ExpProof : Operationalizing Explanations for Confidential Models with ZKPs

Chhavi Yadav*\$, Evan Monroe Laufer*#, Dan Boneh#, Kamalika Chaudhuri\$
\$UC San Diego, #Stanford University *equal contribution

Explanations failure mode

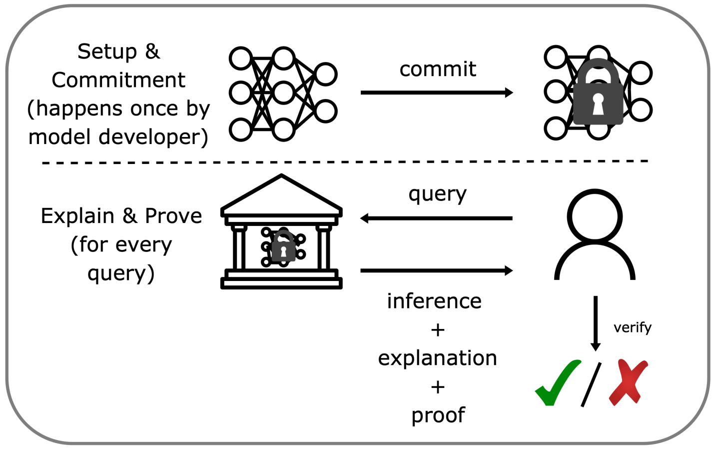
- Explanations are intended as a way to increase trust in ML models by making them transparent in societal applications and are often obligated by regulations.
- Many of these use-cases are adversarial in nature where the involved parties have misaligned interests and are incentivized to manipulate explanations to meet their ends.
- Consequently, despite the demand, explanations fail to be operational as a trust-enhancing tool.

Confidentiality aggravates the problem

- Additionally, organizations keep their models confidential due to IP & legal reasons.
- Leads to model swapping : for inference vs. explanation, for different inputs and post-audits. Moreover, there is no guarantee over how the explanation is generated.
- Canonical Solution : Consistency checks. But these have been shown to be hard. Require the customer to collect different (input, explanation) pairs which makes for a lopsided ask.

Our Solution

- **Public verification of explanations with cryptography**
- We propose ***ExpProof*** — a system that uses **Zero-Knowledge Proofs (ZKPs) & Commitments** to publicly verify explanations, while maintaining confidentiality.
- We also propose ZKP-efficient versions of the explanation algorithm.



(a) Schematic Diagram of ExpProof

Algorithm 1 LIME (Ribeiro et al., 2016)

Input: Input point x, Classifier f
Parameters: Number of points n to be sampled around input point, Length of explanation K, Bandwidth parameter σ for similarity kernel
Output: Explanation e

for $i \in \{1, 2, 3, \dots, n\}$ **do**
  $z_i \leftarrow \text{sample\_around}(x)$
  $\pi_i \leftarrow \exp\left(-\ell_2(x, z_i)^2/\sigma^2\right)$
end for
$\hat{w} \in \arg\min_w \sum_{i=1}^{n} \pi_i \times \left(f(z_i) - w^\top z_i\right)^2 + \|w\|_1$
$e := \text{top-K}(\hat{w}, K)$ ▷ Sorts the weights according to absolute value & returns these along with corresponding features
Return Explanation e

Algorithm 2 STANDARD_LIME_VARIANTS

Input: Input point x, Classifier f
Parameters: Number of points n to be sampled around input point, Length of explanation K, Bandwidth parameter σ for similarity kernel, sampling type $smpl\_type$, kernel type $krnl\_type$
Output: Explanation e

for $i \in \{1, 2, 3, \dots, n\}$ **do**
  if $smpl\_type$==‘uniform’ **then**
    $z_i \leftarrow \text{uniformly\_sample\_around}(x)$
  else if $smpl\_type$==‘gaussian’ **then**
    $z_i \leftarrow \text{gaussian\_sample\_around}(x)$
  end if
  if $krnl\_type$==‘exponential’ **then**
    $\pi_i \leftarrow \exp\left(-\ell_2(x, z_i)^2/\sigma^2\right)$
  else
    $\pi_i = 1$
  end if
end for
$\hat{w} \in \arg\min_w \sum_{i=1}^{n} \pi_i \times \left(f(z_i) - w^\top z_i\right)^2 + \|w\|_1$
$e := \text{top-K}(\hat{w}, K)$
Return Explanation e

(b) LIME and its variants

Algorithm 3 BORDERLIME

Input: Input point x, Classifier f
Parameters: Number of points n to be sampled around input point, Length of explanation K, Bandwidth parameter σ for similarity kernel
Output: Explanation e

$x_{border} :=$ FIND_CLOSEST_POINT_WITH_OPP_LABEL(x, f) ▷ See Alg. 4
$e := \text{LIME}(x_{border}, f)$ ▷ Note that any variant of LIME can be used here
Return Explanation e

(c) BorderLIME

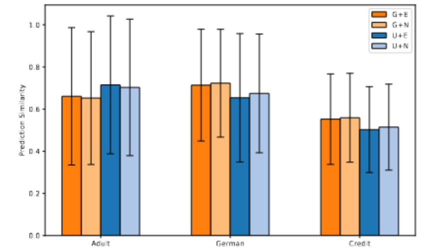
Algorithm 4 FIND_CLOSEST_POINT_WITH_OPP_LABEL

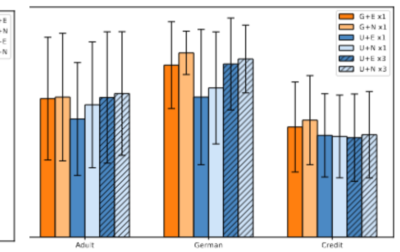
Input: Input point x, Classifier f
Parameters: Number of random directions m, Stability radius δ, Iteration Threshold T
Output: Opposite label point x_{border}

$\{\vec{u}_0, \vec{u}_1 \cdots \vec{u}_{m-1}\}$:= Sample m random directions
Initialize $dist_0 \cdots dist_{m-1}$ as inf
for $\vec{u}_i \in \{\vec{u}_0, \vec{u}_1 \cdots \vec{u}_{m-1}\}$ **do**
  $x_{border_i} := x$
  $iter := 0$
  while $f(x_{border_i}) == f(x)$ and $iter \leq T$ **do**
    $x_{border_i} := x_{border_i} + \delta\vec{u}_i$
    $iter := iter + 1$
  end while
  if $f(x_{border_i})! = f(x)$ **then**
    $dist_i := \ell_2(x, x_{border_i})$
  end if
end for
$x_{border} := x_{border_i}$ such that $i := \arg\min dist_i$
Return x_{border}

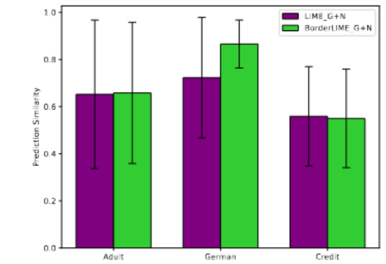
Experiments

- Fidelity of Standard LIME & BorderLIME variants

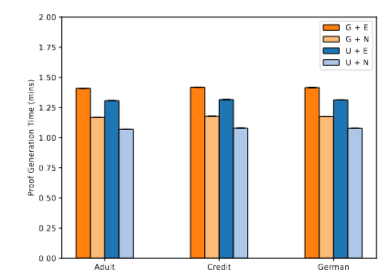


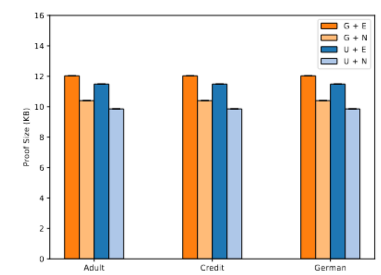


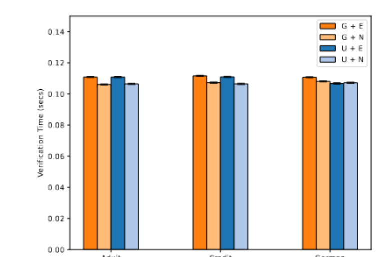
- Standard LIME vs. BorderLIME



- Is ExpProof computationally feasible for LIME?







ZKP Overhead Type	BorderLIME	LIME
Proof Generation Time (mins)	4.85 ± 10^{-2}	1.17 ± 10^{-2}
Verification Time (secs)	0.30 ± 10^{-2}	0.11 ± 10^{-2}
Proof Size (KB)	18.30 ± 0	10.40 ± 0

Table 1. ZKP Overhead of BorderLIME and Standard LIME (both G+N variant) for NNs. Overhead for BorderLIME is larger than that for LIME. Results are consistent across all datasets.