Counterfactual-Based Explanations for Classification Outcomes

Semantics and Reasoning

Leopoldo Bertossi

bertossi@scs.carleton.ca

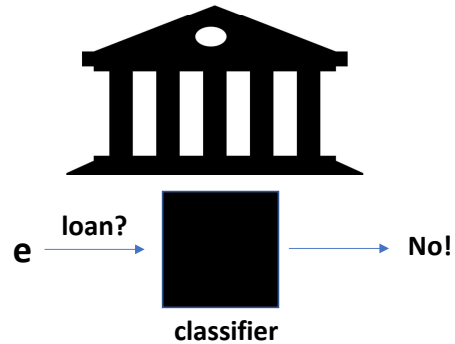
2022 Samsung AI Neuro-Symbolic WS "When Deep Learning Meets Logic"

Explanations in Machine Learning

- Client **e** requests a loan from a bank that uses a black-box classifier
- As an entity represented as a record of feature values:

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

  For Name, Age, Activity, Income, ...



- Which are the most relevant feature values for the classification outcome with label "No"?

  What is the quantitative contribution of each feature value to the outcome?

- A particular but not uncommon form of explanation for an outcome from an ML model

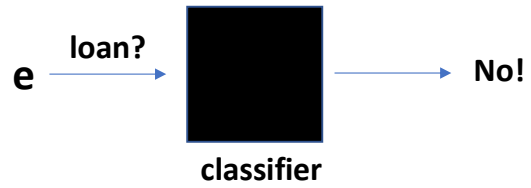
  We will concentrate on this kind of explanations

A Score-Based Approach

- We mentioned two crucial issues:
  1. Relevance, and
  2. An associated Degree of Contribution
- Without being the only possible way, we will approach them from the side of Actual Causality (Halpern & Pearl, 2001)
- We identify relevant feature values as actual causes

  Assign numerical scores to them on the basis of Causal Responsibility, a measure of causal contribution (Chockler & Halpern, 2004)
- Actual causes identified via Counterfactual Interventions

  Hypothetically reasoning: Would the change of this (these) feature value(s) lead to a change of label?



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

- 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 Age is counterfactual cause with explanatory responsibility $x\text{-}Resp(\text{Age}) = 1$
  2. Value for Income is actual cause with $x\text{-}Resp(\text{Income}) = \frac{1}{2}$

     This one needs additional changes ...

- Still, we might "prefer" the second explanation, actually its counterfactual version

  We could do something with this explanation ...

- Responsibility would be one form of attribution score

  There are others, most prominently *Shap*

- This counterfactual and score-based approach can be used with open and black-box models

  We do not need the internals of the model, only the input/output relation

- Having the internals of the model can lead to much faster score computation

  What we did for *Shap* and a certain class of Boolean circuits as classifiers (including decision trees, OBBDs, etc.)

  (Arenas, Barcelo, Bertossi, Monet, AAAI'21)

- We will concentrate on *x-Resp*

  Its counterfactual component is quite explicit ...

The Need for Reasoning

- We (or intelligent systems) receive attribution scores, and counterfactual explanations

  What do we do with them?

- We can reason about/with them, analyze them, select some of them, aggregate them, etc.

  In interaction with both attribution-score model/algorithm and classifier for further exploration

- We need tools for conveying or imposing domain knowledge (domain semantics)

  Possibly only some counterfactuals make sense

  Some combinations of feature values may not be allowed

  Some changes may "trigger" other changes

  To impose preferences on counterfactuals

- We need tools for doing reasoning

  Some sort of logical reasoning
- We need tools for querying explanations

  Are there explanations with this particular property?

  Or any two that differ by ...?
- Specification of high-score actionable explanations (resources)

  We want explanations we can do something with/about

  To get the loan, I cannot decrease my age, but I could try to change my job ...

  We may want to compute those only

  Or others with a different preferred property
- On-the-fly interaction with different ML models and scores

  Do I get same score with this different ML system?

  Or this other attribution score (definition, algorithm or implementation)?

- Imposing conditions on feature values

  What if I leave some feature values fixed?

  Do I get same high-score feature with this "similar" entity?

  Is there a high-score counterfactual version of the entity that changes this specific feature?

  Or never changes that one?

  Why do I get this high score for this feature value?

  Higher level "why" ...

- Summarizing, we need to:
  1. Specify counterfactual interventions
  2. Compute responsibility scores, and explanations
  3. Reason about them, and about explanations
  4. Answer different questions (queries)

Enabling Reasoning

- We need a logic and a general reasoning system for it
  - Supporting the desiderata above
  - Has the right constructs
  - Has the right reasoning and computational power
  - Without overkilling the tasks
- We know that reasoning/computational tasks belong to 2nd level of the polynomial hierarchy (in data)
- We need in particular:
  - Declarative language, and reasoning via QA
  - Possibly several models (representing counterfactuals)
  - Minimality of models, and closed-world assumption
  - Non-monotonicity, and commonsense reasoning (persistence)
  - Program constraints (domain semantics and model preference)
  - Extensions: weak constraints, set and numerical aggregations, interaction with external programs (classifiers)

- We have used Answer-Set Programming (ASP)
  (Gelfond & Lifschitz, 1991)

  A form of logic programming, with stable model semantics

  Disjunctive rule heads come handy

  Well known in the KR community

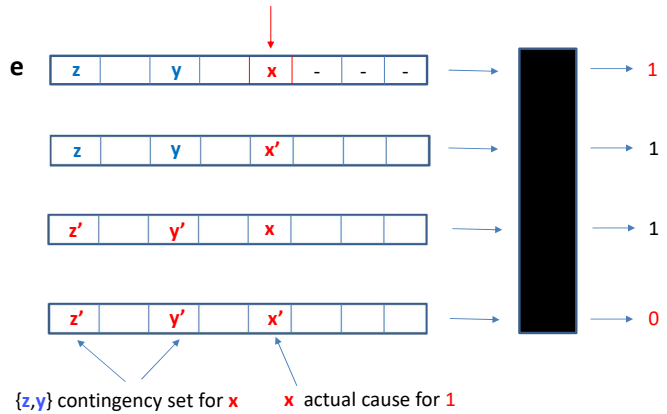
  For KR, reasoning, and solving combinatorial problems

- We used the *DLV System* (with its extensions)

  Developed at the U. of Calabria and T. U. Vienna

The *x-Resp* Score

x-Resp score for value of feature $F^\star$: (simplified version)



- Want explanation for label "1"
- Through value changes for feature $F^\star$, try to get "0"
- Feature value $\mathbf{x} = \mathbf{e}_{F^\star}$

- $\mathbf{x}$ counterfactual explanation for $L(\mathbf{e}) = 1$ if $L(\mathbf{e}\frac{\mathbf{x}}{\mathbf{x}'}) = 0$, for some $\mathbf{x}' \in Dom(F)$
- $\mathbf{x}$ actual explanation for $L(\mathbf{e}) = 1$ if there are values $\mathbf{Y}$ in $\mathbf{e}$, $\mathbf{x} \notin \mathbf{Y}$, and new values $\mathbf{Y}' \cup \{\mathbf{x}'\}$:

  (a) $L(\mathbf{e}\frac{\mathbf{Y}}{\mathbf{Y}'}) = 1$ (b) $L(\mathbf{e}\frac{\mathbf{x}\mathbf{Y}}{\mathbf{x}'\mathbf{Y}'}) = 0$

- For minimum-size contingency set $\mathbf{Y}$: $x\text{-}Resp(F^\star, \mathbf{x}) := \frac{1}{1+|\mathbf{Y}|}$

- We are usually interested in maximum-responsibility feature values

  Associated to minimum (cardinality) contingency sets of feature values

- Sometimes we may be interested in minimal contingency sets, under set-inclusion

  For non-maximum responsibility feature values

- For non-binary features, *Resp* may be better expressed as an expected value (Bertossi, Li, Schleich, Suciu, Vagena; 2020)

Specifying Counterfactual Interventions

Joint work with Gabriela Reyes (PhD student, UAI)

- Reason in ASP about counterfactuals

  In interaction with the classifier

  Specified inside the ASP, or invoked as external predicate

- **Counterfactual Intervention Programs** (CIPs)

  Specify counterfactual interventions on an entity under classification (Bertossi; TPLP'21)

- ASP programs use rules of the form:

  $$D_1(...) \vee \cdots \vee D_n(...) \longleftarrow P_1(...), \ldots, P_k(...), \mathit{not}\ N_1(...), \ldots, \mathit{not}\ N_m(...)$$

- ASP programs may have several (intended) models: answer-sets (or stable models)

- Rule with empty head are program constraints: models are not allowed to satisfy the RHS

  Those that do are eliminated

- We will use *DLV* and *DLV-Complex* notation and implementations
- Easily impose semantic constraints on counterfactuals
- Each counterfactual version leading to a new label corresponds to a model
- Scores can be computed by means of set- and numerical aggregations

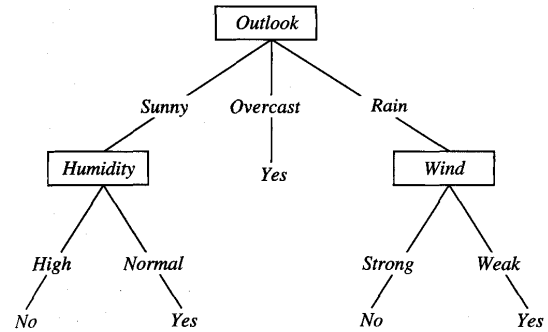
  For minimal and minimum contingency sets

  Supported by *DLV-Complex*
- Reasoning is enabled by cautious and brave query answering

  True in all models vs. true in some model
- Here we will classify and interact with decision-trees

  For naive-Bayes classifiers, c.f. (Bertossi & Reyes, IJCLR'21)

- A decision tree (classic example)



Features $\mathcal{F} = \{\text{Outlook}, \text{Humidity}, \text{Wind}\}$

$Dom(\text{Outlook}) = \{\text{sunny}, \text{overcast}, \text{rain}\}$

$Dom(\text{Humidity}) = \{\text{high}, \text{normal}\}$

$Dom(\text{Wind}) = \{\text{strong}, \text{weak}\}$

Entity $\mathbf{e} = ent(\text{sunny}, \text{normal}, \text{weak})$ gets label Yes (1)

- Want to change label to No (0)

  By successive attribute value changes (interventions)

- CIPs use annotation constants:

Annotation	Intended Meaning
o	original entity
do	do counterfactual intervention (change one feature value)
tr	entity in transition
s	stop, label has changed

$E(..., \texttt{o})$, $E(..., \texttt{do})$, $E(..., \texttt{tr})$, $E(..., \texttt{s})$

- Specifying domains, entity, classification tree, annotations:

```
% facts:
   dom1(sunny). dom1(overcast). dom1(rain). dom2(high). dom2(normal).
   dom3(strong). dom3(weak).
   ent(e,sunny,normal,weak,o).   % original entity at hand

% specification of the decision-tree classifier:
   cls(X,Y,Z,1) :- Y = normal, X = sunny, dom1(X), dom3(Z).
   cls(X,Y,Z,1) :- X = overcast, dom2(Y), dom3(Z).
   cls(X,Y,Z,1) :- Z = weak, X = rain, dom2(Y).
   cls(X,Y,Z,0) :- dom1(X), dom2(Y), dom3(Z), not cls(X,Y,Z,1).

% transition rules: the initial entity or one affected by a value change
   ent(E,X,Y,Z,tr) :- ent(E,X,Y,Z,o).
   ent(E,X,Y,Z,tr) :- ent(E,X,Y,Z,do).
```

- Next is the counterfactual rule (the most important one)

```
% counterfactual rule: alternative single-value changes
   ent(E,Xp,Y,Z,do) v ent(E,X,Yp,Z,do) v ent(E,X,Y,Zp,do) :-
                      ent(E,X,Y,Z,tr), cls(X,Y,Z,1), dom1(Xp), dom2(Yp),
                      dom3(Zp), X != Xp, Y != Yp, Z!= Zp,
                      chosen1(X,Y,Z,Xp), chosen2(X,Y,Z,Yp),
                      chosen3(X,Y,Z,Zp).
```

Only one disjunct in the head becomes true; one per feature

It uses the non-deterministic choice predicate

Chooses a new value in last argument for each combination of the first three

While the label stays 1 (yes)

- Specification of the three choice predicates:

```
% definitions of "chosen" predicates:
   chosen1(X,Y,Z,U) :- ent(E,X,Y,Z,tr), cls(X,Y,Z,1), dom1(U), U != X,
                       not diffchoice1(X,Y,Z,U).
   diffchoice1(X,Y,Z, U) :- chosen1(X,Y,Z, Up), U != Up, dom1(U).
```

ETC.

  Makes the program non-stratified (recursion via negation)

- A program constraint prohibiting going back to initial entity

```
% Not going back to initial entity (program constraint):
   :- ent(E,X,Y,Z,do), ent(E,X,Y,Z,o).
```

  Eliminates models that violate it

  Also contributes to non-stratification

- Non-stratified negation is useful/needed
- Rule defining "stop" annotation, when label becomes 0

```
% stop when label has been changed:
   ent(E,X,Y,Z,s) :- ent(E,X,Y,Z,do), cls(X,Y,Z,0).
```

  Last and desirable version of original entity

- Each counterfactual version represented by a model

- Models where entity does not change label can be discarded via a program constraint

```
entAux(E) :- ent(E,X,Y,Z,s).          % auxiliary predicate to
                                      % avoid unsafe negation
                                      % in the constraint below
:- ent(E,X,Y,Z,o), not entAux(E).     % discard models where
                                      % label does not change
```

- Rules for collecting value changes per feature

```
% collecting changed values for each feature:
   expl(E,outlook,X)   :- ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), X != Xp.
   expl(E,humidity,Y)  :- ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), Y != Yp.
   expl(E,wind,Z)      :- ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), Z != Zp.
```

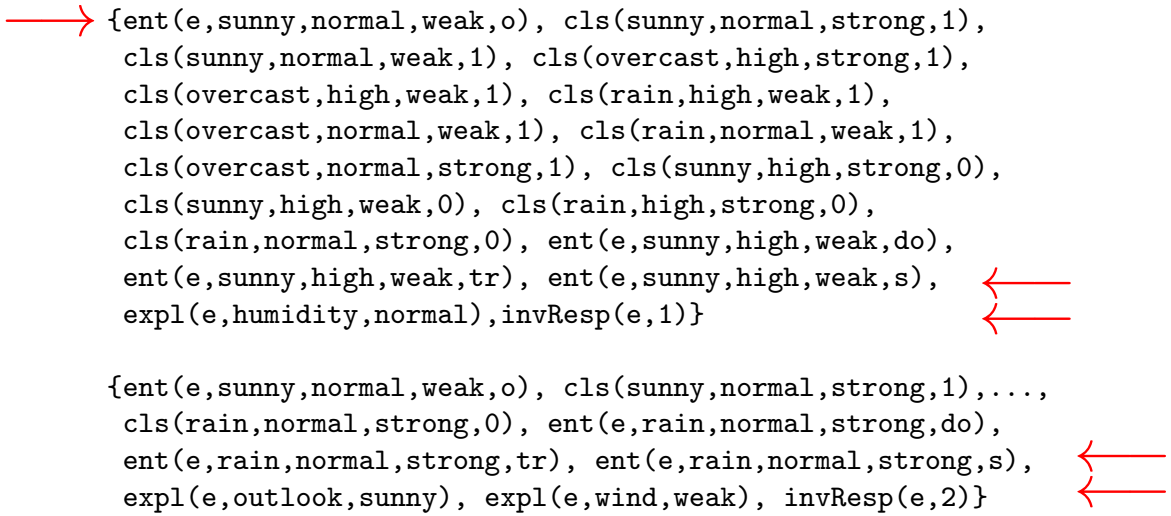
  Sets of changes (in each model) is minimal (for free with ASP)

- Rule with aggregation for counting number of feature value changes

```
% computing the inverse of x-Resp:
   invResp(E,M) :- #count{I: expl(E,I,_)} = M, #int(M), E = e.
```

- For each counterfactual version (or model) this is a "local" x-Resp-score associated to a minimal set of changes

  Not necessarily the "global" Resp-score yet



- Two stable models of the CIP
- Two counterfactual versions with minimal contingency sets
- Only first is minimum counterfactual version: x-Resp($\mathbf{e}$) $= 1$
- Want only maximum responsibility counterfactual versions?

```
{ent(e,sunny,normal,weak,o), cls(sunny,normal,strong,1),
 cls(sunny,normal,weak,1), cls(overcast,high,strong,1),
 cls(overcast,high,weak,1), cls(rain,high,weak,1),
 cls(overcast,normal,weak,1), cls(rain,normal,weak,1),
 cls(overcast,normal,strong,1), cls(sunny,high,strong,0),
 cls(sunny,high,weak,0), cls(rain,high,strong,0),
 cls(rain,normal,strong,0), ent(e,sunny,high,weak,do),
 ent(e,sunny,high,weak,tr), ent(e,sunny,high,weak,s),
 expl(e,humidity,normal),invResp(e,1)}
```

√

```
{ent(e,sunny,normal,weak,o), cls(sunny,normal,strong,1),...,
 cls(rain,normal,strong,0), ent(e,rain,normal,strong,do),
 ent(e,rain,normal,strong,tr), ent(e,rain,normal,strong,s),
 expl(e,outlook,sunny), expl(e,wind,weak), invResp(e,2)}
```

×

- Introduce weak program constraints

```
% Weak constraints to minimize number of changes:
   :~ ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), X != Xp.
   :~ ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), Y != Yp.
   :~ ent(E,X,Y,Z,o), ent(E,Xp,Yp,Zp,s), Z != Zp.
```

Weak program constraints can be violated, but only a minimum number of times

Minimize number of feature value differences between **e** and counterfactual versions

Only first model is kept (and gives global responsibility)

Domain Knowledge and Extensions

- Adding domain knowledge is easy

  There may never be rain with strong wind

  Discard the model:

```
% hard constraint disallowing a particular combination
  :- ent(E,rain,X,strong,tr).
```

- Constraint has the effect of deleting models with this combination

  Computationally better: compile constraints into rule bodies (conditions) (avoiding construction of models that will be discarded)

- Similarly for specifying actionability
- CIPs are quite generic

  Most of their ingredients are general

  Domain/application independent

- We have used DLV in interaction with an external classifier programmed in Python

  DLV offers the right interface

Reasoning via QA

- Counterfactuals can be queried

  Reasoning enabled by query answering

- Under cautious and brave semantics:

  - *Responsibility of feature* Outlook*?*
  - *A counterfactual version with less than 3 changes?*

```
invResp(e,outlook,R)?                          (brave semantics)
fullExpl(E,U,R,S), R<3?
```

  - *An intervened entity with combination of sunny outlook and strong wind, and its label?*
  - *All intervened entities that obtain label* No*?*

```
cls(E,O,T,H,W,_), O = sunny, W = strong?
cls(E,O,T,H,W,no)?
```

  - *Does the wind not change under every counterfactual version?*

```
ent(e,_,_,_,Wp,s), ent(e,_,_,_,W,o), W = Wp?   (cautious semantics)
```

Final Remarks

- Addition of semantic and domain knowledge is important
- Reasoning in general about scores, explanations and counterfactuals is what intelligent agents do

  Higher-level analytics and reasoning should be characterized, formalized and automated:

  - What can I learn through aggregation of attribution scores?
  - Defining and aggregating at higher levels of abstraction

    Categorizing features at a higher level:

    "Your entire socio-economic situation is to be blamed for the rejection of your loan application"
  - Another higher-level ML-system that learns from attribution scores (numbers)?
  - Learning about the application domain and/or the lower level ML system

- Explanations are at the basis of fairness and bias analysis

  Identifying unexpected or undesirable high-score features becomes relevant

- But possibly not enough

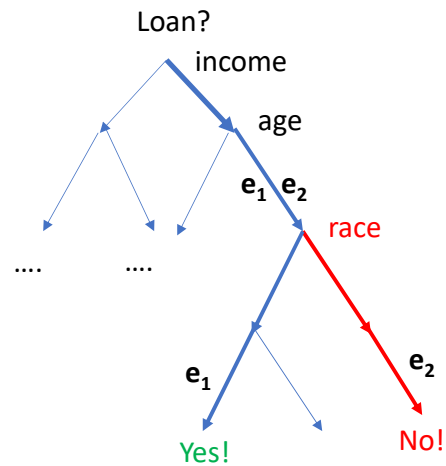
  Understanding decisions in relation to protected features becomes relevant

- Identifying undesirable decisions

- We can query for (the existence of) these cases

- ASP provides support for this

  By keeping track of counterfactual "histories" and their comparison



- Ongoing work:
  - Use probabilistic extensions of ASP with probabilistic ML
  - To impose statistical constraints on population

EXTRA PAGES

A Variation: No contingencies, but average labels

- For binary features the previous version of RESP works fine
- There could be many values that do not change the label, but one of them does

  Better consider all possible values; towards a generalization ...

  (Bertossi, Li, Schleich, Suciu, Vagena; 20)

- $\mathbf{e} = \langle \ldots, \mathbf{e}_F, \ldots \rangle, \quad F \in \mathcal{F}$

  $$\mathit{Counter}(\mathbf{e}, F) := L(\mathbf{e}) - \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}'_{\mathcal{F} \setminus \{F\}} = \mathbf{e}_{\mathcal{F} \setminus \{F\}})$$

- Easy to compute, and gives reasonable results

  Requires underlying probability space on entity population

  No need to access the internals of the classification model

- Changing one value may not switch the label

  No explanations are obtained

- Bring in contingency sets of feature values!

General Version: Contingencies and average labels

- $\mathbf{e}$ entity under classification, with $L(\mathbf{e}) = 1$, and $F^\star \in \mathcal{F}$
- Local *Resp*-score $Resp(\mathbf{e}, F^\star, \mathcal{F}, \Gamma, \bar{w})$
  1. $\Gamma \subseteq \mathcal{F} \setminus \{F^\star\}$
  2. $\mathbf{e}' := \mathbf{e}[\Gamma := \bar{w}]$ with $L(\mathbf{e}') = L(\mathbf{e})$ (no label change)
  3. $\mathbf{e}'' := \mathbf{e}[\Gamma := \bar{w}, F^\star := v]$, with $v \in dom(F^\star)$ (all possible)
- $Resp(\mathbf{e}, F^\star, \mathcal{F}, \Gamma, \bar{w}) := \dfrac{L(\mathbf{e}') - \mathbb{E}[L(\mathbf{e}'') \mid \mathbf{e}''_{\mathcal{F}\setminus\{F^\star\}} = \mathbf{e}'_{\mathcal{F}\setminus\{F^\star\}}]}{1+|\Gamma|}$ $(*)$
- (When $F^\star(\mathbf{e}) \neq v$, $L(\mathbf{e}'') \neq L(\mathbf{e})$, $F^\star(\mathbf{e})$ is *actual causal explanation* for $L(\mathbf{e}) = 1$ with contingency $\langle \Gamma, \mathbf{e}_\Gamma \rangle$)
- Globally: $Resp(\mathbf{e}, F^\star) := \max_{\substack{|\Gamma| \text{ min.}, (*)>0 \\ \langle \Gamma, \bar{w} \rangle}} Resp(\mathbf{e}, F^\star, \mathcal{F}, \Gamma, \bar{w})$