Probabilistic Explanations for Linear Models

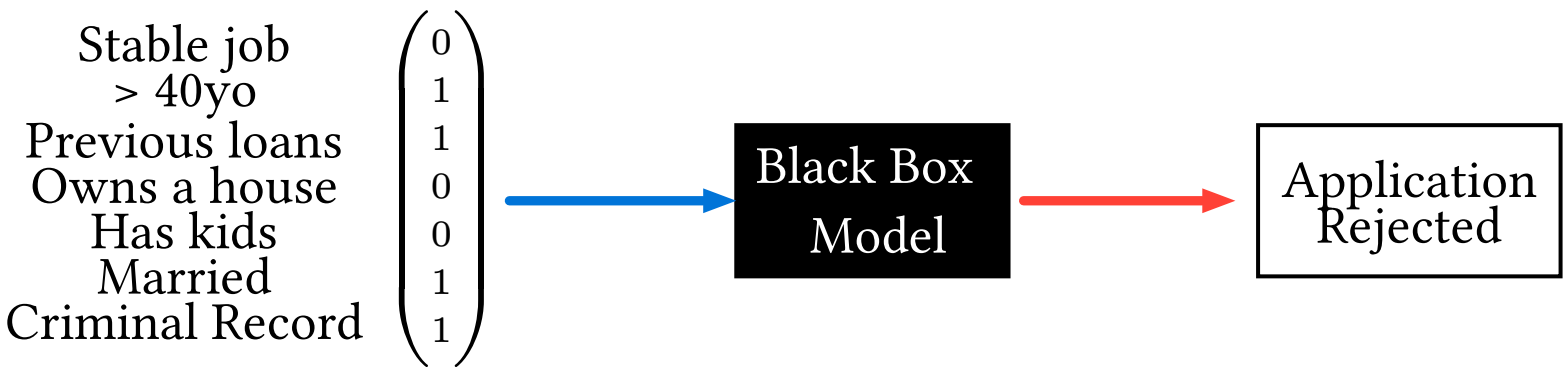
AAAI-25 / IAAI-25 / EAAI-25

FEBRUARY 25 – MARCH 4, 2025 | PHILADELPHIA, USA

M. Arenas, K. S Meel, and **B. Subercaseaux**

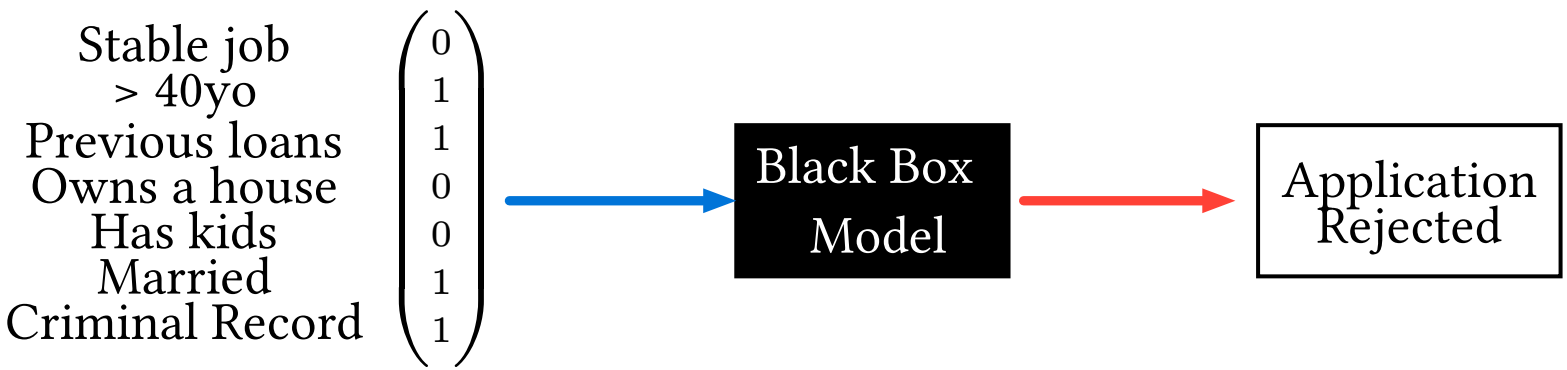
Context

A key problem in Explainable AI:



Context

A key problem in Explainable AI:



Question: Why was the application rejected?

Setting & Notation

- For mathematical simplicity we consider Boolean classification models:

$$F : \{0,1\}^d \to \{0,1\}.$$

Setting & Notation

- For mathematical simplicity we consider Boolean classification models:

$$F : \{0,1\}^d \to \{0,1\}.$$

- We will think of elements of $\{0,1,?\}^d$ as **"partial instances"**, that leave some features unspecified.

Setting & Notation

- For mathematical simplicity we consider Boolean classification models:

$$F : \{0,1\}^d \to \{0,1\}.$$

- We will think of elements of $\{0,1,?\}^d$ as **"partial instances"**, that leave some features unspecified.
- If an instance $\vec{x}$ in $\{0,1\}^d$ completes the information of a partial instance $\vec{y}$, we say its a **"completion"** of $\vec{y}$. The set of completions of $\vec{y}$ is denoted $\mathrm{Comp}(\vec{y})$, and if $\vec{x} \in \mathrm{Comp}(\vec{y})$ we also write $\vec{y} \subseteq \vec{x}$.

Setting & Notation

- For mathematical simplicity we consider Boolean classification models:

$$F : \{0,1\}^d \to \{0,1\}.$$

- We will think of elements of $\{0,1,?\}^d$ as **"partial instances"**, that leave some features unspecified.
- If an instance $\vec{x}$ in $\{0,1\}^d$ completes the information of a partial instance $\vec{y}$, we say its a **"completion"** of $\vec{y}$. The set of completions of $\vec{y}$ is denoted $\mathrm{Comp}(\vec{y})$, and if $\vec{x} \in \mathrm{Comp}(\vec{y})$ we also write $\vec{y} \subseteq \vec{x}$.

Example 1.

$$\mathrm{Comp}\left(\begin{pmatrix}1\\?\\0\\?\end{pmatrix}\right) = \left\{\begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\\1\end{pmatrix}, \begin{pmatrix}1\\1\\0\\0\end{pmatrix}, \begin{pmatrix}1\\1\\0\\1\end{pmatrix}\right\}$$

Setting & Notation (ii)

Definition: Sufficient Reason.

A partial instance $\vec{y} \subseteq \vec{x}$ is a sufficient reason (SR) for the decision $F(\vec{x})$ iff

$$F(\vec{z}) = F(\vec{x}) \text{ for every } \vec{z} \in \text{Comp}(\vec{y}).$$

Setting & Notation (ii)

Definition: Sufficient Reason.

A partial instance $\vec{y} \subseteq \vec{x}$ is a sufficient reason (SR) for the decision $F(\vec{x})$ iff

$$F(\vec{z}) = F(\vec{x}) \text{ for every } \vec{z} \in \text{Comp}(\vec{y}).$$

Example 2.

Suppose the bank's model is a linear model of the form

$$F(\vec{x}) = \begin{cases} 1 \quad \text{if} \quad 7\vec{x}_1 - 2\vec{x}_2 + 20\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - 30\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Then, $(?,?,0,?,?,?,1)$ is a sufficient reason for rejecting $(0,1,1,0,0,1,1)$. The applicant was rejected **because they have a criminal record and don't own a house**.

Setting & Notation (ii)

Definition: Sufficient Reason.

A partial instance $\vec{y} \subseteq \vec{x}$ is a sufficient reason (SR) for the decision $F(\vec{x})$ iff

$$F(\vec{z}) = F(\vec{x}) \text{ for every } \vec{z} \in \text{Comp}(\vec{y}).$$

Example 2.

Suppose the bank's model is a linear model of the form

$$F(\vec{x}) = \begin{cases} 1 \text{ if } 7\vec{x}_1 - 2\vec{x}_2 + 20\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - 30\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Then, $(?,?,0,?,?,?,1)$ is a sufficient reason for rejecting $(0,1,1,0,0,1,1)$. The applicant was rejected **because they have a criminal record and don't own a house**.

Note: the full instance $\vec{x}$ is always a (trivial) SR for its own classification; in practice we are interested in **small** SRs.

Setting & Notation (iii)

Different ways to formalize that an SR is small:

Definition: Minimal SR.

A sufficient reason $\vec{y}$ for the decision $F(\vec{x}) = b$ is **minimal** if no strict subset of $\vec{y}$ is also a sufficient reason.

Namely, no more question marks '?' can be added.

Setting & Notation (iii)

Different ways to formalize that an SR is small:

Definition: Minimal SR.

A sufficient reason $\vec{y}$ for the decision $F(\vec{x}) = b$ is **minimal** if no strict subset of $\vec{y}$ is also a sufficient reason.

Namely, no more question marks '?' can be added.

Definition: Minimum SR.

A sufficient reason $\vec{y}$ for the decision $F(\vec{x}) = b$ is a **minimum sufficient reason** if it uses the minimum number of features of $\vec{x}$, or equivalently, the maximum number of question marks.

These have been substantially studied for a variety of Boolean models!

Setting & Notation (iv)

A way to achieve even smaller SRs is by relaxing the definition:

Definition: Probabilistic Sufficient Reason (Wäldchen et al. (2021).

A partial instance $\vec{y} \subseteq \vec{x}$ is a δ-sufficient reason for the decision $F(\vec{x})$, with $\delta \in [0, 1]$ iff

$$\Pr_{\vec{z} \in \text{ Comp}(\vec{y})}[F(\vec{z}) = F(\vec{x})] \geq \delta.$$

That is, a uniformly random completion of $\vec{y}$ is "very likely" (δ) to have the same classification as $\vec{x}$.

Setting & Notation (iv)

A way to achieve even smaller SRs is by relaxing the definition:

Definition: Probabilistic Sufficient Reason (Wäldchen et al. (2021).

A partial instance $\vec{y} \subseteq \vec{x}$ is a δ-sufficient reason for the decision $F(\vec{x})$, with $\delta \in [0, 1]$ iff

$$\Pr_{\vec{z} \in \text{ Comp}(\vec{y})}[F(\vec{z}) = F(\vec{x})] \geq \delta.$$

That is, a uniformly random completion of $\vec{y}$ is "very likely" (δ) to have the same classification as $\vec{x}$.

Note: When $\delta = 1$ we recover the base definition of SR.

Setting & Notation (iv)

A way to achieve even smaller SRs is by relaxing the definition:

Definition: Probabilistic Sufficient Reason (Wäldchen et al. (2021).

A partial instance $\vec{y} \subseteq \vec{x}$ is a δ-sufficient reason for the decision $F(\vec{x})$, with $\delta \in [0, 1]$ iff

$$\Pr_{\vec{z} \in \text{ Comp}(\vec{y})}[F(\vec{z}) = F(\vec{x})] \geq \delta.$$

That is, a uniformly random completion of $\vec{y}$ is "very likely" (δ) to have the same classification as $\vec{x}$.

Note: When $\delta = 1$ we recover the base definition of SR. **Main prior results**:

- Wäldchen et al. proved NP^{PP}-completeness for computing over neural networks.

Setting & Notation (iv)

A way to achieve even smaller SRs is by relaxing the definition:

Definition: Probabilistic Sufficient Reason (Wäldchen et al. (2021).

A partial instance $\vec{y} \subseteq \vec{x}$ is a δ-sufficient reason for the decision $F(\vec{x})$, with $\delta \in [0, 1]$ iff

$$\Pr_{\vec{z} \in \text{ Comp}(\vec{y})}[F(\vec{z}) = F(\vec{x})] \geq \delta.$$

That is, a uniformly random completion of $\vec{y}$ is "very likely" (δ) to have the same classification as $\vec{x}$.

Note: When $\delta = 1$ we recover the base definition of SR. **Main prior results**:

- Wäldchen et al. proved NP^{PP}-completeness for computing over neural networks.
- Izza et al., studied practical algorithms and advanced the theory for several simple models, including linear models.

Prior work

Theorem 1: (Arenas et al., 2022).

If F is given as a decision tree:

- computing a minimum δ-SR is NP-hard for every $\delta \in (0, 1]$.

Prior work

Theorem 1: (Arenas et al., 2022).

If F is given as a decision tree:

- computing a minimum δ-SR is NP-hard for every $\delta \in (0, 1]$.
- computing a minimal δ-SR is NP-hard, when δ is part of the input. In contrast, the case $\delta = 1$ can be solved efficiently.

Prior work

Theorem 1: (Arenas et al., 2022).

If F is given as a decision tree:

- computing a minimum δ-SR is NP-hard for every $\delta \in (0, 1]$.
- computing a minimal δ-SR is NP-hard, when δ is part of the input. In contrast, the case $\delta = 1$ can be solved efficiently.

Theorem 2: (Kozachinskiy, 2023) Informal.

If F is given as a decision tree, then under standard complexity assumptions, one cannot even approximate decently: no polytime algorithm distinguishes between:

1. The case where a very small 0.99-SR exists,
2. The case where all 0.1-SRs are very large.

Prior work

Theorem 1: (Arenas et al., 2022).

If F is given as a decision tree:

- computing a minimum δ-SR is NP-hard for every $\delta \in (0, 1]$.
- computing a minimal δ-SR is NP-hard, when δ is part of the input. In contrast, the case $\delta = 1$ can be solved efficiently.

Theorem 2: (Kozachinskiy, 2023) Informal.

If F is given as a decision tree, then under standard complexity assumptions, one cannot even approximate decently: no polytime algorithm distinguishes between:

1. The case where a very small 0.99-SR exists,
2. The case where all 0.1-SRs are very large.

Intuition: In decision trees the different features are not independent, and thus to find a small "important" subset of the features it's not easy to do better than trying all subsets.

Problem

In contrast, in linear models the different features are treated independently, and we have a sense of which ones are the most important via the weights!

$$F(\vec{x}) = \begin{cases} 1 \quad \text{if} \quad 7\vec{x}_1 - 2\vec{x}_2 + \mathbf{20}\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - \mathbf{30}\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Problem

In contrast, in linear models the different features are treated independently, and we have a sense of which ones are the most important via the weights!

$$F(\vec{x}) = \begin{cases} 1 \quad \text{if} \quad 7\vec{x}_1 - 2\vec{x}_2 + \mathbf{20}\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - \mathbf{30}\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Main Question.

Can we compute small δ-SRs efficiently over linear models?

Problem

In contrast, in linear models the different features are treated independently, and we have a sense of which ones are the most important via the weights!

$$F(\vec{x}) = \begin{cases} 1 \quad \text{if} \quad 7\vec{x}_1 - 2\vec{x}_2 + \mathbf{20}\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - \mathbf{30}\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Main Question.

Can we compute small δ-SRs efficiently over linear models?

Prior Answer: In practice Izza et al. have obtained good results., but theory was unclear.

Problem

In contrast, in linear models the different features are treated independently, and we have a sense of which ones are the most important via the weights!

$$F(\vec{x}) = \begin{cases} 1 \quad \text{if} \quad 7\vec{x}_1 - 2\vec{x}_2 + \mathbf{20}\vec{x}_3 + 5\vec{x}_4 + 2\vec{x}_5 + 3\vec{x}_6 - \mathbf{30}\vec{x}_7 \geq 10 \\ 0 \text{ otherwise.} \end{cases}$$

Main Question.

Can we compute small δ-SRs efficiently over linear models?

Prior Answer: In practice Izza et al. have obtained good results., but theory was unclear.
Our paper's answer: If you're pedantic, then it's $\#P$-hard, but if you're willing to relax a little bit then it can be done very efficiently!

The difficulty

Theorem 3: The negative result.

Given a linear model L, an input $\vec{x}$, and $\delta \in [0, 1]$, it's not possible to compute the size of the minimum δ-SR unless $P = \#P$.

The difficulty

Theorem 3: The negative result.

Given a linear model L, an input $\vec{x}$, and $\delta \in [0, 1]$, it's not possible to compute the size of the minimum δ-SR unless $P = \#P$.

The proof is based on the fact that, even if we know a good candidate explanation, computing its probability is $\#P$-hard, since it's basically #SubsetSum.

So the main barrier is computing the probability of $L(\vec{z}) = 1$.

The difficulty

Theorem 3: The negative result.

Given a linear model L, an input $\vec{x}$, and $\delta \in [0, 1]$, it's not possible to compute the size of the minimum δ-SR unless $P = \#P$.

The proof is based on the fact that, even if we know a good candidate explanation, computing its probability is $\#P$-hard, since it's basically #SubsetSum.

So the main barrier is computing the probability of $L(\vec{z}) = 1$.

In general, the difficulty of computing these probabilities is not to have a rough estimate, but rather to have a very small (e.g., 2^{-n}) margin of error.

The difficulty

Theorem 3: The negative result.

Given a linear model L, an input $\vec{x}$, and $\delta \in [0, 1]$, it's not possible to compute the size of the minimum δ-SR unless $P = \#P$.

The proof is based on the fact that, even if we know a good candidate explanation, computing its probability is $\#P$-hard, since it's basically #SubsetSum.

So the main barrier is computing the probability of $L(\vec{z}) = 1$.

In general, the difficulty of computing these probabilities is not to have a rough estimate, but rather to have a very small (e.g., 2^{-n}) margin of error.

Main Observation.

A user that wants a small 0.8-SR probably doesn't really care if it's a 0.8-SR or a 0.798-SR or a 0.804-SR

The Relaxation

Main Observation.

A user that wants a small 0.8-SR probably doesn't really care if it's a 0.8-SR or a 0.798-SR or a 0.804-SR

Based on this we define the following relaxation:

The Relaxation

Main Observation.

A user that wants a small 0.8-SR probably doesn't really care if it's a 0.8-SR or a 0.798-SR or a 0.804-SR

Based on this we define the following relaxation:

Definition: (δ, ε)-sufficient reason.

Given a function F, an input $\vec{x}$, and parameters $\delta, \varepsilon \in [0, 1]$, we say that a partial instance $\vec{y} \subseteq \vec{x}$ is a **minimum (δ, ε)-SR** for $F(\vec{x})$ if there exists $\delta^* \in [\delta - \varepsilon, \delta + \varepsilon]$ such that $\vec{y}$ is a minimum δ^*-SR for $F(\vec{x})$.

The Relaxation

Main Observation.

A user that wants a small 0.8-SR probably doesn't really care if it's a 0.8-SR or a 0.798-SR or a 0.804-SR

Based on this we define the following relaxation:

Definition: (δ, ε)-sufficient reason.

Given a function F, an input $\vec{x}$, and parameters $\delta, \varepsilon \in [0, 1]$, we say that a partial instance $\vec{y} \subseteq \vec{x}$ is a **minimum (δ, ε)-SR** for $F(\vec{x})$ if there exists $\delta^* \in [\delta - \varepsilon, \delta + \varepsilon]$ such that $\vec{y}$ is a minimum δ^*-SR for $F(\vec{x})$.

Suppose a user wants a 0.9-SR, and they really **need it to be above 90%** for compliance reasons. Then, by computing a minimum $(0.91, 0.01)$-SR, they will get a small SR whose probability is guaranteed to be above 90%. **Downside**: it could be a minimum 0.92-SR, which can be significantly larger than a minimum 0.9-SR.

Main result

Theorem 4.

Given

- a linear model L of dimension d,
- an input $\vec{x}$
- parameters $\delta, \varepsilon, \gamma \in [0, 1)$

there is an algorithm that computes a minimum (δ, ε)-SR, with probability of error γ, in time

$$O\left(\frac{d}{\varepsilon^2\gamma^2} \cdot \log^3(d) \cdot \log\log\left(\frac{d}{\gamma}\right)\right) = \tilde{O}\left(\frac{d}{\varepsilon^2\gamma^2}\right).$$

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.
- We do a binary search over k to find the smallest k such that

$$\Pr_{\vec{z} \in \text{Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})] \geq \delta^*.$$

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.
- We do a binary search over k to find the smallest k such that

$$\Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})] \geq \delta^*.$$

- We cannot compute $v_k := \Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})]$ directly, since its $\#P$-hard.

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.
- We do a binary search over k to find the smallest k such that

$$\Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})] \geq \delta^*.$$

- We cannot compute $v_k := \Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})]$ directly, since its $\#P$-hard.
- However, since we chose δ^* randomly, it's unlikely that δ^* is super close to some v_k. So if we get a decent estimate $\tilde{v_k}$ for v_k, we will most likely judge right whether $v_k \geq \delta^*$ by using the test $\tilde{v_k} \geq \delta^*$.

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.
- We do a binary search over k to find the smallest k such that

$$\Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})] \geq \delta^*.$$

- We cannot compute $v_k := \Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})]$ directly, since its $\#P$-hard.
- However, since we chose δ^* randomly, it's unlikely that δ^* is super close to some v_k. So if we get a decent estimate $\tilde{v_k}$ for v_k, we will most likely judge right whether $v_k \geq \delta^*$ by using the test $\tilde{v_k} \geq \delta^*$.
- Following Izza et al., we use Montecarlo estimation to compute $\tilde{v_k}$.

Intuition behind main theorem

- Sample a δ^* uniformly at random from $[\delta - \varepsilon, \delta + \varepsilon]$.
- Order all features according to the weights and whether they were 0 or 1 in input $\vec{x}$, that way we know which ones are the "top"-k most explanatory features for the decision $L(\vec{x})$. Let $\vec{y}_k$ be the partial instance formed by the top-k best features.
- We do a binary search over k to find the smallest k such that

$$\Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})] \geq \delta^*.$$

- We cannot compute $v_k := \Pr_{\vec{z} \in \text{ Comp}(\overrightarrow{y_k})}[L(z) = L(\vec{x})]$ directly, since its $\#P$-hard.
- However, since we chose δ^* randomly, it's unlikely that δ^* is super close to some v_k. So if we get a decent estimate $\tilde{v_k}$ for v_k, we will most likely judge right whether $v_k \geq \delta^*$ by using the test $\tilde{v_k} \geq \delta^*$.
- Following Izza et al., we use Montecarlo estimation to compute $\tilde{v_k}$.
- The main difficulty is arguing independences of random variables!