internal node $u \in \mathcal{V}_l^{\mathrm{I}}$, select the univariate split that maximizes the information gain.
  - To determine the internal connections, use a greedy merging policy: as long as the number of sample flows is greater than w_{l+1}, **join** the pair of flows that least decreases the information gain.
- To obtain a better initial diagram, repeat the construction process $N_{\mathrm{IT}} = 5$ times and consider only a random subset of 60% of the features during each split selection.
- Further improve each resulting decision diagram by re-optimizing each split from top to bottom without changing the internal topology.

General Solution Approach

Step 2: Iterative refinement. Improve the decision diagram by considering each internal node $u \in \mathcal{V}^{\mathrm{I}}$ from top to bottom, fixing in the MILP all design variables $\mathbf{a}_u$, b_u, d_u, y^-_{uv}, y^+_{uv}, to their current value except for node u, as well as all sample flows that do not pass through node u. Solve each restricted problem with Gurobi using a time limit of 20 seconds.

Step 3: Solution of the MILP. Finally, apply Gurobi on the complete MILP using the solution found in the previous step as a warm start. Time limit of $T_{\text{MAX}} = 300$ seconds for this phase.

Experimental Setup

- Same selection of 54 data sets as in Bertsimas and Dunn [2].
- All these data sets are publicly available from the UCI machine learning repository [8].
- Reflect a wide range of practical classification applications and contain between 47 to 6435 data points with 2 to 484 features.
    - Each data set is divided each data set into 50% training, 25% validation and 25% test
    - To increase statistical significance, experiments are repeated 10 times with different random seeds for each data set

Hyperparameter Calibration and Performance

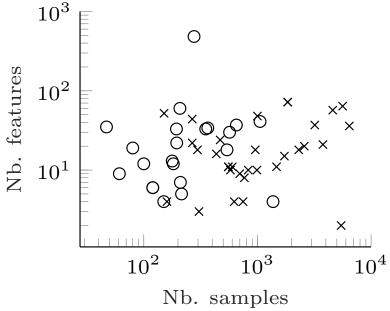
Hyper-parameters tuning using the validation set.

- Consider different values of the regularization parameter $\alpha \in \{0.0, 0.1, 0.25, 0.5, 1.0\}$
- Consider four different decision diagram skeletons: 1–2–4–8, 1–2–4–4–4, 1–2–3–3–3–3, and 1–2–2–2–2–2–2–2, hereby referred to as Skeletons I to IV, respectively.
- Train each configuration: measuring
  - (i) computational effort,
  - (ii) ability to achieve optimal training,
  - (iii) classification performance on the validation set and best configuration of hyper-parameters.

Hyperparameter Calibration and Performance

Computational performance. Table shows, for each skeleton and α combination, the number of runs (out of 54 data sets $\times$ 10 seeds = 540 runs) for which a global optimum was found. Figure further locates data sets that can be solved to optimality in relation to their characteristics.

	α					
Skeleton	0	0.1	0.25	0.5	1	Total
I	282	263	266	270	270	1351
II	305	278	279	281	278	1421
III	294	271	276	268	269	1378
IV	298	298	298	293	299	1486
Total	1179	1110	1119	1112	1116	5636



Regularization Sensitivity

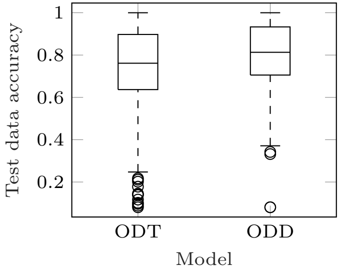
Also analyzing how often each combination of a value for the sparsity parameter α and skeleton yields the best accuracy on the validation set.

	α					
Skeleton	0	0.1	0.25	0.5	1	Total
I	41	26	26	33	41	167
II	48	23	21	26	32	150
III	61	15	27	23	33	159
IV	11	17	7	15	14	64
Total	161	81	81	97	120	540

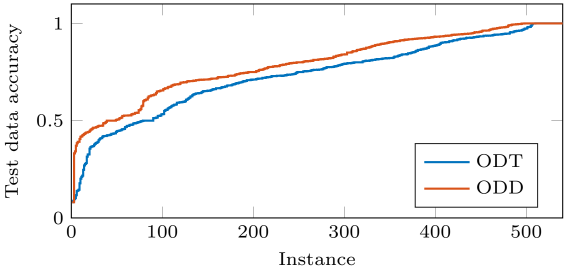
Performance Analysis

- Finally, comparing the performance of the ODDs with those of optimal decision trees (ODTs).
  - Using the best skeleton from the hyper-parameter calibration phase for each data set.
  - ODTs obtained by fixing the decision variables representing the internal topology of the graph. We apply the same hyperparameter calibration process to calibrate their α parameters.

Distribution of the classification accuracy on the test data over all data sets and runs for the respective best ODT and ODD models:

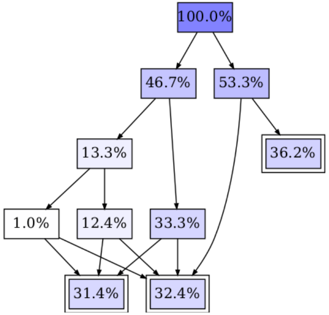


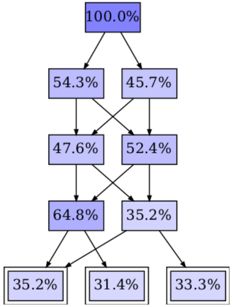
Accuracy of all 54×10 data sets and runs for ODD and ODT, sorted in ascending order:



Performance Analysis

- Performance gain of ODD over ODT comes from additional freedom in topology, permitting to further exploit all decision nodes. The following example illustrates the optimal topologies found on the "seeds" data set, as well as the data fragmentation within the classifiers.





ODDs – Conclusions & Perspectives

- Studied ODDs training from a combinatorial optimization viewpoint.
- First MILP for this task running within short computational time
  - Methodology seems to allow optimal training for half of the data sets considered.
  - Degrees of freedom of the model permit to find good topologies for each data set and avoids data fragmentation $\Rightarrow$ leads to better accuracy and generalization capability on the test set.
- Flexibility of MILP permits to extend the model towards a variety of important side requirements, such as fairness, parsimony and stability.

General Conclusions

- Many research opportunities at the intersection between combinatorial optimization and explainable/interpretable ML.
- Optimal training has received acclaim as a critical asset for maximizing accuracy or performance (mainly due to the risk of over-fitting). **However,** it has a role to play given the growing emphasis towards sparse and interpretable models (compactness needs optimization).
- Some tasks are more critical than training: explanations, guarantees, error diagnosis.
    - Those are domains where search methods grounded on combinatorial optimization can excel.
- Finally, some families of models are more prone for applications of combinatorial optimization (e.g., random forests vs deep neural networks).
    - Evolution of algorithms for these tasks may lead us to reconsider the current dominance of DNNs in a wide variety of domains...

Thanks !

THANK YOU FOR YOUR ATTENTION !

Contact me (also job announcements at the M.Sc., Ph.D. and postdoc levels in relation to the SCALE-AI chair):
`thibaut.vidal@polymtl.ca`

Articles, slides and data sets:
`http://w1.cirrelt.ca/~vidalt/`

Source codes:
`https://github.com/vidalthi/`

Regular updates and announcements:
`https://twitter.com/vidalthi`

Bibliography I

[1] Bergman, D., A.A. Cire, W.-J. Van Hoeve, J. Hooker. 2016. *Decision diagrams for optimization*. Springer International Publishing.

[2] Bertsimas, Dimitris, Jack Dunn. 2017. Optimal classification trees. *Machine Learning* **106**(7) 1039–1082.

[3] Breiman, L., J.H. Friedman, R.A. Olshen, C.J. Stone. 1984. *Classification and regression trees*.

[4] Breslow, L.A., D.W. Aha. 1997. Simplifying decision trees: A survey. *Knowledge Engineering Review* **12**(1) 1–40.

[5] Bryant, R.E. 1986. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers* **C-35**(8) 677–691.

[6] Bryant, R.E. 1992. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys* **24**(3) 293–318.

[7] Choudhary, T., V. Mishra, A. Goswami, J. Sarangapani. 2020. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review* 1–43.

[8] Dua, D., C. Graff. 2017. UCI machine learning repository.

[9] Gossen, F., B. Steffen. 2019. Large random forests: Optimisation for rapid evaluation. *arXiv preprint arXiv:1912.10934* .

Bibliography II

[10] Ignatov, D., A. Ignatov. 2018. Decision stream: Cultivating deep decision trees. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI* 905–912.

[11] Kumar, A., S. Goyal, M. Varma. 2017. Resource-efficient machine learning in 2 KB RAM for the Internet of Things. *34th International Conference on Machine Learning, ICML 2017* **4** 3062–3071.

[12] Oliveira, A.L., A. Sangiovanni-Vincentelli. 1996. Using the minimum description length principle to infer reduced ordered decision graphs. *Machine Learning* **25**(1) 23–50.

[13] Oliver, J. 1993. Decision graphs – An extension of decision trees. *Proceedings of the 4th international workshop on artificial intelligence and statistics (AISTATS)*. 343—350.

[14] Shotton, J., S. Nowozin, T. Sharp, J. Winn, P. Kohli, A. Criminisi. 2013. Decision jungles: Compact and rich models for classification. *Advances in Neural Information Processing Systems* 1–9.