skema BUSINESS SCHOOL | AI R&D CENTER CANADA

Tractability and Optimization of Shap-Score Computation for Explainable AI

Leopoldo Bertossi

France, November 2023

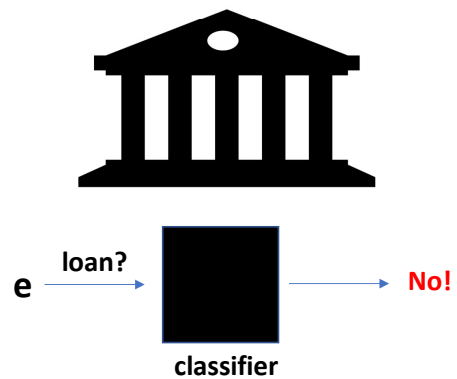
www.scs.carleton.ca/∼bertossi

Explanations in Machine Learning

- Bank client $\mathbf{e} = \langle \text{john}, 18, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle$

  As an entity represented as a record of values for features Name, Age, Activity, Income, ...

- **e** requests a loan from a bank that uses a classifier



- The client asks *Why?*
- What kind of *explanation?*

  How?

  From what?

- Explanations come in different forms
- Some of them are *causal explanations*, some are *explanation scores* a.k.a. *attribution scores*
- They are sometimes related

  E.g. actual causality leads to responsibility scores
- Large part of our recent research is about the use of causality, and score definition and computation

  In data management and machine learning
- Some of them (in data management or ML)
  - Responsibility (in its original and generalized versions)
  - The Causal Effect score
  - The Shapley value (as Shap in ML)

A Score-Based Approach: Responsibility

- Causality has been developed in AI for three decades or so
- In particular: Actual Causality
- Also the quantitative notion of Responsibility: a measure of causal contribution (the *Resp*-score)
- Both based on Counterfactual Interventions
- Hypothetical changes of values in a causal model to detect other changes

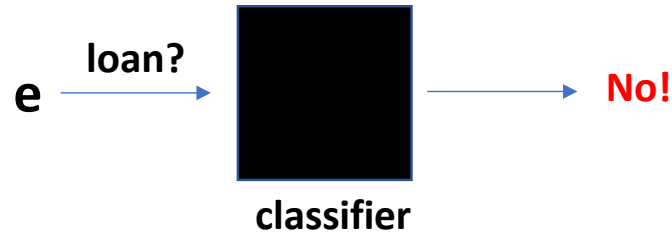
  "What would happen if we change ..."?

  By so doing identify actual causes
- Does the deletion of the DB tuple invalidates the query?
- Does a change of this feature value leads to label "Yes"?

- We have investigated actual causality and responsibility in data management and ML-based classification
- Semantics, computational mechanisms, intrinsic complexity, logic-based specifications, reasoning, etc.
- Assign numbers to, e.g., database tuples or features values to capture their causal, or, more generally, explanatory strength
- They can be applied without knowing "the internals" of a classifier Only input/output relation needed

  It can be a "black box", or treated as such (a complex NN)
- We have experimentally compared responsibility scores with other *local attribution scores*
  - *Shap*
  - *Ad hoc* scores, such as for FICO data on "open-box" model (connected logistic regressions)

- Simplified Case:



$$\mathbf{e} = \langle \text{john}, 18, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle \quad \text{No}$$

- Counterfactual versions:

$$\mathbf{e}' = \langle \text{john}, 25, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle \quad \text{Yes}$$

$$\mathbf{e}'' = \langle \text{john}, 18, \text{plumber}, 80\text{K}, \text{brooklyn}, \ldots \rangle \quad \text{Yes}$$

- For the gist:

1. Value for feature Age is counterfactual cause with explanatory responsibility $\mathit{Resp}(\mathbf{e}, \text{Age}) = 1$
2. Value change Income $:=$ 80K needs an additional, minimum contingent change: $\Gamma = \{\text{Area} := \text{brooklin}\}$

   Income $:=$ 70K is actual cause with $\mathit{Resp}(\mathbf{e}, \text{Income}) = \frac{1}{1+|\Gamma|} = \frac{1}{2}$

The Generalized *Resp* Score

- For binary (two-valued) features the previous "definition" works fine (previous example is non-binary)
- Otherwise, there may be many values for a feature that do not change the label: original value not great explanation

  Similarly for features in a potential contingency set
- Better consider average labels obtained via counterfactual interventions

  Resp, our extended version of responsibility, will be expressed in terms of an expected value[1]

[1] Bertossi, Li, Schleich, Suciu,Vagena; SIGMOD Deem WS'20

- Below, $\mathcal{F}$ is the set of features, the classifier is binary, not necessarily the features

  For $F \in \mathcal{F}$, and entity $\mathbf{e}$, $F(\mathbf{e})$ is value for F in $\mathbf{e}$

  Label $L(\mathbf{e}) = 1$ is the one we want to explain

- Assume $L(\mathbf{e}) = 1$, feature $F^\star$: want $Resp(\mathbf{e}, F^\star)$

  In the example, F* =Salary, F*(e) =70K, and L(e) = 1

- With $F^\star(\mathbf{e})$ fixed, want to define "local" score for fixed contingent assignment $\Gamma := \bar{w}$ $\quad F^\star \notin \Gamma \subseteq \mathcal{F}$

  $\mathbf{e}^{\Gamma,\bar{w}} := \mathbf{e}[\Gamma := \bar{w}]$ (entity obtained changing feature values in $\mathbf{e}$ according to $\Gamma, \bar{w}$)

  Γ ={Location}, and $\bar{\mathbf{w}}$:= $\langle$brooklin$\rangle$, a contingent (new) value for Location

  $\mathbf{e}^{\{\text{Location}\},\langle\text{brooklin}\rangle}$ = **e[Location := brooklin]**

  = $\langle$**john, 25, plumber, 70K, <u>brooklin</u>, 10K, basic**$\rangle$

- Assume $L(\mathbf{e}^{\Gamma,\bar{w}}) = L(\mathbf{e}) = 1$

  Contingent changes alone do not switch label, only after change for $F^\star$

  Assume $\mathbf{L}(\mathbf{e}[\mathbf{Location} := \mathbf{brooklin}]) = \mathbf{L}(\mathbf{e}) = \mathbf{1}$

  Or maybe $\mathbf{L}(\mathbf{e}^{\Gamma',\bar{w}'}) = \mathbf{1}$, with $\Gamma' = \{\mathbf{Activity}, \mathbf{Education}\}$,
  $\bar{w}' = \langle \mathbf{accountant}, \mathbf{medium} \rangle$

- For fixed $\mathbf{e}^{\Gamma,\bar{w}}$, consider entities $\mathbf{e}'$ obtained additionally changing value for $F^\star$ in all possible ways

  (fix values for other features)

  For $\mathbf{e}[\mathbf{Location} := \mathbf{brooklin}]$ fixed, consider:

  $\mathbf{e}'_1 := \mathbf{e}[\mathbf{Location} := \mathbf{brooklin}; \underline{\mathbf{Salary} := \mathbf{60K}}]$
  $= \mathbf{e}^{\mathbf{Location},\langle \mathbf{brooklin} \rangle}[\mathbf{Salary} := \mathbf{60K}]$), maybe with $\mathbf{L}(\mathbf{e}'_1) = \mathbf{1}$

  Or $\mathbf{e}'_2 := \mathbf{e}[\mathbf{Location} := \mathbf{brooklin}; \underline{\mathbf{Salary} := \mathbf{80}}]$, maybe with $\mathbf{L}(\mathbf{e}'_2) = \mathbf{0}$

- Fixed contingency $(\Gamma, \bar{w})$ on $\mathbf{e}$ as above, define its *local* responsibility score

  Difference between original label and the expected label over all possible $\mathbf{e}'$

$$Resp(\mathbf{e}, F^\star, \underline{\Gamma, \bar{w}}) \quad := \quad \frac{L(\mathbf{e}) - \mathbb{E}(\ L(\mathbf{e}') \mid F(\mathbf{e}') = \ F(\mathbf{e}^{\Gamma,\bar{w}}),\ \forall F \in (\mathcal{F} \setminus \{F^\star\})\ )}{1 + |\Gamma|}$$

$$= \quad \frac{1 - \mathbb{E}(\ L(\mathbf{e}^{\Gamma,\bar{w}}[F^\star := v]) \mid v \in Dom(F^\star)\ )}{1 + |\Gamma|} \qquad (*)$$

- Takes into account the size of contingency Γ
- Assumes a probability distribution over entity population (which becomes relevant)
- $F^\star(\mathbf{e})$ is *actual cause* for label 1 if, for some $(\Gamma, \bar{w})$, (*) is positive
- $F^\star(\mathbf{e})$ is a *counterfactual cause* if $\Gamma = \emptyset$ ($\bar{w}$ is empty) and (*) is positive
- Counterfactual causes (as original values in $\mathbf{e}$) may have different causal strengths

  $F_i(\mathbf{e}), F_j(\mathbf{e})$ could be counterfactual causes with different values for (*)

  If changes on the former switch label "fewer times" than for the latter

- Pass from a local score (local for Γ and associated assignment $\bar{w}$)

$$Resp(\mathbf{e}, F^\star, \Gamma, \bar{w}) := \frac{L(\mathbf{e}) - \mathbb{E}(\ L(\mathbf{e}') \mid F(\mathbf{e}') = F(\mathbf{e}^{\Gamma,\bar{w}}),\ \forall F \in (\mathcal{F} \setminus \{F^\star\})\ )}{1+|\Gamma|}$$

To global score, with "best" contingencies $(\Gamma, \bar{w})$

$$Resp(\mathbf{e}, F^\star) := \max_{\Gamma,\bar{w}:\ |\Gamma| \text{ is min. } \&\ (\star) > 0} Resp(\mathbf{e}, F^\star, \Gamma, \bar{w})$$

In particular with Γ of minimum size

- Computation:
  1. First find minimum-size contingency sets Γ's with associated updates $\bar{w}$ with (*) greater that 0
  2. Next, find the maximum value for (*) over those pairs $(\Gamma, \bar{w})$
  3. Starting with $\Gamma = \emptyset$, and iteratively increasing the cardinality of Γ find a $(\Gamma, \bar{w})$
  4. Stop increasing the cardinality, and just check if there is $(\Gamma', \bar{w}')$ with a greater value for (*) and same cardinality

- We are usually interested in feature values with maximum scores

  Associated to minimum (cardinality) contingency sets

- Already with binary domains, *Resp* is intractable[2]

- Can we compute it faster when we have access to the internals?

  This kind of research was done for *Shap* (coming)

[2] Bertossi; TPLP'23

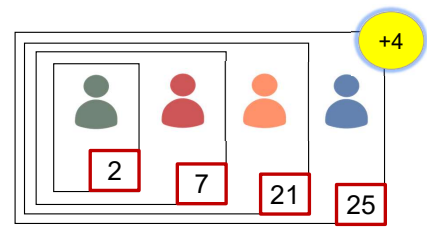
Coalition Games and the Shapley Value

- Usually *several tuples together* violate an IC or produce a query result
- Like players in a *coalition game* contributing, possibly differently, to a shared wealth-distribution function
- Apply standard measures used in game theory: the Shapley value of a player (as a measure of its contribution)
- The Shapley value is a established measure of contribution by players to a wealth function
- It emerges as the only measure enjoying certain properties
- We need a game (function) ...

- Set of players D, and game function $\mathcal{G}: \quad \mathcal{P}(D) \longrightarrow \mathbb{R}$
  ($\mathcal{P}(D)$ the power set of D)
- The Shapley value of player p among a set of players D:

$$Shapley(D, \mathcal{G}, p) := \sum_{S \subseteq D \setminus \{p\}} \frac{|S|!(|D| - |S| - 1)!}{|D|!} (\mathcal{G}(S \cup \{p\}) - \mathcal{G}(S))$$

- $|S|!(|D| - |S| - 1)!$ is number of permutations of D with all players in S coming first, then p, and then all the others
- Expected contribution of player p under all possible additions of p to a partial random sequence of players followed by a random sequence of the rest of the players



- For each application one defines an appropriate game function

- Shapley is difficult to compute

  Naive approach: exponentially many counterfactual combinations

- Actually, Shapley computation is #P-hard in general

- A complexity class of (possibly implicitly) computational counting problems

- Being #P-hard is evidence of difficulty: $\#SAT$ is #P-hard

  Counting satisfying assignments for a propositional formula

  At least as difficult as SAT

Shap **Scores**

- Based on the general Shapley value
- Set of players $\mathcal{F}$ contain features, relative to classified entity $\mathbf{e}$
- We need an appropriate $\mathbf{e}$-dependent game function that maps (sub)sets of players to real numbers
- For $S \subseteq \mathcal{F}$, and $\mathbf{e}_S$ the projection of $\mathbf{e}$ on S:

$$\mathcal{G}_{\mathbf{e}}(S) \;:=\; \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}' \in \mathcal{E} \;\;\&\;\; \mathbf{e}'_S = \mathbf{e}_S)$$

- For a feature $F^\star \in \mathcal{F}$, compute: $\mathit{Shap}(\mathcal{F}, \mathcal{G}_{\mathbf{e}}, F^\star)$

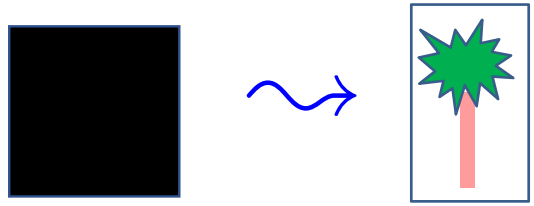
$$\sum_{S \subseteq \mathcal{F} \setminus \{F^\star\}} \frac{|S|!(|\mathcal{F}|-|S|-1)!}{|\mathcal{F}|!} [\underbrace{\mathbb{E}(L(\mathbf{e}')|\mathbf{e}'_{S\cup\{F^\star\}} = \mathbf{e}_{S\cup\{F^\star\}})}_{\mathcal{G}_{\mathbf{e}}(S\cup\{F^\star\})} - \underbrace{\mathbb{E}(L(\mathbf{e}')|\mathbf{e}'_S = \mathbf{e}_S)}_{\mathcal{G}_{\mathbf{e}}(S)}]$$

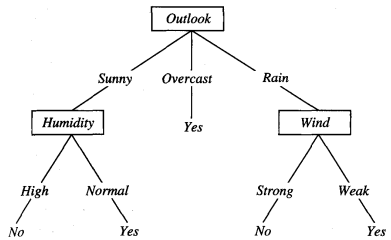
- *Shap* score has become popular (Lee & Lundberg, 2017)
- Assumes a probability distribution on entity population

- *Shap* may end up considering exponentially many combinations

  And multiple passes through the black-box classifier

- Can we do better with an open-box classifier?





  Exploiting its elements and internal structure?

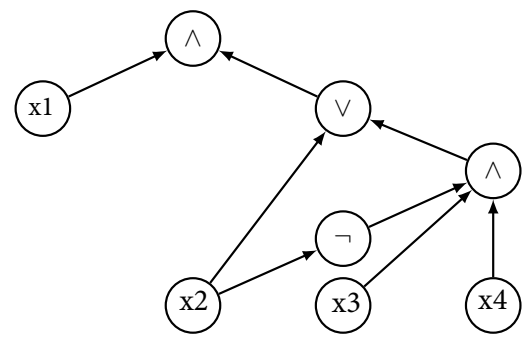
- What if we have a decision tree, or a random forest, or a Boolean circuit?

- Can we compute *Shap* in polynomial time?

Tractability for BC-Classifiers: Big Picture

- We investigated this problem in detail[3]
- Tractable and intractable cases, with algorithms for the former

  Investigated good approximation algorithms
- Choosing the right abstraction (model) is crucial
- We considered Boolean-Circuit Classifiers (BCCs), i.e. propositional formulas with (binary) output gate
- We had shown already that *Shap* is intractable for "Monotone 2CNF" classifiers under the product distribution
  (at most 2 variables per clause, and positive)
- So, it had to be a broad and interesting class of BCs



[3] Arenas, Bertossi, Barcelo, Monet; AAAI'21; JMLR'23

Shap for Boolean-Circuit Classifiers

- Features $F_i \in \mathcal{F},\ i = 1, \ldots, n, \quad Dom(F_i) = \{0, 1\},$
  $\mathbf{e} \in \mathcal{E} := \{0, 1\}^n, \quad L(\mathbf{e}) \in \{0, 1\}$
- There is also a probability distribution P on $\mathcal{E}$
- For BC-classifier L: $\quad Shap(\mathcal{F}, G_{\mathbf{e}}, F^\star) \ =$

  $\sum_{S \subseteq \mathcal{F} \setminus \{F^\star\}} \frac{|S|!(|\mathcal{F}|-|S|-1)!}{|\mathcal{F}|!} [\mathbb{E}(L(\mathbf{e}'|\mathbf{e}'_{S \cup \{F^\star\}} = \mathbf{e}_{S \cup \{F^\star\}}) - \mathbb{E}(L(\mathbf{e}')|\mathbf{e}'_S = \mathbf{e}_S)]$

  Depends on $\mathbf{e}$ and L
- $SAT(L) := \{\mathbf{e}' \mid L(\mathbf{e}') = 1\} \qquad\qquad \#SAT(L) := |SAT(L)|$

  Counting the number of inputs that get label 1
- We established that *Shap* is at least as hard as model counting for the BC:

  Proposition: For the uniform distribution P^u, and $\mathbf{e} \in \mathcal{E}$

  $$\#SAT(L) \ = \ 2^{|\mathcal{F}|} \times (\ L(\mathbf{e}) - \textstyle\sum_{i=1}^{n} Shap(\mathcal{F}, G_{\mathbf{e}}, F_i)\ )$$

- When $\#SAT(L)$ is hard for a Boolean classifier L, *Shap* is also hard
- Corollary: Computing *Shap* is $\#P$-hard for Boolean classifiers defined by Monotone 2DNF or Monotone 2CNF
  (Provan & Ball, 1983)
- Can we do better for other classes of binary classifiers?

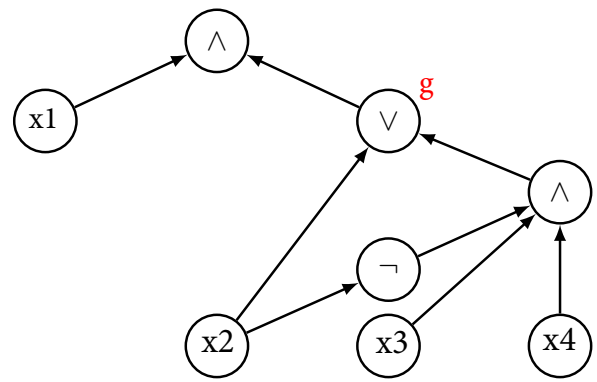
  Other classes of Boolean-circuit classifiers?

Deterministic and Decomposable BCs

- A Boolean circuit over set of variables X is a DAG $\mathcal{C}$ with:
  - Each input (source) node labeled with a variable or a constant in $\{0, 1\}$
  - Other nodes labeled with a gate in $\{\neg, \wedge, \vee\}$
  - Single sink node, O, the output
- For gate g of $\mathcal{C}$, $\mathcal{C}(g)$ is the induced subgraph containing gates on a path in $\mathcal{C}$ to g

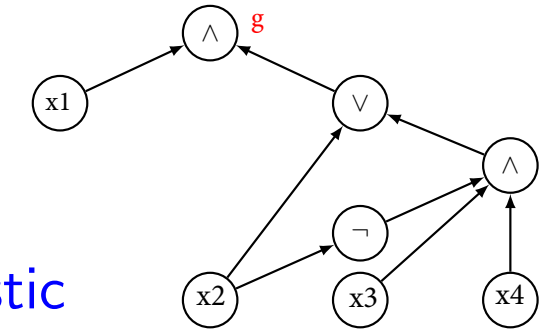
  $Var(g)$ is the set of variables of $\mathcal{C}(g)$

  $Var(g) = \{x2, x3, x4\}$



- $\mathcal{C}$ is deterministic if every $\vee$-gate g with input gates g_1, g_2: $\mathcal{C}(g_1)(\mathbf{e}) \neq \mathcal{C}(g_2)(\mathbf{e})$, for every $\mathbf{e}$

- $\mathcal{C}$ is decomposable if every $\wedge$-gate g with input gates g_1, g_2: $Var(g_1) \cap Var(g_2) = \emptyset$



- We concentrated on the class of deterministic and decomposable Boolean circuits (dDBCs)
- *Shap* computation in polynomial time not initially precluded
- A class of BCCs that includes -via efficient (knowledge) compilation- many interesting ones, syntactic and not ... (more coming)

Shap for dDBCs

- Proposition: For dDBCs $\mathcal{C}$, $\#SAT(\mathcal{C})$ can be computed in polynomial time ($\not\Longleftrightarrow$ the same for *Shap*)

  Idea: Bottom-up procedure that inductively computes $\#SAT(\mathcal{C}(g))$, for each gate g of $\mathcal{C}$

- To show that *Shap* can be computed efficiently for dDBCs, we need a detailed analysis

- We assume the uniform distribution for the moment

- Theorem: *Shap* can be computed in polynomial time for dDBCs under the uniform distribution

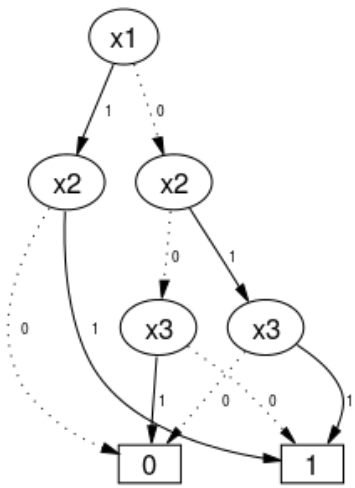
- It can be extended to any product distribution on $\{0,1\}^{|X|}$

- Corollary: Via polynomial time transformations, under the uniform and product distributions, *Shap* can be computed in polynomial time for
  - Decision trees (and random forests)
  - Ordered binary decision diagrams (OBDDs)

    $(\neg x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (x_1 \wedge x_2) \vee (x_2 \wedge x_3)$

    Compatible variable orders along full paths

    Compact representation of Boolean formulas



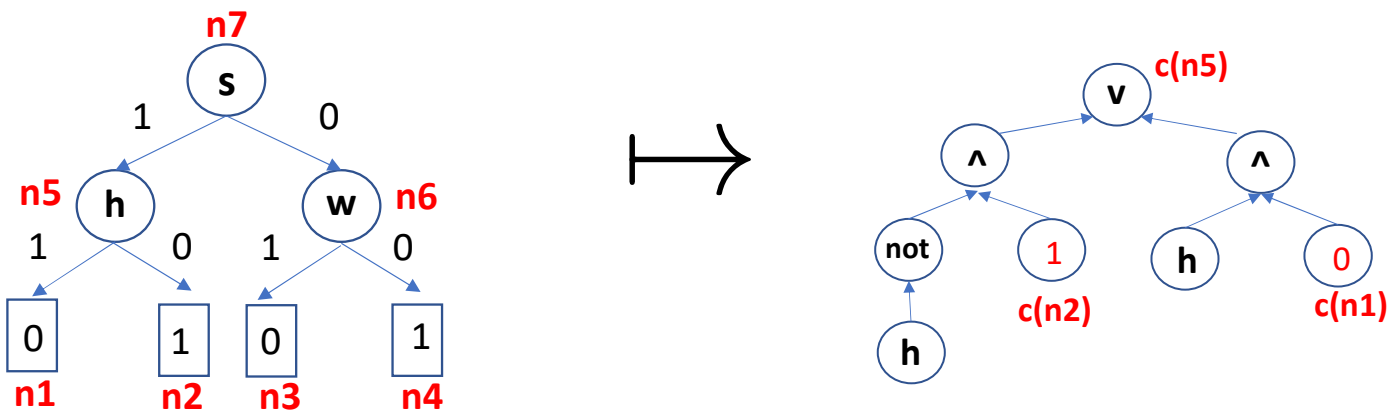
  - Sentential decision diagrams (SDDs)

    Generalization of OBDDs
  - Deterministic-decomposable negation normal-form (dDNNFs)

    As dDBC, with negations affecting only input variables
- All the latter relevant in *Knowledge Compilation*
- An optimized efficient algorithm for *Shap* computation can be applied to any of these

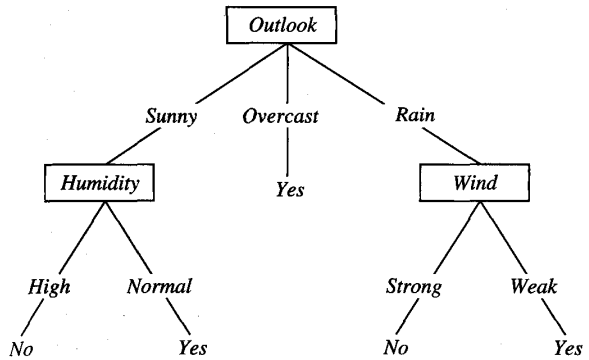
Shap for Decision Trees and ...

- Compiling binary decision trees into dDBCs
- An inductive construction starting from the bottom of the DT
- Leaves of DT become constant binary gates in dDBC
- By induction one can prove the resulting circuit is dDBC
- Final dDBC is the compilation $c(r)$ of root node r of DT

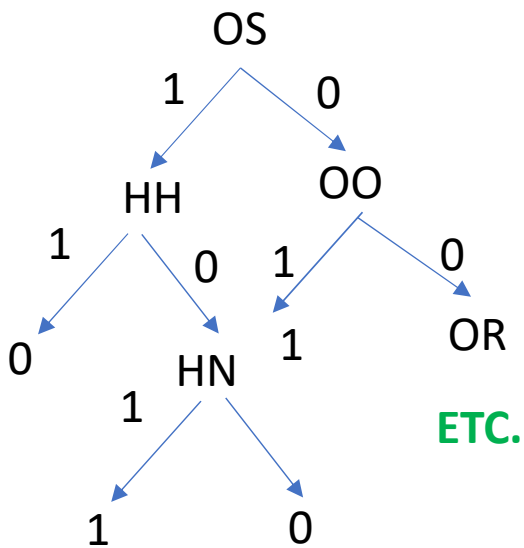


- Final equivalent dDBC: $c(n7)$
- Computable in linear time

- Beyond binary features?



- "Binarize" features
- *OutlookSunny* (OS)
  OutlookOvercast, *OutlookRain*, etc.
  become propositional features



Certain entities become impossible (probability 0)

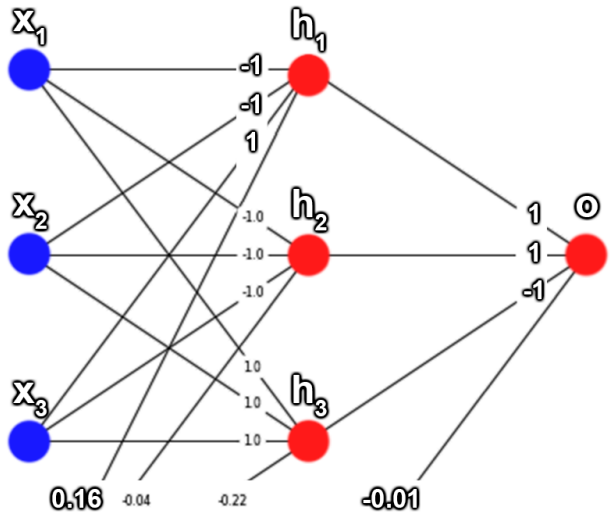
$\mathbf{e} = \langle \underbrace{0, 1, 1}_{\text{for OS, OO, OR}}, \ldots \rangle$ ×

$\mathbf{e} = \langle \underbrace{0, 1, 0}_{\text{for OS, OO, OR}}, \ldots \rangle$ *ok*

Shap on Neural Networks

- Binary Neural Networks (BNNs) are commonly considered black-box models
- Naively computing *Shap* on a BNN is bound to be complex
- Better try to compile the BNN into an open-box BC where *Shap* can be computed efficiently
- We have experimented with *Shap* computation with a black-box BNN and with its compilation into a dDBC[4]
- Even if the compilation is not entirely of polynomial time, it may be worth performing this one-time computation
- Particularly if the target dDBC will be used multiple times, as is the case for explanations
- We illustrate the approach by means of an example

[4] Bertossi, Leon; JELIA'23



$$\phi_g(\bar{i}) \quad = \quad sp(\bar{w}_g \bullet \bar{i} + b_g)$$

$$:= \quad \begin{cases} 1 & \text{if } \bar{w}_g \bullet \bar{i} + b_g \geq 0, \\ -1 & \text{otherwise,} \end{cases}$$

- The BNN is described by means of a propositional formula, which is further transformed and optimized into CNF

$$\begin{aligned} o \longleftrightarrow\ & (-[(x_3 \wedge (x_2 \vee x_1)) \vee (x_2 \wedge x_1)] \wedge \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \vee \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)])) \vee \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \wedge \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)]). \end{aligned}$$

In CNF:

$$o \longleftrightarrow (-x_1 \vee -x_2) \wedge (-x_1 \vee -x_3) \wedge (-x_2 \vee -x_3)$$

- The CNF is transformed into an SDD

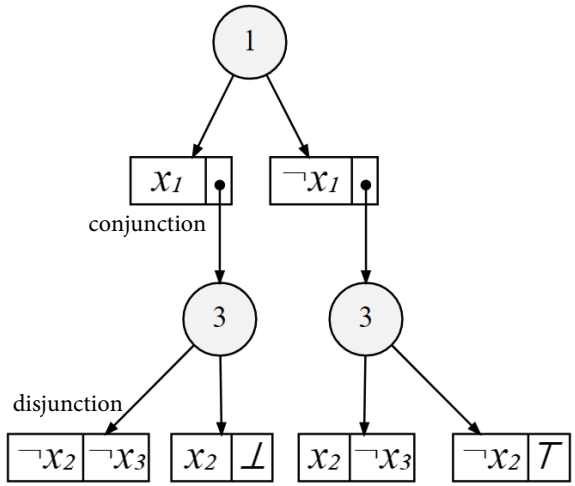
  It succinctly represents the CNF

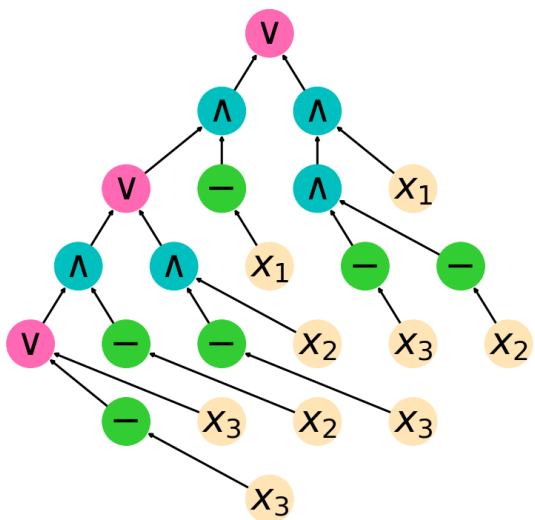
- The expensive compilation step

  But upper-bounded by an exponential only in the tree-width of the CNF

  TW of the associated undirected graph: an edge between variables if together in a clause

  A measure of how close it is to a tree
  (In example, graph is clique, TW is #vars -1 =2)

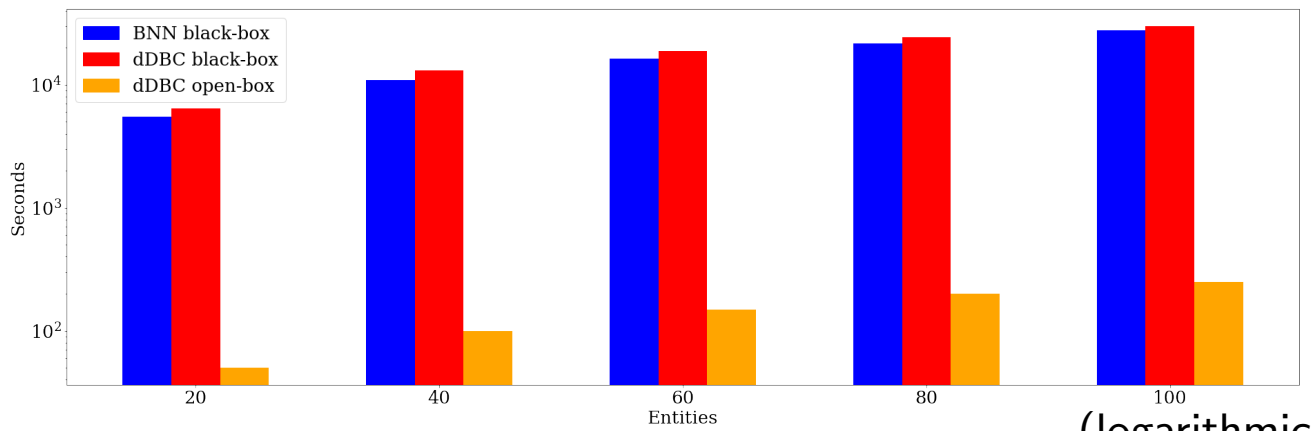




- The SDD is easily transformed into a dDBC
- On it *Shap* is computed, possibly multiple times
- With considerable efficiency gain

- In our experiments, we used a BNN with 14 gates
- It was compiled into a dDBC with 18,670 nodes
- A one-time computation that fully replaces the BNN
- We compared *Shap* computation time for black-box BNN, open-box dDBC, and black-box dDBC

  Total time for computing all Shap scores with increasing number of classification inputs



(logarithmic scale)

- The uniform distribution was used

Some Remarks

- The above results on *Shap* computation hold under the uniform and product distributions

  The latter imposes independence among features

- Other distributions have been considered for *Shap* and other scores

  The empirical and product-empirical distributions

  They naturally arise when no more information available about the distribution

- Imposing domain semantics (domain knowledge) is relevant

- Can we modify *Shap* definition and computation accordingly?

  Or the distribution?

- Do we still have an efficient algorithm?

Some Research Directions

- Explanation scores commonly use the classifier plus a probability distribution over the underlying entity population

  Imposing or using explicit and additional domain semantics or domain knowledge is relevant to explore

  Can we modify *Shap*'s definition and computation accordingly?

  Or the probability distribution?

- Shapley values satisfy desirable properties for general coalition game theory

  Specific properties for Explanations Scores (in AI)?

  Existing scores have been criticized or under-explored in terms of general properties

- Features (in ML and in general) may be hierarchically ordered according to categorical dimensions

  $$\text{address} \rightarrow \text{neighborhood} \rightarrow \textit{city} \rightarrow \cdots$$

  We may want to define and compute explanations (scores) at different levels of abstraction

  How to do this in a systematic way, possibly reusing results at different levels?

  Multi-dimensional explanations?

- There is a need for principled and sensible algorithms for explanations and score aggregation

  At the individual level as in ~~(3)~~ or at the group level, e.g. categories of instances 

  Hopefully guided by a declarative and flexible specifications (about what to aggregate and at which level)

EXTRA SLIDES

The Shapley Value

Consider a set $\mathbf{X} = \{X_1, \ldots, X_n\}$ of n *agents* or *variables* or *features*, and a *utility function* $g : 2^{\mathbf{X}} \to \mathbb{R}$, and define the Shapley and Banzhaf values as:

$$\phi_{\mathbf{X},g}(X_j) = \frac{1}{n!} \sum_{\pi \in \Pi_n} \left(g(\pi^{<X_j} \cup \{X_j\}) - g(\pi^{<X_j}) \right) \tag{1}$$

$$\beta_{\mathbf{X},g}(X_j) = \frac{1}{2^n} \sum_{S \subseteq \mathbf{X}} (g(S \cup \{X_j\}) - g(S)) \tag{2}$$

where Π_n is the set of permutations of $\mathbf{X}$, and $\pi^{<X_j}$ is the set of agents that come before X_j in the permutation π

More generally, given $n+1$ numbers $a_0, \ldots, a_n \in \mathbb{R}$, we define the *generalized value* as

$$\gamma_{\mathbf{x},g}(X_j) = \sum_{S \subseteq \mathbf{x}} a_{|S|} \left(g(S \cup \{X_j\}) - g(S) \right) \tag{3}$$

The Shapley and Banzhaf values are the special cases $a_k := k!(n-k-1)!$ and $a_k := 1$ respectively

The Shapley value is the unique value satisfying:

Efficiency $\sum_{j=1,m} \phi_{\mathbf{X},g}(X_j) = g(\mathbf{X})$

Symmetry $\phi_{\mathbf{X},g}(X_i) = \phi_{\mathbf{X},g}(X_j)$ whenever X_i, X_j are interchangeable, meaning that $g(S \cup \{X_i\}) = g(S \cup \{X_j\})$ for all sets S not containing X_i, X_j

Dummy $\phi_{\mathbf{X},g}(X_i) = 0$ if i does not contribute to the utility function, i.e. $g(S \cup \{X_i\}) = g(S)$ forall S

Additivity $\phi_{\mathbf{X},g_1+g_2} = \phi_{\mathbf{X},g_1} + \phi_{\mathbf{X},g_2}$

The Banzhaf value satisfies all properties above except for efficiency

The utility function $g : 2^{\mathbf{X}} \to \mathbb{R}$ is exponentially large

Various applications restrict the way the function is specified

- We can compute the expectation of the label: $\phi_0(L) = \mathbb{E}(L)$, which will give a value in $[0, 1]$, say 0.4
- Now fix $\mathbf{e}^\star$, for which we have the label $L(\mathbf{e}^\star)$, e.g. 1

  We want to account for the difference between this label and $\phi_0(L)$: $L(\mathbf{e}^\star) - \phi_0(L) = 1 - 0.4 = 0.6$
- The question is which feature value contributes the most to the difference $L(\mathbf{e}^\star) - \phi_0(L)$
  In our experiments we usually concentrate on entities $\mathbf{e}^\star$ with $L(\mathbf{e}^\star) = 1$, that means "rejection", which has to be explained
- The difference is expressed as a sum of individual contributions, $\phi_i(L, \mathbf{e}^\star)$, from the different features $F_i \in \mathcal{F}$:

$$\sum_i \phi_i(L, \mathbf{e}^\star) = L(\mathbf{e}^\star) - \phi_0(L)$$