S-LIME: Stabilized-LIME for Model Explanation

Zhengze Zhou
Cornell University
Ithaca, New York, USA
zz433@cornell.edu

Giles Hooker
Cornell University
Ithaca, New York, USA
gjh27@cornell.edu

Fei Wang
Weill Cornell Medicine
New York City, New York, USA
few2001@med.cornell.edu

ABSTRACT

An increasing number of machine learning models have been deployed in domains with high stakes such as finance and healthcare. Despite their superior performances, many models are black boxes in nature which are hard to explain. There are growing efforts for researchers to develop methods to interpret these black-box models. Post hoc explanations based on perturbations, such as LIME [39], are widely used approaches to interpret a machine learning model after it has been built. This class of methods has been shown to exhibit large instability, posing serious challenges to the effectiveness of the method itself and harming user trust. In this paper, we propose S-LIME, which utilizes a hypothesis testing framework based on central limit theorem for determining the number of perturbation points needed to guarantee stability of the resulting explanation. Experiments on both simulated and real world data sets are provided to demonstrate the effectiveness of our method.

CCS CONCEPTS

• **Computing methodologies** → **Feature selection**; *Supervised learning by classification*; • **Mathematics of computing** → **Hypothesis testing and confidence interval computation**.

KEYWORDS

interpretability; stability; LIME; hypothesis testing

ACM Reference Format:
Zhengze Zhou, Giles Hooker, and Fei Wang. 2021. S-LIME: Stabilized-LIME for Model Explanation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3447548.3467274

1 INTRODUCTION

Data Mining and machine learning models have been widely deployed for decision making in many fields, including criminal justice [54] and healthcare [35, 37]. However, many models act as "black boxes" in that they only provide predictions but with little guidance for humans to understand the process. It has been a desiderata to develop approaches for understanding these complex models, which can help increase user trust [39], assess fairness and privacy [4, 11], debug models [28] and even for regulation purposes [19].

Model explanation methods can be roughly divided into two categories [12, 52]: intrinsic explanations and post hoc explanations. Models with intrinsically explainable structures include linear models, decision trees [6], generalized additive models [20], to name a few. Due to complexity constraints, these models are usually not powerful enough for modern tasks involving heterogeneous features and enormous numbers of samples.

Post hoc explanations, on the other hand, provide insights after a model is trained. These explanations can be either model-specific, which are typically limited to specific model classes, such as split improvement for tree-based methods [57] and saliency maps for convolutional networks [42]; or model-agnostic that do not require any knowledge of the internal structure of the model being examined, where the analysis is often conducted by evaluating model predictions on a set of perturbed input data. LIME [39] and SHAP [31] are two of the most popular model-agnostic explanation methods.

Researchers have been aware of some drawbacks for post hoc model explanation. [25] showed that widely used permutation importance can produce diagnostics that are highly misleading due to extrapolation. [17] demonstrated how to generate adversarial perturbations that produce perceptively indistinguishable inputs with the same predicted label, yet have very different interpretations. [1] showed that explanation algorithms can be exploited to systematically rationalize decisions taken by an unfair black-box model. [40] argued against using post hoc explanations as these methods can provide explanations that are not faithful to what the original model computes.

In this paper, we focus on post hoc explanations based on perturbations [39]: one of the most popular paradigm for designing model explanation methods. We argue that the most important property of any explanation technique is *stability* or *reproducibility*: repeated runs of the explanation algorithm under the same conditions should ideally yield the same results. Unstable explanations provide little insight to users as how the model actually works and are considered unreliable. Unfortunately, LIME is not always stable. [55] separated and investigated sources of instability in LIME. [51] highlighted a trade-off between explanation's stability and adherence and propose a framework to maximise stability. [30] improved the sensitivity of LIME by averaging multiple output weights for individual images.

We propose a hypothesis testing framework based on a central limit theorem for determining the number of perturbation samples required to guarantee stability of the resulting explanation. Briefly, LIME works by generating perturbations of a given instance and learning a sparse linear explanation, where the sparsity is usually achieved by selecting top features via LASSO [49]. LASSO is known

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467274

to exhibit early occurrence of false discoveries [33, 47] which, combined with the randomness introduced in the sampling procedure, results in practically-significant levels of instability. We carefully analyze the Least Angle Regression (LARS) [13] for generating the LASSO path and quantify the aymptotics for the statistics involved in selecting the next variable. Based on a hypothesis testing procedure, we design a new algorithm call S-LIME (Stabilized-LIME) which can automatically and adaptively determine the number of perturbations needed to guarantee a stable explanation.

In the following, we review relevant background on LIME and LASSO along with their instability in Section 2. Section 3 statistically analyzes the asymptotic distribution of the statistics which is at the heart of variable selection in LASSO. Our algorithm S-LIME is introduced in Section 4. Section 5 presents empirical studies on both simulated and real world data sets. We conclude in Section 6 with some discussions.

2 BACKGROUND

In this section, we review the general framework for constructing *post hoc* explanations based on perturbations using Local Interpretable Model-agnostic Explanations (LIME) [39]. We then briefly discuss LARS and LASSO, which are the internal solvers for LIME to achieve the purpose of feature selection. We illustrate LIME's instability with toy experiments.

2.1 LIME

Given a black box model f and a target point $\boldsymbol{x}$ of interest, we would like to understand the behavior of the model locally around $\boldsymbol{x}$. No knowledge of f's internal structure is available but we are able to query f many times. LIME first samples around the neighborhood of $\boldsymbol{x}$, query the black box model f to get its predictions and form a pseudo data sets $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$ with $y_i = f(\boldsymbol{x}_i)$ and a hyperparameter n specifying the number of perturbations. The model f can be quite general as regression ($y_i \in \mathcal{R}$) or classification ($y_i \in \{0, 1\}$ or $y_i \in [0, 1]$ if f returns a probability). A model g from some interpretable function spaces G is chosen by solving the following optimization

$$\arg\min_{g \in G} L(f, g, \pi_{\boldsymbol{x}}) + \Omega(g) \tag{1}$$

where

- $\pi_{\boldsymbol{x}}(\boldsymbol{z})$ is a proximity measure between a perturbed instance $\boldsymbol{z}$ to $\boldsymbol{x}$, which is usually chosen to be a Gaussian kernel.
- $\Omega(g)$ measures complexity of the explanation $g \in G$. For example, for decision trees $\Omega(g)$ can be the depth of the tree, while for linear models we can use the number of non-zero weights.
- $L(f, g, \pi_{\boldsymbol{x}})$ is a measure of how unfaithful g is in approximating f in the locality defined by $\pi_{\boldsymbol{x}}$.

[39] suggests a procedure called k-LASSO for selecting top k features using LASSO. In this case, G is the class of linear models with $g = \boldsymbol{\omega}_g \cdot \boldsymbol{x}$, $L(f, g, \pi_{\boldsymbol{x}}) = \sum_{i=1}^{n} \pi_{\boldsymbol{x}}(\boldsymbol{x}_i)(y_i - g(\boldsymbol{x}_i))^2$ and $\Omega = \infty \mathbb{1}[||\omega_g||_0 > k]$. Under this setting, (1) can be approximately solved by first selecting K features with LASSO (using the regularization path) and then learning the weights via least square [39].

We point out here the resemblance between post hoc explanations and knowledge distillation [7, 22]; both involve obtaining predictions from the original model, usually on synthetic examples, and using these to train a new model. Differences lie in both the scope and intention in the procedure. Whereas LIME produces interpretable models that apply closely to the point of interest, model distillation is generally used to provide a global compression of the model representation in order to improve both computational and predictive performance [18, 34]. Nonetheless, we might expect that distillation methods to also exhibit the instability described here; see [56] which documents instability of decision trees used to provide global interpretation.

2.2 LASSO and LARS

Even models that are "interpretable by design" can be difficult to understand, such as a deep decision tree containing hundreds of leaves, or a linear model that employs many features with non-zero weights. For this reason LASSO [49], which automatically produces sparse models, is often the default solver for LIME.

Formally, suppose $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$ with $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ for $1 \le i \le n$, LASSO solves the following optimization problem:

$$\hat{\beta}^{LASSO} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{2}$$

where λ is the multiplier for l_1 penalty. (2) can be efficiently solved via a slight modification of the LARS algorithm [13], which gives the entire LASSO path as λ varies. This procedure is described in Algorithm 1 and 2 below [14], where we denote $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ and assume $n > p$.

Algorithm 1: Least Angle Regression (LARS)

(1) Standardize the predictors to have zero mean and unit norm. Start with residual $\boldsymbol{r} = \boldsymbol{y} - \bar{\boldsymbol{y}}, \beta_1, \beta_2, \ldots, \beta_p = 0$.
(2) Find the predictor $\boldsymbol{x}_{\cdot j}$ most correlated with $\boldsymbol{r}$, and move β_j from 0 towards its least-squares coefficient $\langle \boldsymbol{x}_{\cdot j}, \boldsymbol{r} \rangle$, until some other competitors $\boldsymbol{x}_{\cdot k}$ has as much correlation with the current residual as does $\boldsymbol{x}_{\cdot j}$.
(3) Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\boldsymbol{x}_{\cdot j}, \boldsymbol{x}_{\cdot k})$, until some other competitors $\boldsymbol{x}_{\cdot l}$ has as much correlation with the current residual.
(4) Repeat step 2 and 3 until all p predictors have been entered, at which point we arrive at the full least squares solution.

Algorithm 2: LASSO: Modification of LARS

3a. In step 3 of Algorithm 1, if a non-zero coefficient hits zero, drop the corresponding variable from the active set of variables and recompute the current joint least squares direction.

Both Algorithm 1 and 2 can be easily modified to incorporate a weight vector $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_n)$ on the data set $\mathcal{D}$, by transforming it to $\mathcal{D} = \{(\sqrt{\omega_1}\boldsymbol{x}_1, \sqrt{\omega_1}y_1), (\sqrt{\omega_2}\boldsymbol{x}_2, \sqrt{\omega_2}y_2), \ldots, (\sqrt{\omega_n}\boldsymbol{x}_n, \sqrt{\omega_n}y_n)\}$.

2.3 Instability with LIME

Both [55] and [53] have demonstrated that the random generation of perturbations results in instability in the generated explanations. We apply LIME on Breast Cancer Data (see Section 5.1 for details) to illustrate of this phenomenon. A random forests [5] with 500 trees is built as the black box model, and we apply LIME to explain the prediction of a randomly selected test point multiple times. Each time 1000 synthetic data are generated around the test point and top 5 features are selected via LASSO. We repeat the experiment 100 times and calculate the empirical selection probability of features. The result is shown in Figure 1.

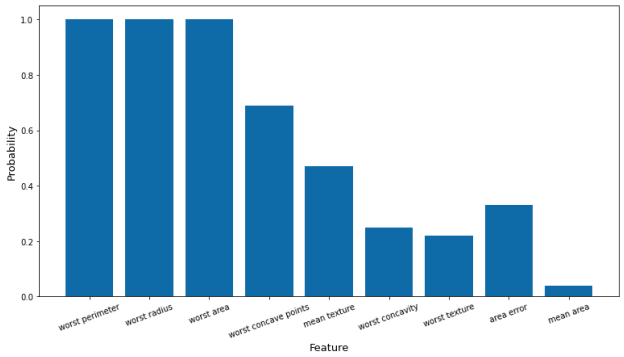


Figure 1: Empirical selection probability for features in Breast Cancer Data. The black box model is a random forests classifier with 500 trees. LIME is run 100 times on a randomly selected test point and top 5 features are selected via LASSO.

We can see that across 100 repetitions, only three features are consistently selected by LIME while there is considerable variability in the remaining features. Note that this does not consider the order of the features entered: even the top three features exhibit different orderings in the selection process.

This experiment illustrates an important weakness of LIME: its instability or irreproducibility. If repeated runs using the *same* explanation algorithm on the *same* model to interpret the *same* data point yield different results, the utility of the explanation is brought into question. The instability comes from the randomness introduced when generating synthetic samples around the input, and the l_1 penalty employed in LASSO further increases the chance of selecting spurious features [48]. In Appendix A we show the instability with LASSO using a simple linear model.

One way to stabilize the LIME model is to use a larger corpus of the synthetic data, but it is difficult to determine how much larger *as a priori* without repeated experiments. In the next section, we examine how feature selection works in LASSO and LARS, and then design a statistically justified approach to automatically and adaptively determine the number of perturbations required to guarantee stability.

3 ASYMPTOTIC PROPERTIES OF LARS DECISIONS

Consider at any given step when LARS needs to choose a new variable to enter the model. With sample size of n, let the current residuals be given by $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)$, and two candidate variables being $\boldsymbol{x_{\cdot i}} = (x_{1i}, x_{2i}, \ldots, x_{ni})$ and $\boldsymbol{x_{\cdot j}} = (x_{1j}, x_{2j}, \ldots, x_{nj})$ where we assume the predictors have been standardized to have zero mean and unit norm. LARS chooses the predictor that has the highest (absolute) correlation with the residuals to enter the model. Equivalently, one needs to compare $\hat{c}_1 = \frac{1}{n}\sum_{t=1}^{n} r_t x_{ti}$ with $\hat{c}_2 = \frac{1}{n}\sum_{t=1}^{n} r_t x_{tj}$. We use $\hat{c}_1$ and $\hat{c}_2$ to emphasize these are finite sample estimates, and our purpose is to obtain the probability that their order would be different if the query points were regenerated. To that end, we introduce uppercase symbols $\boldsymbol{R}, \boldsymbol{X_{\cdot i}}, \boldsymbol{X_{\cdot j}}$ to denote the corresponding random variables of the residuals and two covariates; these are distributed according to the current value of the coefficients in the LASSO path and we seek to generate enough data to return the same ordering as the expected values $c_1 = E(\boldsymbol{R} \cdot \boldsymbol{X_{\cdot i}})$ and $c_2 = E(\boldsymbol{R} \cdot \boldsymbol{X_{\cdot j}})$ with high probability. Our algorithm is based on pairwise comparisons between candidate features; we therefore consider the decision between two covariates in this section, and extensions to more general cases involving multiple pairwise comparisons will be discussed in Section 4.

By the multivariate Central Limit Theorem (CLT), we have

$$\sqrt{n}\left(\begin{bmatrix}\hat{c}_1\\ \hat{c}_2\end{bmatrix} - \begin{bmatrix}c_1\\ c_2\end{bmatrix}\right) \longrightarrow N(0,\ \Sigma),$$

where

$$\Sigma = \operatorname{cov}\begin{bmatrix}\boldsymbol{R} \cdot \boldsymbol{X_{\cdot i}}\\ \boldsymbol{R} \cdot \boldsymbol{X_{\cdot j}}\end{bmatrix} = \begin{bmatrix}\sigma_{11}^2 & \sigma_{12}^2\\ \sigma_{21}^2 & \sigma_{22}^2\end{bmatrix}.$$

Without loss of generality we assume $\hat{c}_1 > \hat{c}_2 > 0$. In general if the correlation is negative, we can simply negate the corresponding feature values for all the calculations involved in this section. Let $\hat{\Delta}_n = \hat{c}_1 - \hat{c}_2$ and $\Delta_n = c_1 - c_2$. Consider function $f(a_1, a_2) = a_1 - a_2$. Delta method implies that

$$\sqrt{n}\left(f\left(\begin{bmatrix}\hat{c}_1\\ \hat{c}_2\end{bmatrix}\right) - \left(f\begin{bmatrix}c_1\\ c_2\end{bmatrix}\right)\right) \longrightarrow N(0,\ \sigma_{11}^2 + \sigma_{22}^2 - \sigma_{12}^2 - \sigma_{21}^2).$$

Or approximately,

$$\hat{\Delta}_n - \Delta_n \sim N\Big(0,\ \frac{\hat{\sigma}_{11}^2 + \hat{\sigma}_{22}^2 - \hat{\sigma}_{12}^2 - \hat{\sigma}_{21}^2}{n}\Big) \tag{3}$$

where the variance estimates are estimated from the empirical covariance of the values $r_t x_{ti}$ and $r_t x_{tj}$, $t = 1, \ldots, n$.

In similar spirits of [56], we assess the probability that $\hat{\Delta}_n > 0$ will still hold in a repeated experiment. Assume we have another independently generated data set denoted by $\{r_t^*, x_{ti}^*, x_{tj}^*\}_{t=1}^n$. It follows from (3) that

$$\hat{\Delta}_n^* - \hat{\Delta}_n \sim N\Big(0,\ 2 \cdot \frac{\hat{\sigma}_{11}^2 + \hat{\sigma}_{22}^2 - \hat{\sigma}_{12}^2 - \hat{\sigma}_{21}^2}{n}\Big),$$

which leads to the approximation that

$$\hat{\Delta}_n^* \Big| \left(\hat{\Delta}_n = \hat{c}_1 - \hat{c}_2\right) \sim N\Big(\hat{c}_1 - \hat{c}_2,\ 2 \cdot \frac{\hat{\sigma}_{11}^2 + \hat{\sigma}_{22}^2 - \hat{\sigma}_{12}^2 - \hat{\sigma}_{21}^2}{n}\Big).$$

In order to control $P(\hat{\Delta}_n^* > 0)$ at a confidence level $1-\alpha$, we need

$$\hat{c}_1 - \hat{c}_2 > Z_\alpha \sqrt{2\frac{\hat{\sigma}_{11}^2 + \hat{\sigma}_{22}^2 - \hat{\sigma}_{12}^2 - \hat{\sigma}_{21}^2}{n}}, \tag{4}$$

where Z_α is the $(1-\alpha)$-quantile of a standard normal distribution.

For a fixed confidence level α and n, suppose we get the corresponding p-value $p_n > \alpha$. From (4) we have

$$\sqrt{n}\frac{\hat{c}_1 - \hat{c}_2}{\sqrt{2(\hat{\sigma}_{11}^2 + \hat{\sigma}_{22}^2 - \hat{\sigma}_{12}^2 - \hat{\sigma}_{21}^2)}} = Z_{p_n}.$$

This implied we would need approximately n' samples to get a significant result where

$$\sqrt{\frac{n}{n'}} = \frac{Z_{p_n}}{Z_\alpha}. \tag{5}$$

4 STABILIZED-LIME

Based on the theoretical analysis developed in Section 3, we can run LIME equipped with hypothesis testing at each step when a new variable enters. If the testing result is significant, we continue to the next step; otherwise it indicates that the current sample size of perturbations is not large enough. We thus generate more synthetic data according to Equation (5) and restart the whole process. Note that we view any intermediate step as *conditioned on* previous obtained estimates of $\hat{\beta}$. A high level sketch of the algorithm is presented below in Algorithm 3.

In practice, we may need to set an upper bound on the number of synthetic samples generated (denoted by n_{max}), such that whenever the new n' is greater than n_{max}, we'll simply set $n = n_{max}$ and go though the outer while loop one last time *without* testing at each step. This can prevent the algorithm from running too long and wasting computation resources in cases where two competing features are equally important in a local neighborhood; for example, if the black box model is indeed locally linear with equal coefficients for two predictors.

We note several other possible variations of the Algorithm 3.

Multiple testing. So far we've only considered comparing a pair of competing features (the top two). But when choosing the next predictor to enter the model at step m (with $m-1$ active features), there are $p-m+1$ candidate features. We can modify the procedure to select the best feature among all the remaining candidates, by conducting pairwise comparisons between the feature with largest correlation ($\hat{c}_1$) against the rest ($\hat{c}_2, \ldots, \hat{c}_{p-m+1}$). This is a multiple comparisons problem, and one can use an idea analogous to Bonferroni correction. Mathematically:

- Test the hypothesis $H_{i,0} : \hat{c}_1 \le \hat{c}_i$, $i = 2, \ldots, p-m+1$. Obtain p-values $p_2, \ldots, p_{p-m+1}$.
- Reject the null hypothesis if $\sum_{i=2}^{p-m+1} p_i < \alpha$.

Although straightforward, this Bonferroni-like correction ignores much of the correlation among these statistics and will result in a conservative estimate. In the experiments, we only conduct hypothesis testing for top two features without resorting to multiple testing, as it is more efficient and empirically we do not observe any performance degradation.

Efficiency. Several modifications can be made to improve the efficiency of Algorithm 3. At each step when n is increased to n', we can reuse the existing synthetic samples and only generate additional $n'-n$ perturbation points. One may also note that whenever the outer while loop restarts, we conduct repetitive testings for the first several variables entering the model. To achieve better efficiency, each new run can *condition on* previous runs: if a variable enters the LASSO path in the same order as before and has been tested with significant statistics, no additional testing is needed. Hypothesis testing is only invoked when we select more features than previous runs, or in some rare cases, the current iteration disagrees with previous results. In our experiments, we do not implement the conditioning step for implementation simplicity, as we find the efficiency gain is marginal when selecting a moderate size of features.

Algorithm 3: S-LIME

Input : A black box model f, data sample to explain $\boldsymbol{x}$, initial size for perturbation samples n_0, significance level α, number of features to select k, proximity measure $\pi_{\boldsymbol{x}}$.
Output : Top k features selected for interpretation.

```
Generate D = {n_0 synthetic samples around x} and
  calculate weight vector ω using π_x;
Set n = n_0;
while True do
    Run Algorithm 2 on D with weight ω along with
      hypothesis testing at each step:
    while active features less than k do
        Select top two predictors most correlated with the
          current residual from remaining covariates, with
          covariance ĉ_1 and ĉ_2;
        Calculate test statistic:
            t = ĉ_1 − ĉ_2 − Z_α sqrt(2 (σ̂_11^2 + σ̂_22^2 − σ̂_12^2 − σ̂_21^2) / n)
        if t >= 0 then
            Continue with this selection;
        else
            Calculate n' = n * (Z_α / Z_{p_n})^2 and set n = n';
            Break;
        end
    end
    if active features less than k then
        Generate D = {n' synthetic samples around x} and
          calculate weight vector ω using π_x;
    else
        Return k selected features;
    end
end
```

5 EMPIRICAL STUDIES

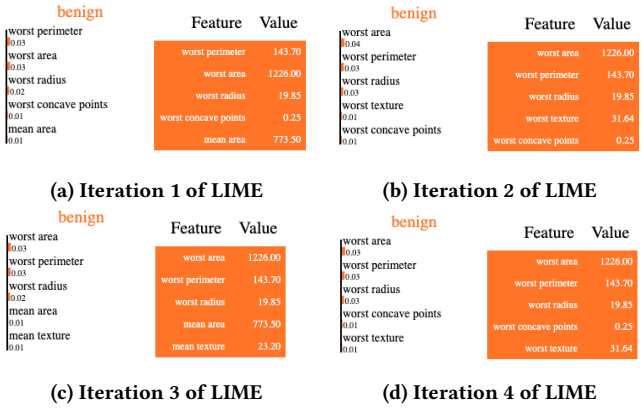
Rather than performing a broad-scale analysis, we look at several specific cases as illustrations to show the effectiveness of S-LIME in generating stabilized model explanations. Scikit-learn [36] is used

for building black box models. Code for replicating our experiments is available at https://github.com/ZhengzeZhou/slime.

5.1 Breast Cancer Data

We use the widely adopted Breast Cancer Wisconsin (Diagnostic) Data Set [32], which contains 569 samples and 30 features[1]. A random forests with 500 trees is trained on 80% of the data as the black box model to predict whether an instance is benign or malignant. It achieves around 95% accuracy on the remaining 20% test data. Since our focus is on producing stabilized explanations for a specific instance, we do not spend additional efforts in hyperparameter tuning to further improve model performance.

Figure 1 in Section 2.3 has already demonstrated the inconsistency of the selected feature returned by original LIME. In Figure 2 below, we show a graphical illustration of four LIME replications on a randomly selected test instance, where the left column of each sub figure shows selected features along with learned linear parameters, and the right column is the corresponding feature value for the sample. These repetitions of LIME applied on the same instance have different orderings for the top two features, and also disagree on the fourth and fifth features.



(a) Iteration 1 of LIME (b) Iteration 2 of LIME

(c) Iteration 3 of LIME (d) Iteration 4 of LIME

Figure 2: Four iterations of LIME on Breast Cancer Data. The black box model is a random forests classifier with 500 trees. LIME explanations are generated with 1000 synthetic perturbations.

To quantify the stability of the generated explanations, we measure the Jaccard index, which is a statistic used for gauging the similarity and diversity of sample sets. Given two sets A and B (in our case, the sets are selected features from LIME), the Jaccard coefficient is defined as the size of the intersection divided by the size of the union:

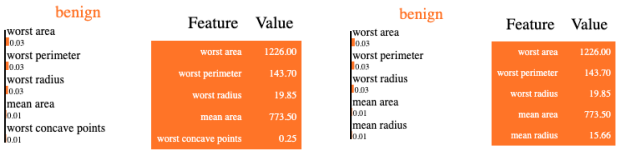
$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

One disadvantage of the Jaccard index is that it ignores ordering within each feature set. For example, if top two features returned from two iterations of LIME are $A = \{worst\ perimeter,\ worst\ area\}$ and $B = \{worst\ area,\ worst\ perimeter\}$, we have $J(A,B) = 1$ but it does not imply LIME explanations are stable. To better quantify stability, we look at the Jaccard index for the top k features for $k = 1, \ldots, 5$. Table 1 shows the average Jaccard across all pairs for 20 repetitions of both LIME and S-LIME on the selected test instance. We set $n_{max} = 10000$ for S-LIME.

Table 1: Average Jaccard index for 20 repetitions for LIME and S-LIME. The black box model is a random forests with 500 trees.

Position	LIME	S-LIME
1	0.61	1.0
2	1.0	1.0
3	1.0	1.0
4	0.66	1.0
5	0.59	0.85

As we can see, for top four positions the average Jaccard index of S-LIME is 1, meaning the algorithm is stable across different iterations. There is some variability in the fifth feature selected, as two features *mean radius* and *worst concave points* have pretty close impact locally. Further increasing n_{max} will make the selection of fifth variable more consistent. Figure 3 shows the only two explanations we observed in simulations for S-LIME, where the difference is at the fifth variable.



(a) Iteration 1 of S-LIME (b) Iteration 2 of S-LIME

Figure 3: Two iterations of S-LIME on Breast Cancer Data. The black box model is a random forests classifier with 500 trees.

As a contrast, we've already seen instability for LIME even for the first variable selected. Although LIME consistently selects the same top two and the third feature, there is much variably for the fourth and fifth feature. This experiment demonstrates the stability of S-LIME compared to LIME. In Appendix B.1, we apply S-LIME on other types of black box models. Stability results on a large cohort of test samples are included in Appendix B.2.

5.2 MARS Test Function

Here we use a modification of the function given in [15] (to test the MARS algorithm) as the black box model so we know the underlying true local weights of variables. Let $y = f(\boldsymbol{x}) = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.05)^2 + 5.2x_4 + 5x_5$, where $X \sim U([0,1]^5)$. The test point $\boldsymbol{x}$ is chosen to be $(0.51, 0.49, 0.5, 0.5, 0.5)$. We can easily calculate the local linear weights of the five variables around $\boldsymbol{x}$ and the expected selection order is $(x_3, x_1, x_2, x_4, x_5)$. Note here the specific choice of parameters in $f(x)$ and the location of test point $\boldsymbol{x}$ makes it difficult to distinguish between x_1, x_2 and x_4, x_5.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html

Table 2 presents the average Jaccard index for the selected feature sets by LIME and S-LIME, where LIME is generated with 1000 synthetic samples and we set $n_0 = 1000$ and $n_{max} = 10000$ for S-LIME. The close local weights between x_1, x_2 and x_4, x_5 causes some instability in LIME, as can be seen from the drop in the index at position 2 and 4. S-LIME outputs consistent explanations in this case.

Table 2: Average Jaccard index for 20 repetitions for LIME and S-LIME on test point $(0.51, 0.49, 0.5, 0.5, 0.5)$**. The black box model is MARS.**

Position	LIME	S-LIME
1	1.0	1.0
2	0.82	1.0
3	1.0	1.0
4	0.79	1.0
5	1.0	1.0

5.3 Early Prediction of Sepsis From Electronic Health Records

Sepsis is a major public health concern which is a leading cause of death in the United States [3]. Early detection and treatment of a sepsis incidence is a crucial factor for patient outcomes [38]. Electronic health records (EHR) store data associated with each individual's health journey and have seen an increasing use recently in clinical informatics and epidemiology [46, 50]. There have been several work to predict sepsis based on EHR [16, 21, 29]. Interpretability of these models are essential for them to be deployed in clinical settings.

We collect data from MIMIC-III [26], which is a freely accessible critical care database. After pre-processing, there are 15309 patients in the cohort for analysis, out of which 1221 developed sepsis based on Sepsis-3 clinical criteria for sepsis onset [43]. For each patient, the record consists of a combination of hourly vital sign summaries, laboratory values, and static patient descriptions. We provide the list of all variables involved in Appendix C. ICULOS is a timestamp which denotes the hours since ICU admission for each patient, and thus is not used directly for training the model.

For each patient's records, missing values are filled with the most recent value if available, otherwise a global average. Negative samples are down sampled to achieve a class ratio of 1:1. We randomly select 90% of the data for training and leave the remaining 10% for testing. A simple recurrent neural network based on LSTM [23] module is built with Keras [9] for demonstration. Each sample fed into the network has 25 features with 24 timestamps, then goes through a LSTM with 32 internal units with dropout rate 0.2, and finally a dense layer with softmax activation to output a probability. The network is optimized by Adam [27] with an initial learning rate of 0.0001 and we train it for 500 epochs on a batch size of 50.

The model achieves around 0.75 AUC score on the test set. Note that we do not fine tune the architecture of the network through cross validation. The purpose of this study is not on achieving a superior performance as it usually requires more advanced modeling techniques for temporal data [16, 29] or exploiting missing value patterns [8]. Instead, we would like to demonstrate the effectiveness of our proposed method in reliably explaining a relatively large scale machine learning model applied to medical data.

To deal with temporal data where each sample in the training set is of shape $(n\_timesteps, n\_features)$, LIME reshapes the data such that it becomes a long vector of size $n\_timesteps \times n\_features$. Essentially it transforms the temporal data to the regular tabular shape while increasing the number of features by a multiple of available timestamps. Table 3 presents the average Jaccard index for the selected feature sets by LIME and S-LIME on two randomly selected test samples, where LIME is generated with 1000 synthetic samples and we set $n_0 = 1000$ and $n_{max} = 100000$ for S-LIME.

LIME exhibits undesirable instability in this example, potentially due to the complex black box model applied and the large number of features ($24 \times 25 = 600$). S-LIME achieves much better stability compared to LIME, although we can still observe some uncertainty in choosing the fifth feature in the second test sample.

Table 3: Average Jaccard index for 20 repetitions for LIME and S-LIME on two randomly selected test samples. The black box model is a recurrent neural network.

Position	LIME	S-LIME
1	0.37	1.0
2	0.29	1.0
3	0.33	1.0
4	0.25	0.89
5	0.26	1.0

(a) test sample 1

Position	LIME	S-LIME
1	0.31	1.0
2	0.24	1.0
3	0.19	1.0
4	0.17	0.96
5	0.18	0.78

(b) test sample 2

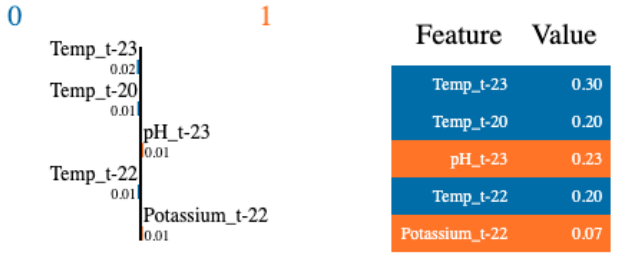
Figure 4 below shows the output of S-LIME on two different test samples. We can see that for sample 1, most recent temperatures play an important role, along with the latest pH and potassium values. While for sample 2, latest pH values are the most important ones.

We want to emphasize that extra caution must be taken by practitioners in applying LIME, especially for some complex problems. The local linear model with a few features might not be suitable to approximate a recurrent neural network built on temporal data. How to apply perturbation based explanation algorithms to temporal data is still an open problem, and we leave it for future work. That being said, the experiment in this section demonstrates the effectiveness of S-LIME in producing stabilized explanations.

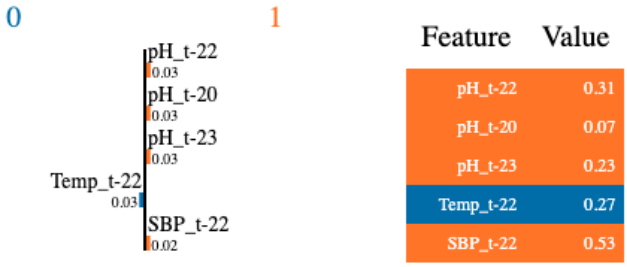
6 DISCUSSIONS

An important property for model explanation methods is stability: repeated runs of the algorithm on the same object should output consistent results. In this paper, we show that post hoc explanations based on perturbations, such as LIME, are not stable due to the randomness introduced in generating synthetic samples. Our proposed algorithm S-LIME is based on a hypothesis testing framework and can automatically and adaptively determine the appropriate number of perturbations required to guarantee stability.

The idea behind S-LIME is similar to [56] which tackles the problem of building stable approximation trees in model distillation. In the area of online learning, [10] uses Hoeffding bounds [24] to



(a) test sample 1



(b) test sample 2

Figure 4: Output of S-LIME for two randomly selected test samples. The black box model is a recurrent neural network.

guarantee correct choice of splits in a decision tree by comparing two best attributes. We should mention that S-LIME is not restricted to LASSO as its feature selection mechanism. In fact, to produce a ranking of explanatory variables, one can use any sequential procedures which build a model by sequentially adding or removing variables based upon some criterion, such as forward-stepwise or backward-stepwise selection [14]. All of these methods can be stabilized by a similar hypothesis testing framework like S-LIME.

There are several works closely related to ours. [55] identifies three sources of uncertainty in LIME: sampling variance, sensitivity to choice of parameters and variability in the black box model. We aim to control the first source of variability as the other two depend on specific design choices of the practitioners. [51] highlight a trade-off between explanation's stability and adherence. Their approach is to select a suitable kernel width for the proximity measure, but it does not improve stability *given* any kernel width. In [53], the authors design a deterministic version of LIME by only looking at existing training data through hierarchical clustering without resorting to synthetic samples. However, the number of samples in a dataset will affect the quality of clusters and a lack of nearby points poses additional challenges; this strategy also relies of having access to the training data. Most recently, [45] develop a set of tools for analyzing explanation uncertainty in a Bayesian framework for LIME. Our method can be viewed as a frequentist counterpart without the need to choose priors and evaluate a posterior distribution.

Another line of work concerns adversarial attacks to LIME. [44] propose a scaffolding technique to hide the biases of any given classifier by building adversarial classifiers to detect perturbed instances. Later, [41] utilize a generative adversarial network to sample more realistic synthetic data for making LIME more robust to adversarial attacks. The technique we developed in this work is *orthogonal* to these directions, as . We also plan to explore other data generating procedures which can help with stability.

ACKNOWLEDGMENTS

Giles Hooker is supported by NSF DMS-1712554. Fei Wang is supported by NSF 1750326, 2027970, ONR N00014-18-1-2585, Amazon Web Service (AWS) Machine Learning for Research Award and Google Faculty Research Award.

REFERENCES

[1] Ulrich Aïvodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. 2019. Fairwashing: the risk of rationalization. *arXiv preprint arXiv:1901.09749* (2019).
[2] Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards Understanding Ensemble, Knowledge Distillation and Self-Distillation in Deep Learning. *arXiv preprint arXiv:2012.09816* (2020).
[3] Derek C Angus, Walter T Linde-Zwirble, Jeffrey Lidicker, Gilles Clermont, Joseph Carcillo, and Michael R Pinsky. 2001. Epidemiology of severe sepsis in the United States: analysis of incidence, outcome, and associated costs of care. *Read Online: Critical Care Medicine| Society of Critical Care Medicine* 29, 7 (2001), 1303–1310.
[4] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias. ProPublica. *See https://www. propublica. org/article/machine-bias-risk-assessments-in-criminal-sentencing* (2016).
[5] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[6] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. 1984. *Classification and regression trees.* CRC press.
[7] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 535–541.
[8] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8, 1 (2018), 1–12.
[9] François Chollet et al. 2015. Keras. https://keras.io.
[10] Pedro Domingos and Geoff Hulten. 2000. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 71–80.
[11] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).
[12] Mengnan Du, Ninghao Liu, and Xia Hu. 2019. Techniques for interpretable machine learning. *Commun. ACM* 63, 1 (2019), 68–77.
[13] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. 2004. Least angle regression. *The Annals of statistics* 32, 2 (2004), 407–499.
[14] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York.
[15] Jerome H Friedman. 1991. Multivariate adaptive regression splines. *The annals of statistics* (1991), 1–67.
[16] Joseph Futoma, Sanjay Hariharan, Katherine Heller, Mark Sendak, Nathan Brajer, Meredith Clement, Armando Bedoya, and Cara O'Brien. 2017. An improved multi-output gaussian process rnn with real-time validation for early sepsis detection. In *Machine Learning for Healthcare Conference*. PMLR, 243–254.
[17] Amirata Ghorbani, Abubakar Abid, and James Zou. 2019. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3681–3688.
[18] Robert D Gibbons, Giles Hooker, Matthew D Finkelman, David J Weiss, Paul A Pilkonis, Ellen Frank, Tara Moore, and David J Kupfer. 2013. The computerized adaptive diagnostic test for major depressive disorder (CAD-MDD): a screening tool for depression. *The Journal of clinical psychiatry* 74, 7 (2013), 1–478.
[19] Bryce Goodman and Seth Flaxman. 2017. European Union regulations on algorithmic decision-making and a "right to explanation". *AI magazine* 38, 3 (2017), 50–57.
[20] Trevor J Hastie and Robert J Tibshirani. 1990. *Generalized additive models*. Vol. 43. CRC press.
[21] Katharine E Henry, David N Hager, Peter J Pronovost, and Suchi Saria. 2015. A targeted real-time early warning score (TREWScore) for septic shock. *Science translational medicine* 7, 299 (2015), 299ra122–299ra122.
[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).
[23] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[24] Wassily Hoeffding. 1994. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*. Springer, 409–426.

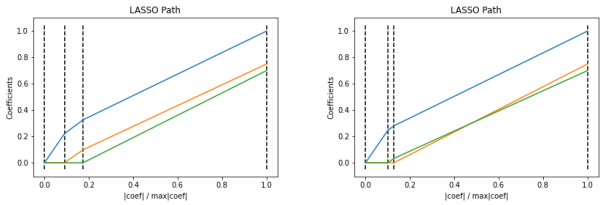
[25] Giles Hooker and Lucas Mentch. 2019. Please stop permuting features: An explanation and alternatives. *arXiv preprint arXiv:1905.03151* (2019).
[26] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3, 1 (2016), 1–9.
[27] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[28] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730* (2017).
[29] Simon Meyer Lauritsen, Mads Ellersgaard Kalør, Emil Lund Kongsgaard, Katrine Meyer Lauritsen, Marianne Johansson Jørgensen, Jeppe Lange, and Bo Thiesson. 2020. Early detection of sepsis utilizing deep learning on electronic health record event sequences. *Artificial Intelligence in Medicine* 104 (2020), 101820.
[30] Eunjin Lee, David Braines, Mitchell Stiffler, Adam Hudler, and Daniel Harborne. 2019. Developing the sensitivity of LIME for better machine learning explanation. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, Vol. 11006. International Society for Optics and Photonics, 1100610.
[31] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*. 4765–4774.
[32] Olvi L Mangasarian, W Nick Street, and William H Wolberg. 1995. Breast cancer diagnosis and prognosis via linear programming. *Operations Research* 43, 4 (1995), 570–577.
[33] Nicolai Meinshausen and Peter Bühlmann. 2010. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72, 4 (2010), 417–473.
[34] Aditya Krishna Menon, Ankit Singh Rawat, Sashank J Reddi, Seungyeon Kim, and Sanjiv Kumar. 2020. Why distillation helps: a statistical perspective. *arXiv preprint arXiv:2005.10419* (2020).
[35] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. 2018. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics* 19, 6 (2018), 1236–1246.
[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[37] Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. 2018. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine* 1, 1 (2018), 18.
[38] Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. 2019. Early prediction of sepsis from clinical data: the PhysioNet/Computing in Cardiology Challenge 2019. In *2019 Computing in Cardiology (CinC)*. IEEE, Page–1.
[39] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
[40] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.
[41] Sean Saito, Eugene Chua, Nicholas Capel, and Rocco Hu. 2020. Improving LIME Robustness with Smarter Locality Sampling. *arXiv preprint arXiv:2006.12302* (2020).
[42] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).
[43] Mervyn Singer, Clifford S Deutschman, Christopher Warren Seymour, Manu Shankar-Hari, Djillali Annane, Michael Bauer, Rinaldo Bellomo, Gordon R Bernard, Jean-Daniel Chiche, Craig M Coopersmith, et al. 2016. The third international consensus definitions for sepsis and septic shock (Sepsis-3). *Jama* 315, 8 (2016), 801–810.
[44] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2020. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 180–186.
[45] Dylan Slack, Sophie Hilgard, Sameer Singh, and Himabindu Lakkaraju. 2020. How Much Should I Trust You? Modeling Uncertainty of Black Box Explanations. *arXiv preprint arXiv:2008.05030* (2020).
[46] Jose Roberto Ayala Solares, Francesca Elisa Diletta Raimondi, Yajie Zhu, Fatemeh Rahimian, Dexter Canoy, Jenny Tran, Ana Catarina Pinho Gomes, Amir H Payberah, Mariagrazia Zottoli, Milad Nazarzadeh, et al. 2020. Deep learning for electronic health records: A comparative review of multiple deep neural architectures. *Journal of biomedical informatics* 101 (2020), 103337.
[47] Weijie Su, Małgorzata Bogdan, Emmanuel Candes, et al. 2017. False discoveries occur early on the lasso path. *The Annals of statistics* 45, 5 (2017), 2133–2150.
[48] Weijie J Su. 2018. When is the first spurious variable selected by sequential regression procedures? *Biometrika* 105, 3 (2018), 517–527.
[49] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58, 1 (1996), 267–288.
[50] Akhil Vaid, Suraj K Jaladanki, Jie Xu, Shelly Teng, Arvind Kumar, Samuel Lee, Sulaiman Somani, Ishan Paranjpe, Jessica K De Freitas, Tingyi Wanyan, et al. 2020. Federated learning of electronic health records improves mortality prediction in patients hospitalized with covid-19. *medRxiv* (2020).
[51] Giorgio Visani, Enrico Bagli, and Federico Chesani. 2020. OptiLIME: Optimized LIME Explanations for Diagnostic Computer Algorithms. *arXiv preprint arXiv:2006.05714* (2020).
[52] Fei Wang, Rainu Kaushal, and Dhruv Khullar. 2020. Should Health Care Demand Interpretable Artificial Intelligence or Accept" Black Box" Medicine? *Annals of internal medicine* 172, 1 (2020), 59–60.
[53] Muhammad Rehman Zafar and Naimul Mefraz Khan. 2019. DLIME: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. *arXiv preprint arXiv:1906.10263* (2019).
[54] Jiaming Zeng, Berk Ustun, and Cynthia Rudin. 2015. Interpretable classification models for recidivism prediction. *arXiv preprint arXiv:1503.07810* (2015).
[55] Yujia Zhang, Kuangyan Song, Yiming Sun, Sarah Tan, and Madeleine Udell. 2019. " Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations. *arXiv preprint arXiv:1904.12991* (2019).
[56] Yichen Zhou, Zhengze Zhou, and Giles Hooker. 2018. Approximation trees: Statistical stability in model distillation. *arXiv preprint arXiv:1808.07573* (2018).
[57] Zhengze Zhou and Giles Hooker. 2019. Unbiased measurement of feature importance in tree-based methods. *arXiv preprint arXiv:1903.05179* (2019).

A INSTABILITY WITH LASSO

Instability with LASSO has been studied previously by several researchers. [33] introduce stability selection based on subsampling which provides finite sample control for some error rates of false discoveries. [48] find that sequential regression procedures select the first spurious variable unexpectedly early, even in settings of low correlations between variables and strong true effect sizes. [47] further develop a sharp asymptotic trade-off between false and true positive rates along the LASSO path.

We demonstrate this phenomenon using a simple linear case. Suppose $t = \rho_1 x_1 + \rho_2 x_2 + \rho_3 x_3$, where x_1, x_2 and x_3 are independent and generated from a standard normal distribution $\mathcal{N}(0, 1)$. Note that we do not impose any additional noise in generating the response y. We choose $\rho_1 = 1$, $\rho_2 = 0.75$ and $\rho_3 = 0.7$, such that when one uses LARS to solve LASSO, x_1 always enter the model first, while x_2 and x_3 have closer coefficients and will be more challenging to distinguish.

We focus on the ordering of the three covariates entering the model. The "correct" ordering should be (x_1, x_2, x_3). For multiple runs of LASSO with $n = 1000$, we observe roughly 20% of the results have order (x_1, x_3, x_2) instead. Figure 5 below shows two representative LASSO paths.



(a) Variable ordering in LASSO path: (x_1, x_2, x_3). **(b) Variable ordering in LASSO path: (x_1, x_3, x_2).**

Figure 5: Two cases of variable ordering in LASSO path.

This toy experiment demonstrates the instability of LASSO itself. Even in this ideal noise-free setting where we have an independent design with Gaussian distribution for the variables, 20% of the time LASSO exhibits different paths due to random sampling. Intuitively, the solutions at the beginning of the LASSO path is overwhelmingly biased and the residual vector contains many of the true effects. Thus some less relevant or irrelevant variable could exhibit high correlations with the residual and gets selected early. $n = 1000$ seems to be a reasonable large number of samples to achieve consistency results, but when applying the idea of S-LIME, the hypothesis testing is always inconclusive at the second step when it needs to choose between x_2 and x_3. Increasing n in this case can indeed yield significant testing results and stabilize the LASSO paths.

B ADDITIONAL EXPERIMENTS

B.1 S-LIME on other model types

Besides the randomness introduced in generating synthetic perturbations, the output of model explanation algorithms is also dependent on several other factors, including the black box model itself. There may not be a universal truth to the explanations of a given instance, as it depends on how the underlying model captures the relationship between covariates and responses. Distinct model types, or even the same model structure trained with random initialization, can utilize different correlations between features and responses [2], and thus result in different model explanations.

We apply S-LIME on other model types to illustrate two points:

- Compared to LIME, S-LIME can generate stabilized explanations, though for some model types more synthetic perturbations are required.
- Different model types can have different explanations for the same instance. This does not imply that S-LIME is unstable or not reproducible, but practitioners need to be aware of this dependency on the underlying black box model when apply any model explanation methods.

We use support-vector machines (SVM) and neural networks (NN) as the underlying black box models and apply LIME and S-LIME. Basic setups is similar to Section 5.1. For SVM training, we use default parameters[2] where rbf kernel is applied. The NN is constructed with two hidden layers, each with 12 and 8 hidden units. ReLU activations are used between hidden layers while the last layer use sigmoid functions to output a probability. The network is implemented in Keras [9]. Both models achieve over 90% accuracy on the test set.

Table 4 lists the average Jaccard index across 20 repetitions for each setting on a randomly selected test instance. LIME is generated with 1000 synthetic samples, while for S-LIME we set $n_{max} = 100000$ for SVM and $n_{max} = 10000$ for NN. Compared with LIME, S-LIME achieves better stability at each position.

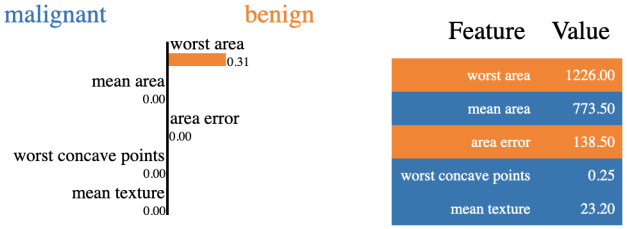
Figure 6 shows the graphical exhibition of the explanations generated by S-LIME for both SVM and NN being the black box models. We can see that they differ in the features selected.

One important observation is that the underlying black box model also affects the stability of local explanations. For example, the original LIME is extremely unstable for SVM. S-LIME needs a larger n_{max} to produce consistent results.

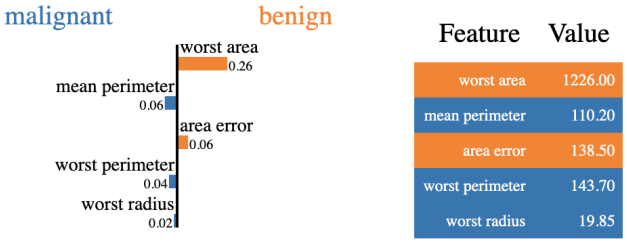
[2]https://scikit-learn.org/stable/modules/svm.html#svm-classification

Table 4: Average Jaccard index for 20 repetitions for LIME and S-LIME. The black box models are SVM and NN.

Position	SVM		NN	
	LIME	S-LIME	LIME	S-LIME
1	1	1.0	0.73	1.0
2	0.35	0.87	0.87	1.0
3	0.23	0.83	0.71	0.74
4	0.19	1.0	0.66	1.0
5	0.18	0.67	0.55	1.0



(a) S-LIME on SVM.



(b) S-LIME on NN.

Figure 6: S-LIME on Breast Cancer Data with SVM and NN as black box models.

B.2 A large cohort of test samples

Most of the experiments in this paper are targeted at a randomly selected test sample, which allows us to examine specific features easily. That being said, one can expect the instability of LIME and the improvement of S-LIME to be universal. In this part we conduct experiments on a large cohort of test samples for both Breast Cancer (Section 5.1) and Sepsis (Section 5.3) data.

In each application, we randomly select 50 test samples. For each test instance, LIME and S-LIME are applied for 20 repetitions and we calculate average Jaccard index across all pairs out of 20 as before. Finally, we report the overall average Jaccard index for 50 test samples. The results are shown in Table 5. LIME explanations are generated with 1000 synthetic samples.

For Breast Cancer Data, we pick $n_{max} = 10000$ as in Section 5.1. We can see that in general there is some instability from the features selected by LIME, while S-LIME can improve stability. By

further increasing n_{max} we may get better stability metrics, but at the cost of computational costs.

For the sepsis prediction task, LIME performs much worse exhibiting undesirable instability across 50 test samples at all 5 positions. S-LIME with $n_{max} = 100000$ achieves obviously stability improvement. The reason for invoking a larger value of n_{max} is due to the fact that there are 600 features to select from. It is an interesting future direction to see how one can use LIME to explain temporal models more efficiently.

Table 5: Overall average Jaccard index for 20 repetitions for LIME and S-LIME across 50 randomly chosen test samples.

Position	LIME	S-LIME
1	0.90	0.98
2	0.85	0.96
3	0.82	0.92
4	0.81	0.96
5	0.80	0.84

(a) Breast Cancer Data

Position	LIME	S-LIME
1	0.54	1.0
2	0.43	1.0
3	0.37	0.78
4	0.35	0.90
5	0.34	0.99

(b) Sepsis Data

C VARIABLES LIST FOR SEPSIS DETECTION

Table 6: Variables list and description for data used in sepsis prediction.

#	Variables	Description
1	Age	age(years)
2	Gender	male (1) or female (0)
3	ICULOS	ICU length of stay (hours since ICU admission)
4	HR	hea1t rate
5	Potassium	potassium
6	Temp	temperature
7	pH	pH
8	PaCO2	partial pressure of carbon dioxide from arterial blood
9	SBP	systolic blood pressure
10	FiO2	fraction of inspired oxygen
11	SaO2	oxygen saturation from arterial blood
12	AST	aspartate transaminase
13	BUN	blood urea nitrogen
14	MAP	mean arterial pressure
15	Calcium	calcium
16	Chloride	chloride
17	Creatinine	creatinine
18	Bilirubin	bilirubin
19	Glucose	glucose
20	Lactate	lactic acid
21	DBP	diastolic blood pressure
22	Troponin	troponin I
23	Resp	respiration rate
24	PTT	partial thromboplastin time
25	WBC	white blood cells count
26	Platelets	platelet count