A Appendix

A.1 Proofs: Existing Methods are Instances of the LFA Framework (Section 3)

A.1.1 LIME

The instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 \odot \xi$ where $\xi\ (\in \{0,1\}^d) \sim \pi_{\mathbf{x}_0}$ with $\pi_{\mathbf{x}_0}$ being the exponential kernel (defined below), and (3) loss function as squared-error loss given by $\ell(f, g, \mathbf{x}_0, \xi) = (f(\mathbf{x}_\xi) - g(\xi))^2$ is equivalent to `LIME`.

As defined in [4] (Section 3.4), the exponential kernel $\pi_{\mathbf{x}_0}(\xi) \propto exp\{-\frac{D(\mathbf{x}_0, \mathbf{x}_\xi)}{\sigma^2}\}$ with distance function D (such as cosine distance or L2 distance) and width σ.

Proof. For this instance of the LFA framework, by definition, the interpretable model g is given by:

$$\begin{aligned} g^* &= \arg\min_{g \in \mathcal{G}} \mathop{\mathbb{E}}_{\xi \sim \pi_{\mathbf{x}_0}} \ell(f, g, \mathbf{x}_0, \xi) \\ &= \arg\min_{g \in \mathcal{G}} \mathop{\mathbb{E}}_{\xi \sim p} [\ell(f, g, \mathbf{x}_0, \xi) \cdot \pi_{\mathbf{x}_0}(\xi)] \text{ where } p \text{ is the Bernouilli(0.5) distribution} \end{aligned}$$

Through importance sampling using a Bernouilli(0.5) proposal distribution (i.e., a Uniform(0,1) distribution over the space of binary inputs), the optimization setting of the LFA framework is that described for `LIME` by Ribeiro et al. [4] (Equations 1 and 2). □

A.1.2 KernelSHAP

The instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 \odot \xi$ where $\xi\ (\in \{0,1\}^d) \sim \pi$ with π being the Shapley kernel (defined below), and (3) loss function as squared-error loss given by $\ell(f, g, \mathbf{x}_0, \xi) = (f(\mathbf{x}_\xi) - g(\xi))^2$ is equivalent to `KernelSHAP`.

As defined in [6] (Theorem 2), the Shapley kernel $\pi(\xi) \propto \frac{M-1}{\binom{M}{k} \cdot k \cdot (M-k)}$ where M is the total number of elements in ξ and k is the number of ones in ξ.

Proof. For this instance of the LFA framework, by definition, the interpretable model g is given by:

$$\begin{aligned} g^* &= \arg\min_{g \in \mathcal{G}} \mathop{\mathbb{E}}_{\xi \sim \pi} \ell(f, g, \mathbf{x}_0, \xi) \\ &= \arg\min_{g \in \mathcal{G}} \mathop{\mathbb{E}}_{\xi \sim p} [\ell(f, g, \mathbf{x}_0, \xi) \cdot \pi(\xi)] \text{ where } p \text{ is the Bernouilli(0.5) distribution} \end{aligned}$$

Through importance sampling using a Bernouilli(0.5) proposal distribution (i.e., a Uniform(0,1) distribution over the space of binary inputs), the optimization setting of the LFA framework is that described for `KernelSHAP` by Lundberg and Lee [6] (Equation 2 and Theorem 2). □

A.1.3 Occlusion

The instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 \odot \xi$ where $\xi\ (\in \{0,1\}^d)$ is a random one-hot vector, and (3) loss function as squared-error loss given by $\ell(f, g, \mathbf{x}_0, \xi) = (\Delta f - g(\xi))^2$ where $\Delta f = f(\mathbf{x}_0) - f(\mathbf{x}_0(1-\xi))$ converges to `Occlusion`.

Proof. This instance of the LFA framework optimizes $g(\xi)$ to approximate Δf. For ξ_i (a one-hot vector with element i equal to 1), $g(\xi_i) = w_i$ and Δf_i is the difference in the model prediction when feature i takes the original value versus when feature i is set to zero. Δf is the definition of explanations generated by `Occlusion`. Thus, in this instance of the LFA framework, the weights of g recover the explanations of `Occlusion`. □

A.1.4 C-LIME

The instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 + \xi$ where $\xi\ (\in \mathbb{R}^d) \sim \text{Normal}(0, \sigma^2)$, and (3) loss function as squared-error loss given by $\ell(f, g, \mathbf{x}_0, \xi) = (f(\mathbf{x}_\xi) - g(\xi))^2$ is equivalent to `C-LIME`.

Proof. This instance of the LFA framework is equivalent to `C-LIME` by definition of `C-LIME`. □

A.1.5 SmoothGrad

In this section, we provide two derivations showing the connection between the LFA framework and `SmoothGrad`. When using gradient-matching loss, the instance of the LFA framework is exactly equivalent to `SmoothGrad` given the same n perturbations. When using squared-error loss, the instance of the LFA framework is equivalent to `SmoothGrad` asymptotically for a large number of perturbations.

Gradient-matching loss function

This instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 + \xi$ where $\xi\ (\in \mathbb{R}^d) \sim \text{Normal}(0, \sigma^2)$, and (3) loss function as gradient-matching loss given by $\ell_{gm}(f, g, \mathbf{x}_0, \xi) = \|\nabla_\xi f(\mathbf{x}_\xi) - \nabla_\xi g(\xi)\|_2^2$ is equivalent to `SmoothGrad`. In other words, for the same n perturbations, this instance of the LFA framework and `SmoothGrad` yield the same explanation.

Proof. For this instance of the LFA framework, by definition, the interpretable model g is given by $g^* = \arg\min_{g \in \mathcal{G}} L$ where:

$$\begin{aligned} L &= \mathbb{E}_\xi \ell(f, g, \mathbf{x}_0, \xi) \\ &= \frac{1}{n} \sum_n \|\nabla_\xi f(\mathbf{x}_\xi) - \nabla_\xi g(\xi)\|_2^2 \\ &= \frac{1}{n} \sum_n \|\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi) - \mathbf{w}\|_2^2 \end{aligned}$$

To derive the solution for $\mathbf{w}$, take the partial derivative of L w.r.t. $\mathbf{w}$, set the partial derivative to zero, and solve for $\mathbf{w}$.

$$\nabla_{\mathbf{w}} L = 0 \Rightarrow (-2)\frac{1}{n} \sum_n [\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi) - \mathbf{w}] = 0 \Rightarrow \mathbf{w} = \frac{1}{n} \sum_n \nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)$$

Therefore, for the same n perturbations, the weights $\mathbf{w}$ of the interpretable model g are equivalent to the `SmoothGrad` explanations. □

Squared-error loss function

Consider the instance of the LFA framework corresponding to `SmoothGrad` described above, except with loss function as squared-error loss given by $\ell(f, g, \mathbf{x}_0, \xi) = (f(\mathbf{x}_\xi) - g(\xi))^2$. This instance of the LFA framework converges to `SmoothGrad` in expectation. Note that this instance of the LFA framework is `C-LIME` and its convergence to `SmoothGrad` in expectation is consistent with the results of [5] which previously proved the same convergence (using a different approach).

Proof. For this instance of the LFA framework, by definition, the interpretable model g is given by $g^* = \arg\min_{g \in \mathcal{G}} L$ where:

$$\begin{aligned} L &= \mathbb{E}_\xi \ell(f, g, \mathbf{x}_0, \xi) \\ &= \mathbb{E}_\xi [(f(\mathbf{x}_\xi) - g(\xi))^2] \\ &= \mathbb{E}_\xi [(f(\mathbf{x}_\xi) - \mathbf{w}^\top \xi)^2] \end{aligned}$$

To derive the solution for $\mathbf{w}$, take the partial derivative of L w.r.t. $\mathbf{w}$, set the partial derivative to zero, and solve for $\mathbf{w}$.

$$
\begin{aligned}
\nabla_{\mathbf{w}} L &= 0 \\
-2\mathbb{E}_\xi[(f(\mathbf{x}_\xi) - \mathbf{w}^\top \xi)\xi^\top] &= 0 \\
\mathbb{E}_\xi[f(\mathbf{x}_\xi)\xi^\top - \mathbf{w}^\top \xi\xi^\top] &= 0 \\
\mathbb{E}_\xi[f(\mathbf{x}_\xi)\xi^\top] - \mathbf{w}^\top \mathbb{E}[\xi\xi^\top] &= 0 \\
\sigma^2 \mathbb{E}_\xi[\nabla_{\mathbf{x}_\xi} f(\mathbf{x}_\xi)^\top] - \sigma^2 \mathbf{w}^\top &= 0 \text{ by Stein's Lemma} \\
\sigma^2 \mathbb{E}_\xi[\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)^\top] - \sigma^2 \mathbf{w}^\top &= 0 \\
\mathbf{w} &= \mathbb{E}_\xi[\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)]
\end{aligned}
$$

Therefore, the weights $\mathbf{w}$ of the interpretable model g converge to `SmoothGrad` explanations in expectation. □

A.1.6 Vanilla gradients

Consider the instance of the LFA framework corresponding to `SmoothGrad` described above (with loss function as either squared-error loss or gradient-matching loss). As $\sigma \to 0$, this instance of the LFA framework converges to `Vanilla Gradients`.

Proof. Starting with the solution for $\mathbf{w}$ derived for `SmoothGrad`, take the limit of $\mathbf{w}$ as $\sigma \to 0^+$.

$$
\begin{aligned}
\lim_{\sigma \to 0^+} \mathbf{w} &= \lim_{\sigma \to 0^+} \mathbb{E}_\xi[\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)] \\
&= \lim_{\sigma \to 0^+} \int_{-\infty}^{\infty} \nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)\, p(\xi; 0, \sigma)\, d\xi \\
&= \lim_{\sigma \to 0^+} \int_{-\infty}^{\infty} \nabla_{\mathbf{x}_0} f(\mathbf{x}_0 + \xi)\, \eta_\xi(\xi)\, d\xi \text{ where } \eta_\xi(\xi) = p(\xi; 0, \sigma) \\
&= \nabla_{\mathbf{x}_0} f(\mathbf{x}_0) \text{ by property of the Dirac delta distribution}
\end{aligned}
$$

To derive the third line from the second line, we view the Normal density function $p(\xi; 0, \sigma)$ as a nascent delta function $\eta_\xi(\xi)$ (which is defined such that $\lim_{\sigma \to 0^+} \int_{-\infty}^{\infty} p(\xi; 0, \sigma) = \delta(\xi)$, where δ is the Dirac delta distribution) and by assuming that $\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)$ has a compact support.

Therefore, the weights $\mathbf{w}$ of the interpretable model g converge to `Vanilla Gradients` explanations. □

A.1.7 Integrated Gradients

This instance of the LFA framework with (1) interpretable model class $\mathcal{G}$ as the class of linear models where $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, (2) perturbations of the form $\mathbf{x}_\xi = \mathbf{x}_0 \odot \xi$ where $\xi\ (\in \mathbb{R}^d) \sim \text{Uniform}(0, 1)$, and (3) loss function as gradient-matching loss given by $\ell_{gm}(f, g, \mathbf{x}_0, \xi) = \|\nabla_\xi f(\mathbf{x}_\xi) - \nabla_\xi g(\xi)\|_2^2$ is equivalent to `Integrated Gradients`. In other words, for the same n perturbations, this instance of the LFA framework and `Integrated Gradients` yield the same explanation.

Proof. For this instance of the LFA framework, by definition, the interpretable model g is given by $g^* = \arg\min_{g \in \mathcal{G}} L$ where:

$$
\begin{aligned}
L &= \mathbb{E}_\xi \ell(f, g, \mathbf{x}_0, \xi) \\
&= \mathbb{E}_\xi \|\nabla_\xi f(\mathbf{x}_\xi) - \nabla_\xi g(\xi)\|_2^2 \\
&= \mathbb{E}_\xi \|\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi) \odot \mathbf{x}_0 - \mathbf{w}\|_2^2
\end{aligned}
$$

Note that, by the chain rule, $\nabla_\xi f(\mathbf{x}_\xi) = \nabla_\xi f(\mathbf{x}_0 \odot \xi) = \nabla_{\mathbf{x}_\xi} f(\mathbf{x}_\xi) \odot \nabla_\xi \mathbf{x}_\xi = \nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi) \odot \mathbf{x}_0$.

To derive the solution for $\mathbf{w}$, take the partial derivative of L w.r.t. $\mathbf{w}$, set the partial derivative to zero, and solve for $\mathbf{w}$.

$$
\begin{aligned}
\nabla_{\mathbf{w}} L &= 0 \\
-2\mathbb{E}_\xi[\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi) \odot \mathbf{x}_0 - \mathbf{w}] &= 0 \\
\mathbf{w} &= \mathbf{x}_0 \odot \mathbb{E}_\xi[\nabla_{\mathbf{x}_0} f(\mathbf{x}_\xi)]
\end{aligned}
$$

Therefore, the weights $\mathbf{w}$ of the interpretable model g converge to `Integrated Gradients` explanations in expectation. $\square$

A.1.8 Gradient $\times$ Input

Consider the instance of the LFA framework corresponding to `Integrated Gradients` described above, except with $\xi\,(\in \mathbb{R}^d) \sim \text{Uniform}(a, 1)$. As $a \to 1$, this instance of the LFA framework converges to `Gradient x Input`.

Proof. As $a \to 1$, $\xi \to \vec{1}$, and $\mathbf{w} \to \mathbf{x}_0 \odot \nabla_{\mathbf{x}_0} f(\mathbf{x}_0)$. Therefore, the weights $\mathbf{w}$ of the interpretable model g converge to `Gradient x Input` explanations. $\square$

A.2 Which Explanations Are Not Function Approximations?

In this section, we briefly discuss explanation methods that cannot be viewed as instances of the LFA framework. In the cases below, the lack of connection to the LFA framework is mainly due to a property of the explanation method.

Model-independent methods. Some explanation methods are known to produce attributions that are independent of the model they intend to explain. These methods cannot be cast in the LFA framework in a meaningful way due to the model recovery conditions we impose. Such model-independent methods include guided backpropagation [24] and DeconvNet [25], following theory by Nie et al. [33], as well as logit-gradient-based methods [34] such as Grad-CAM [26], Grad-CAM++ [27], and FullGrad [28].

Modified-backpropagation methods. Some explanation methods such as DeepLIFT [9], guided backpropagation [24], DeconvNet [25], and layer-wise relevance propagation [35] work by modifying the backpropagation equations and propagating attributions using finite-difference-like methods. Such methods violate an important property called "implementation invariance", first identified by Sundararajan et al. [11], which states that two functionally identical models can have different attributions due to the lack of a chain rule for modified backpropagation methods. This property ensures that such methods cannot be function approximators, as the attribution changes based on the function implementation.

Unsigned-gradient methods. Some gradient-based methods return unsigned attribution values instead of the full signed values. Such methods can be written in the LFA framework using the following loss function $\ell(f, g, \mathbf{x}_0, \xi) = \| |\nabla_\xi f(\mathbf{x}_0 \oplus \xi)| - \mathbf{w}_g \|^2$ where $\mathbf{w}_g$ consists of the weights of the interpretable model g. Using this loss function with different choices for neighborhoods gives unsigned versions of different gradient methods. However, this loss function is not a valid loss function, i.e., $\ell = 0 \not\Longrightarrow f = g$. Using this loss function, $\mathbf{w}_g$ is always positive and thus cannot recover an underlying model's negative weights.

A.3 Proof: No Free Lunch Theorem (Section 4)

Theorem. *Consider explaining a black-box model f around point $\mathbf{x}_0$ using an interpretable model g from model class $\mathcal{G}$ and a valid loss function ℓ where the distance between f and $\mathcal{G}$ is given by $d(f, \mathcal{G}) = \min_{g \in \mathcal{G}} \max_{\mathbf{x} \in \mathcal{X}} \ell(f, g, 0, \mathbf{x})$.*

Then, for any explanation g^* over a neighbourhood distribution $\xi_1 \sim \mathcal{Z}_1$ such that $\max_{\xi_1} \ell(f, g^*, \mathbf{x}_0, \xi_1) \leq \epsilon$, there always exists another neighbourhood $\xi_2 \sim \mathcal{Z}_2$ such that $\max_{\xi_2} \ell(f, g^*, \mathbf{x}_0, \xi_2) \geq d(f, \mathcal{G})$.

Proof. Given an explanation g^*, we can find an "adversarial" input $\mathbf{x}_{adv}$ such that $\mathbf{x}_{adv} = \arg\max_{\mathbf{x} \in \mathcal{X}} \ell(f, g^*, 0, \mathbf{x})$ has a large error ℓ. Construct perturbation $\mathbf{x}_2 = \mathbf{x}_0 + \xi_2$ such that $p(\xi_2) = \text{Uniform}(0, \mathbf{x}_{adv} - \mathbf{x}_0)$, which implies $p(\mathbf{x}_2) = \text{Uniform}(\mathbf{x}_0, \mathbf{x}_{adv})$. In this proof, $\text{Uniform}(a, b)$ denotes uniformly sampling along the straight line connecting a and b.

By definition $\max_{\xi_2} \ell(f, g^*, \mathbf{x}_0, \xi_2) = \ell(f, g^*, \mathbf{x}_0, \mathbf{x}_{adv} - \mathbf{x}_0) = \max_{\mathbf{x} \in \mathcal{X}} \ell(f, g^*, 0, \mathbf{x}) \geq \min_{g \in \mathcal{G}} \max_{\mathbf{x} \in \mathcal{X}} \ell(f, g, 0, \mathbf{x}) = d(f, \mathcal{G})$ $\square$

A salient feature of this proof is that it makes no assumptions about the form of model, input or output domains. This implies that the result applies equally to discrete and continuous domains, regression and classification tasks, and for any model type.

A.4 Summary of Properties of Existing Explanation Methods

Method	**Characteristics of ξ**	**g recovers f?**	**Scale of g's weights when $\mathcal{X} \in \mathbb{R}^d$**
C-LIME	Continuous, Additive	When $\mathcal{X} \in \mathbb{R}^d$	Gradient
SmoothGrad	Continuous, Additive	When $\mathcal{X} \in \mathbb{R}^d$	Gradient
Vanilla Gradients	Continuous, Additive	When $\mathcal{X} \in \mathbb{R}^d$	Gradient
Integrated Gradients	Continuous, Multiplicative	No	Gradient $\times$ Input
Gradients $\times$ Input	Continuous, Multiplicative	No	Gradient $\times$ Input
LIME	Binary, Multiplicative	When $\mathcal{X} \in \{0, 1\}^d$	Gradient $\times$ Input
KernelSHAP	Binary, Multiplicative	When $\mathcal{X} \in \{0, 1\}^d$	Gradient $\times$ Input
Occlusion	Binary, Multiplicative	When $\mathcal{X} \in \{0, 1\}^d$	Gradient $\times$ Input

Table 2: Summary of properties of existing explanation methods in relation to the LFA framework. In this table, we consider the scale of g's weights when $\mathcal{X} \in \mathbb{R}^d$.

A.5 Setup of Experiments

Datasets. The first dataset is the life expectancy dataset from the Global Health Observatory data repository of the World Health Organization (WHO) [29]. The WHO dataset consists of demographic, economic, and health factors of 193 countries from 2000 to 2015, including a country's population, gross domestic product, health expenditure, human development index, infant mortality rate, hepatitis B immunization rate, and life expectancy. The other dataset is the home equity line of credit (HELOC) dataset from the Explainable Machine Learning Challenge organized by FICO [30]. The HELOC dataset contains information on HELOC applications made by homeowners, including an applicant's installment balance, number of trades, longest delinquency period, and risk category (whether an applicant made payments without being 90 days overdue). To our knowledge, these datasets do not contain personally identifiable information or offensive content.

For the WHO dataset, missing values were imputed using kNN imputation with $k = 5$. For the HELOC dataset, missing values were dropped. For both datasets, continuous features were mean-centered and then normalized to [0, 1] range.

Models. For the WHO dataset, we train four models: a linear regression model (train MSE: 9.39×10^{-5}; test MSE: 9.80×10^{-5}) and three feed-forward neural networks. The neural networks have 8-node hidden layers with tanh activation and a linear output layer. The first neural network has 3 hidden layers (train MSE: 7.83×10^{-5}; test MSE: 8.23×10^{-5}), the second has 5 hidden layers (train MSE: 7.76×10^{-5}; test MSE: 8.11×10^{-5}), and the third has 8 hidden layers (train MSE: 7.78×10^{-5}; test MSE: 8.20×10^{-5}). The neural networks are referred to as NN1, NN2, and NN3, respectively.

For the HELOC dataset, we train four models: a logistic regression model (train accuracy: 0.73; test accuracy: 0.74) and three feed-forward neural networks. The neural networks have 8-node hidden layers with relu activation and an output layer with sigmoid activation. The first neural network has 3 hidden layers (train accuracy: 0.75; test accuracy: 0.75), the second has 5 hidden layers (train accuracy: 0.75; test accuracy: 0.75), and the third has 8 hidden layers (train accuracy: 0.75; test accuracy: 0.75). The neural networks are referred to as NNA, NNB, and NNC, respectively.

Models were trained based on an 80/20 train/test split using stochastic gradient descent. Hyperparameters were selected to reach decent model performance. The emphasis is on generating explanations for individual model predictions, not on high model performance. Thus, we do not focus on tuning model hyperparameters. Linear and logistic regression models trained for 100 epochs while neural network models trained for 300 epochs. All models used a batch size of 64 and a cosine annealing scheduler for the learning rate. Hyperparameters for all models are included in the code accompanying this paper.

Explanation Methods. Each explanation method is implemented using (1) the existing method and (2) the LFA framework. For (1), we used Meta's Captum library [31]. When using Captum, methods with number of perturbations as a parameter (i.e., LIME, KernelSHAP, SmoothGrad, and Integrated Gradients) used 1000 perturbations, a number of perturbations at which explanations for the method converged. For (2), we implemented the LFA framework, instantiating each method based on Table 1. For each method, the number of perturbations is set to 1000 for the same reason above. The interpretable model g is optimized using stochastic gradient descent. The perturbations are split into a train and test set (80/20 split) and g^* is optimized based on test set performance.

Analyses were performed on GPUs. The total amount of compute is approximately 54 GPU-hours.

A.6 Full Results for Experiments

A.6.1 Experiment 1: Existing Methods Are Instances of the LFA Framework

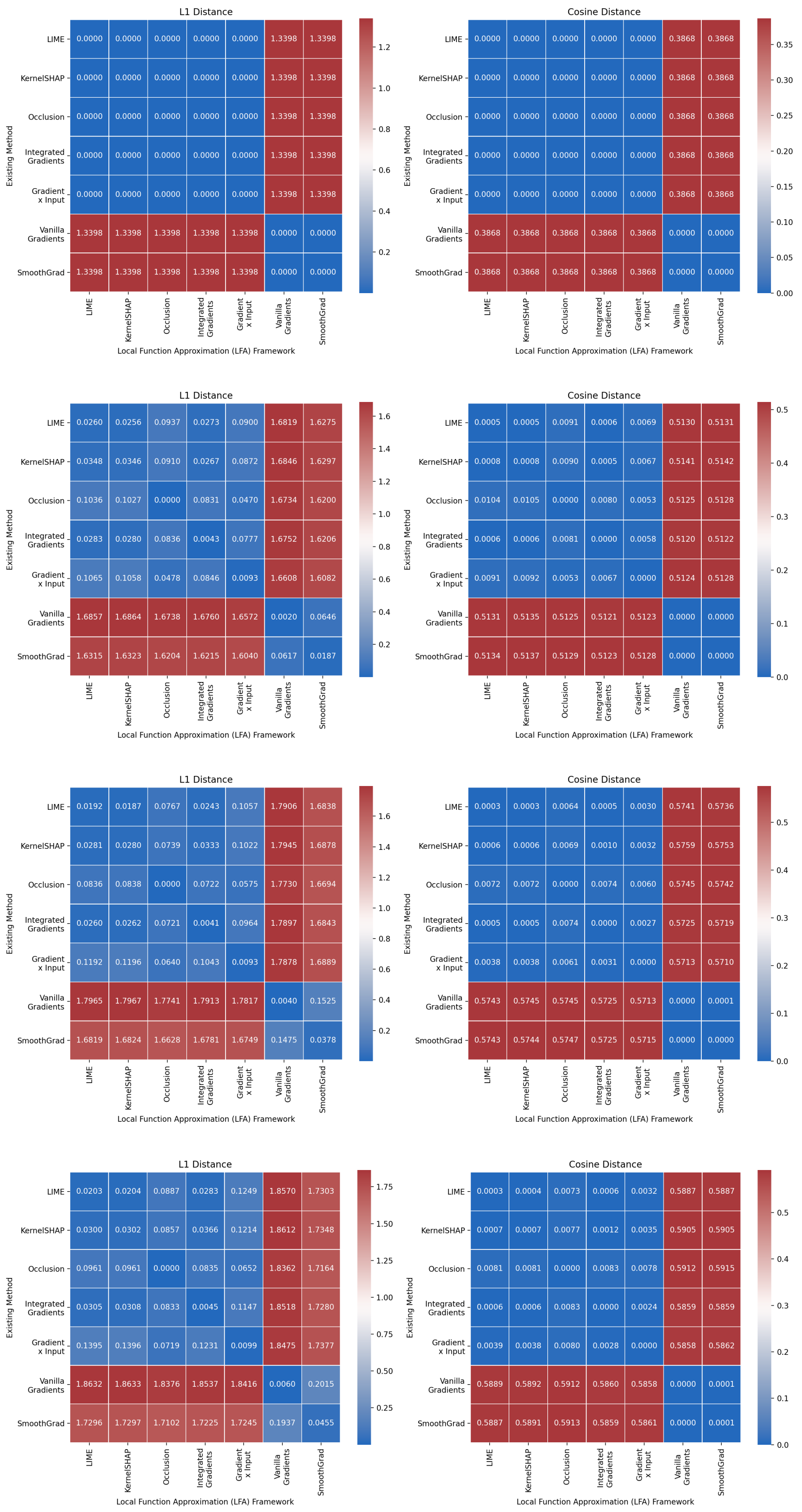

Figure 4: Correspondence of existing methods to instances of the LFA framework. Experiments performed on the WHO dataset for linear regression (Row 1), NN1 (Row 2), NN2 (Row 3), and NN3 (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

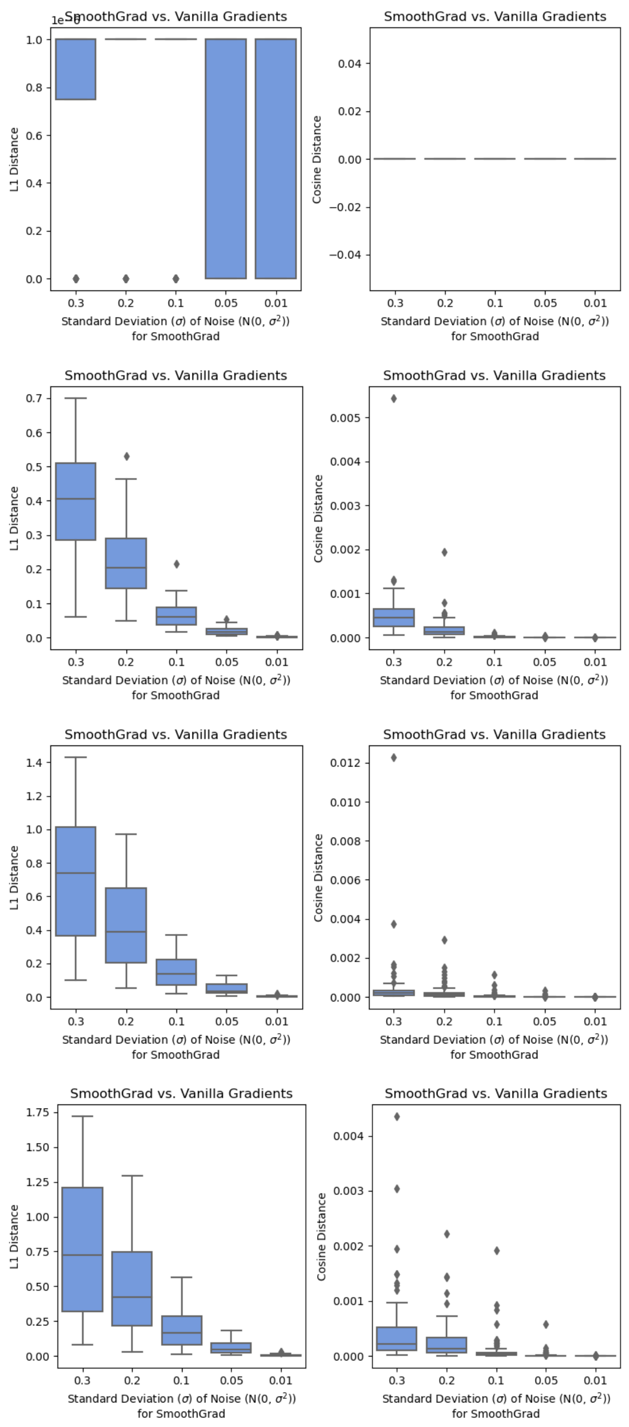

Figure 5: Using the LFA framework, explanations generated by SmoothGrad converge to those generated by Vanilla Gradients. Experiments performed on the WHO dataset for linear regression (Row 1), NN1 (Row 2), NN2 (Row 3), and NN3 (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

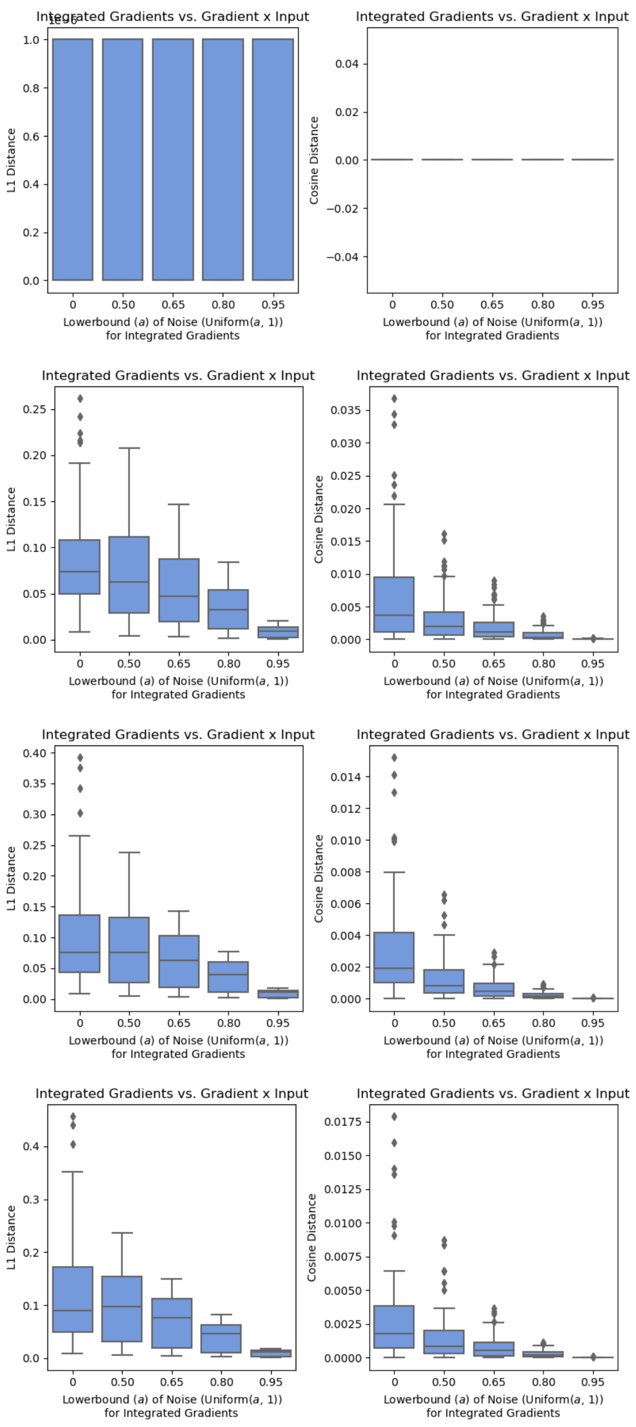

Figure 6: Using the LFA framework, explanations generated by Integrated Gradients converge to those generated by Gradient $\times$ Input. Experiments performed on the WHO dataset for linear regression (Row 1), NN1 (Row 2), NN2 (Row 3), and NN3 (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

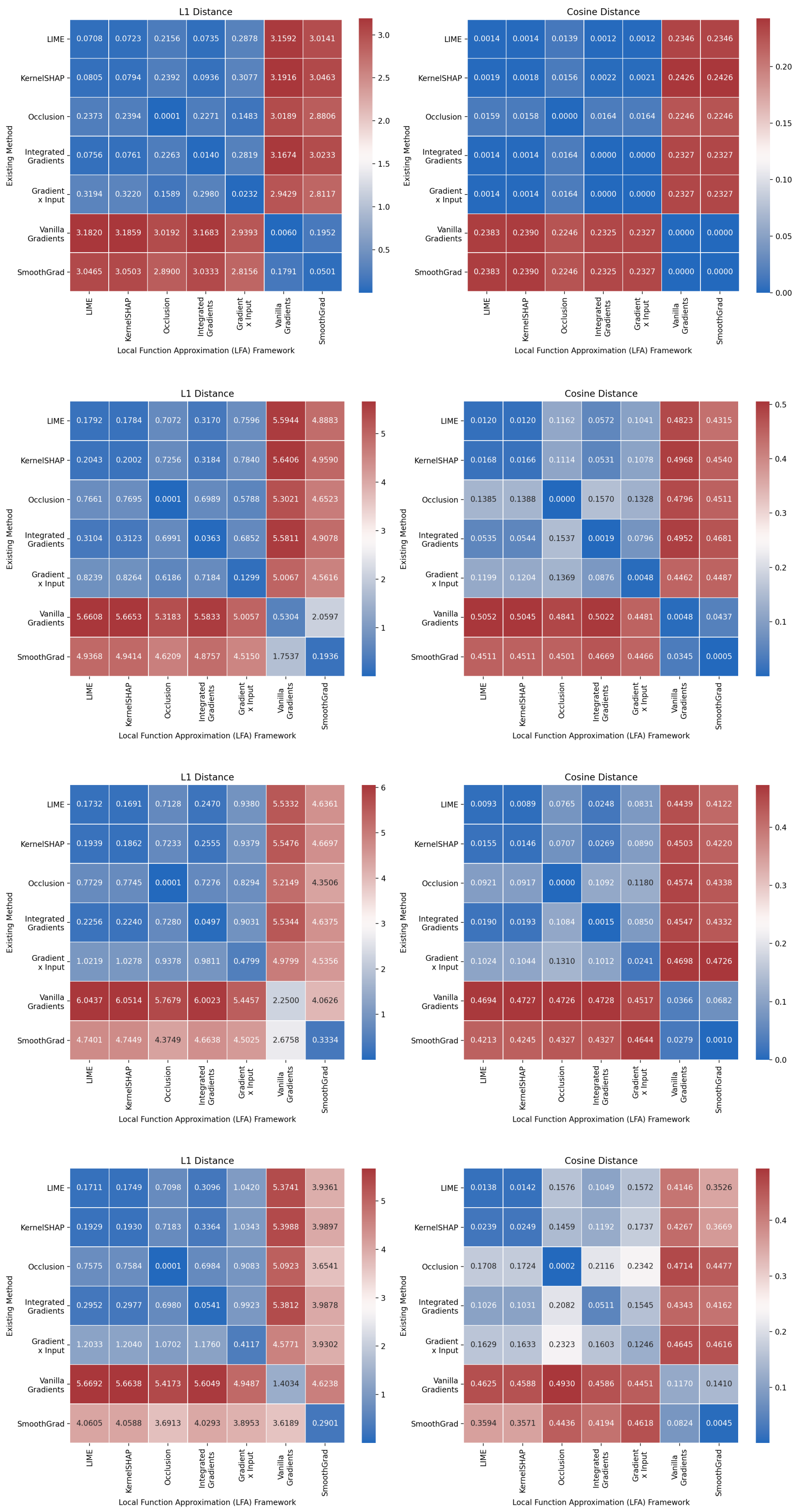

Figure 7: Correspondence of existing methods to instances of the LFA framework. Experiments performed on the HELOC dataset for logistic regression (Row 1), NNA (Row 2), NNB (Row 3), and NNC (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

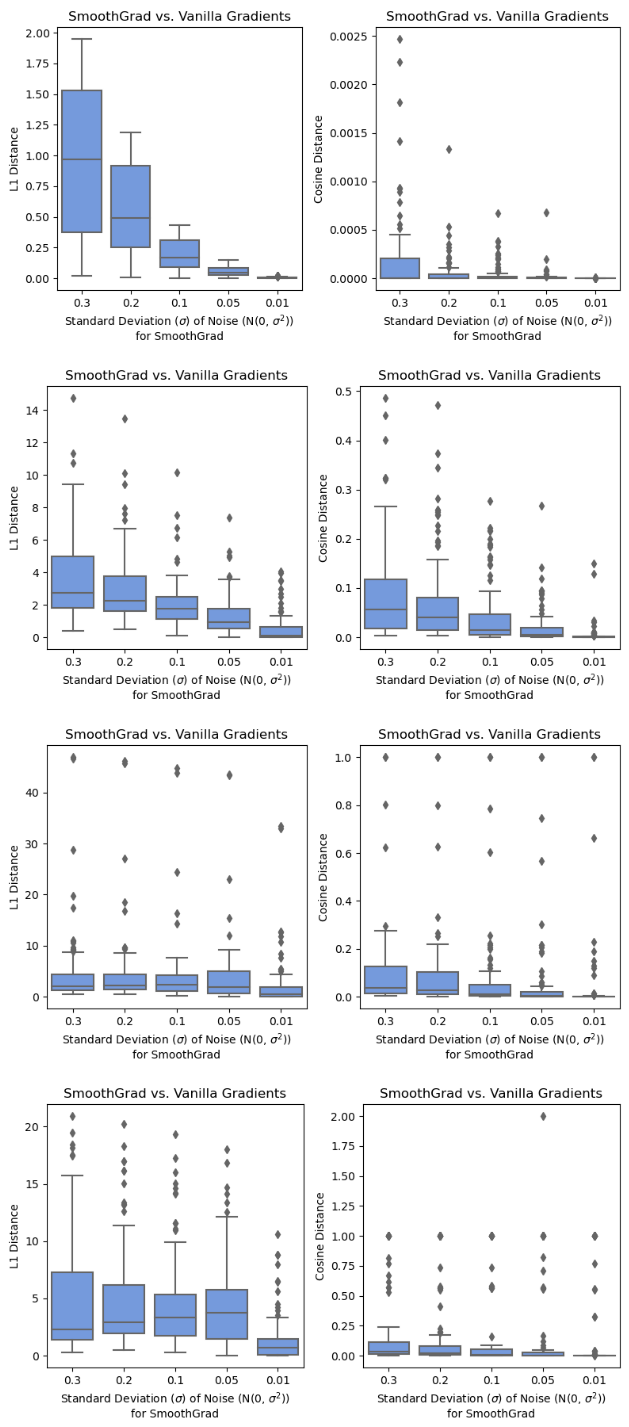

Figure 8: Using the LFA framework, explanations generated by SmoothGrad converge to those generated by Vanilla Gradients. Experiments performed on the HELOC dataset for logistic regression (Row 1), NNA (Row 2), NNB (Row 3), and NNC (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

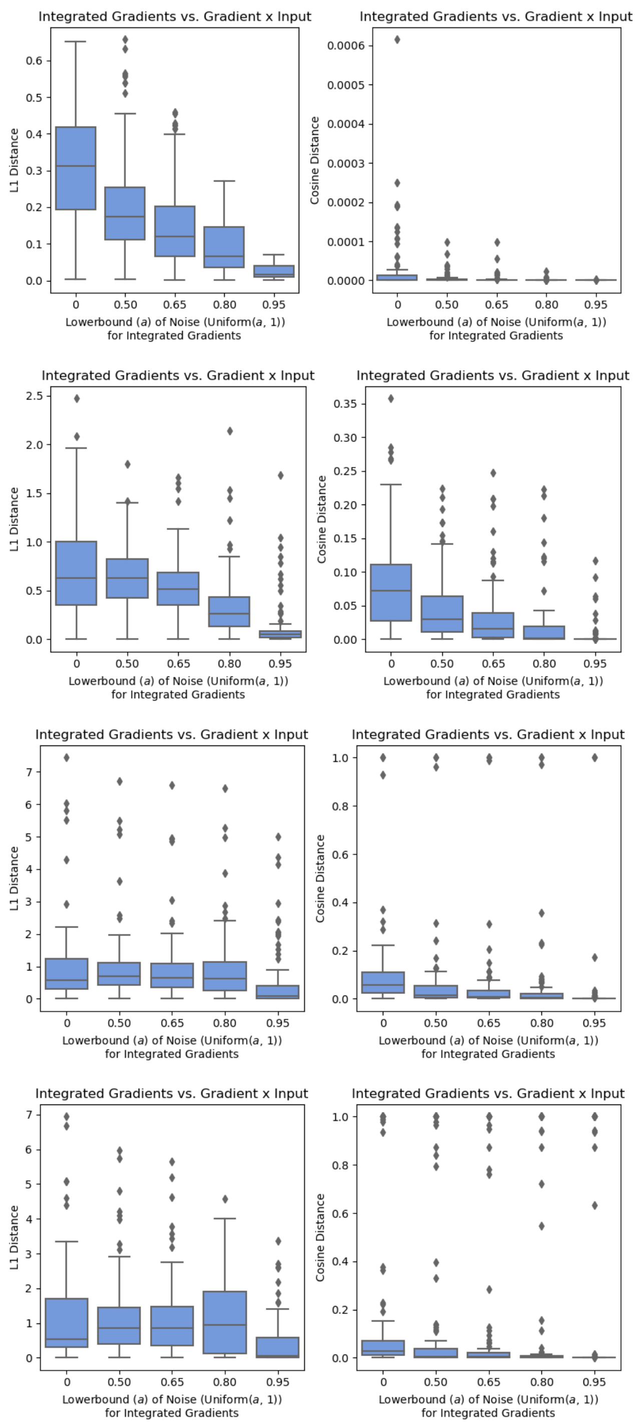

Figure 9: Using the LFA framework, explanations generated by Integrated Gradients converge to those generated by Gradient $\times$ Input. Experiments performed on the HELOC dataset for logistic regression (Row 1), NNA (Row 2), NNB (Row 3), and NNC (Row 4). The similarity of pairs of explanations are measured based on L1 distance (left column) and cosine distance (right column).

A.6.2 Experiment 2: g's recovery of f

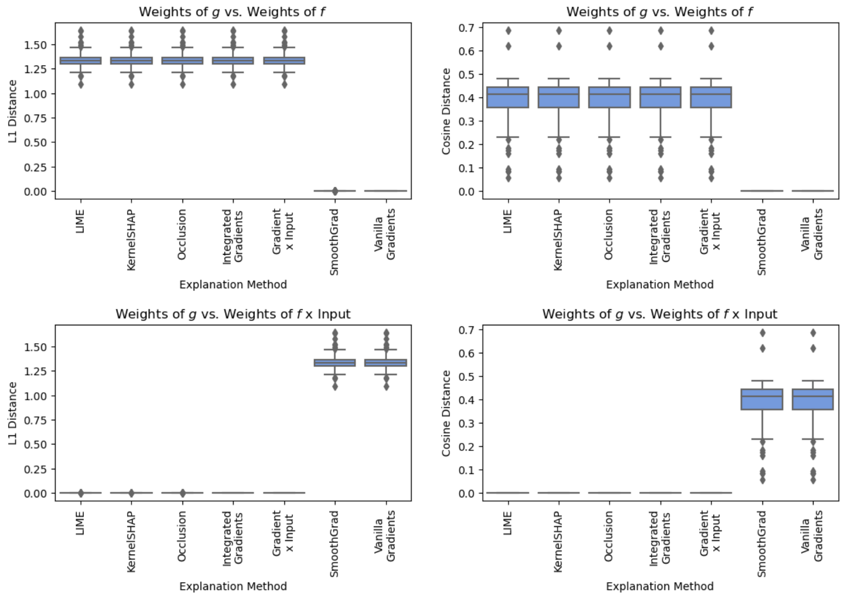

Figure 10: Analysis of g's recovery of f using a linear regression model trained on the WHO dataset. g's weights are compared with f's weights (top row) or f's weights multiplied by the input (bottom row) based on L1 distance (left column) or cosine distance (right column).

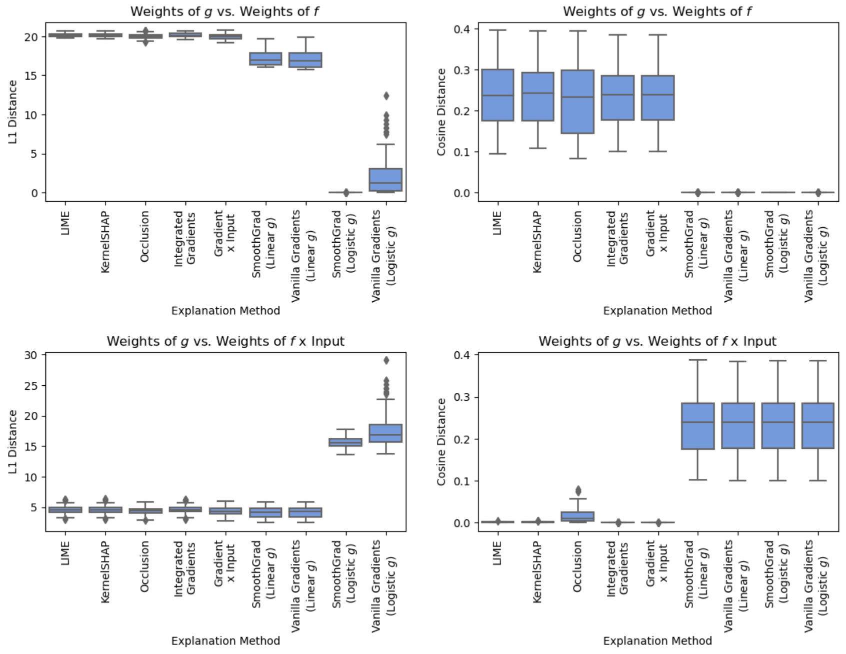

Figure 11: Analysis of g's recovery of f using a logistic regression model trained on the HELOC dataset. g's weights are compared with f's weights (top row) or f's weights multiplied by the input (bottom row) based on L1 distance (left column) or cosine distance (right column).

A.6.3 Experiment 3: Perturbation Tests

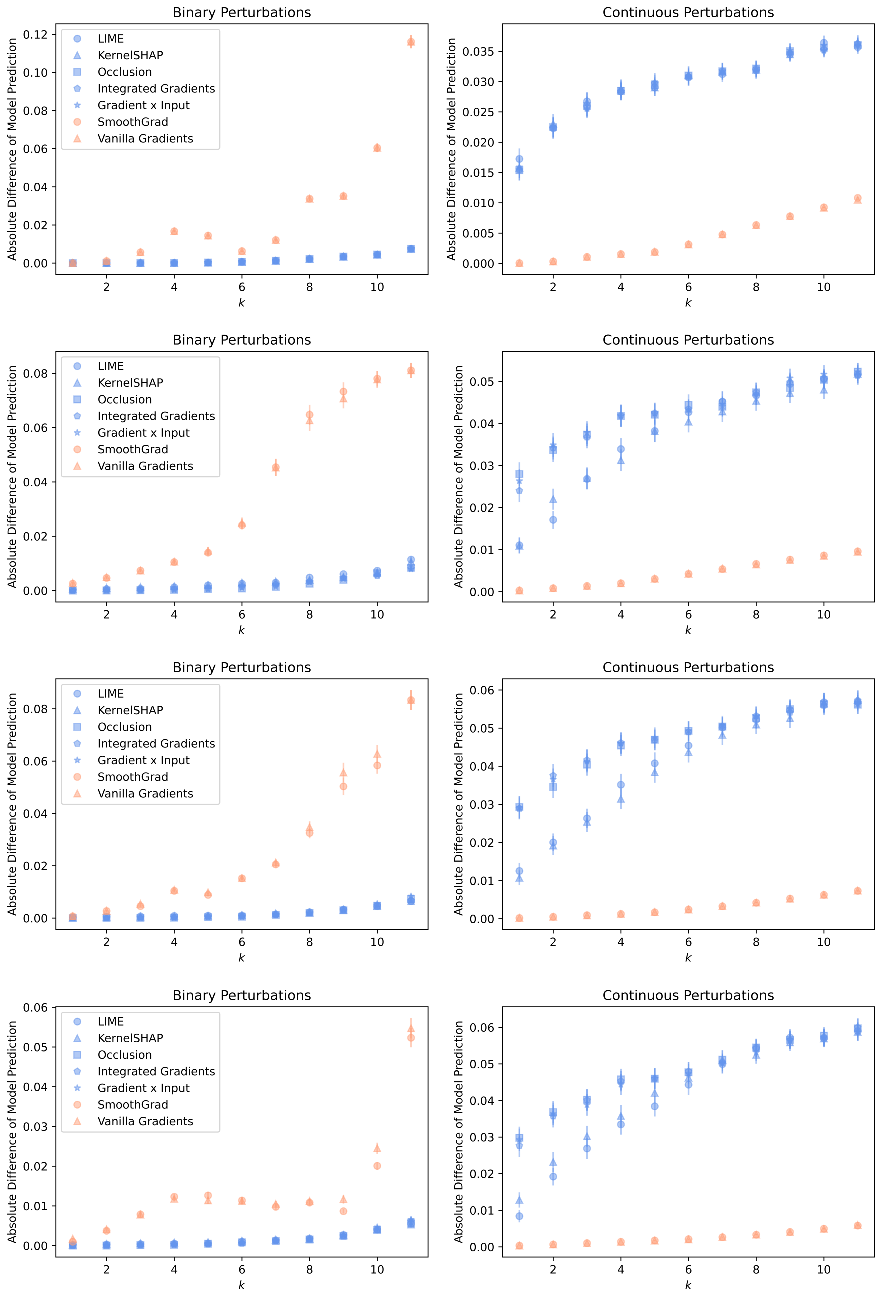

Figure 12: Perturbation tests based on bottom-k features using binary noise (left column) or continuous noise (right column) performed on the WHO dataset for linear regression (Row 1), NN1 (Row 2), NN2 (Row 3), and NN3 (Row 4). The lower the curve, the better a method identifies unimportant features. (Note: Row 2 is a duplicate of Figure 3).

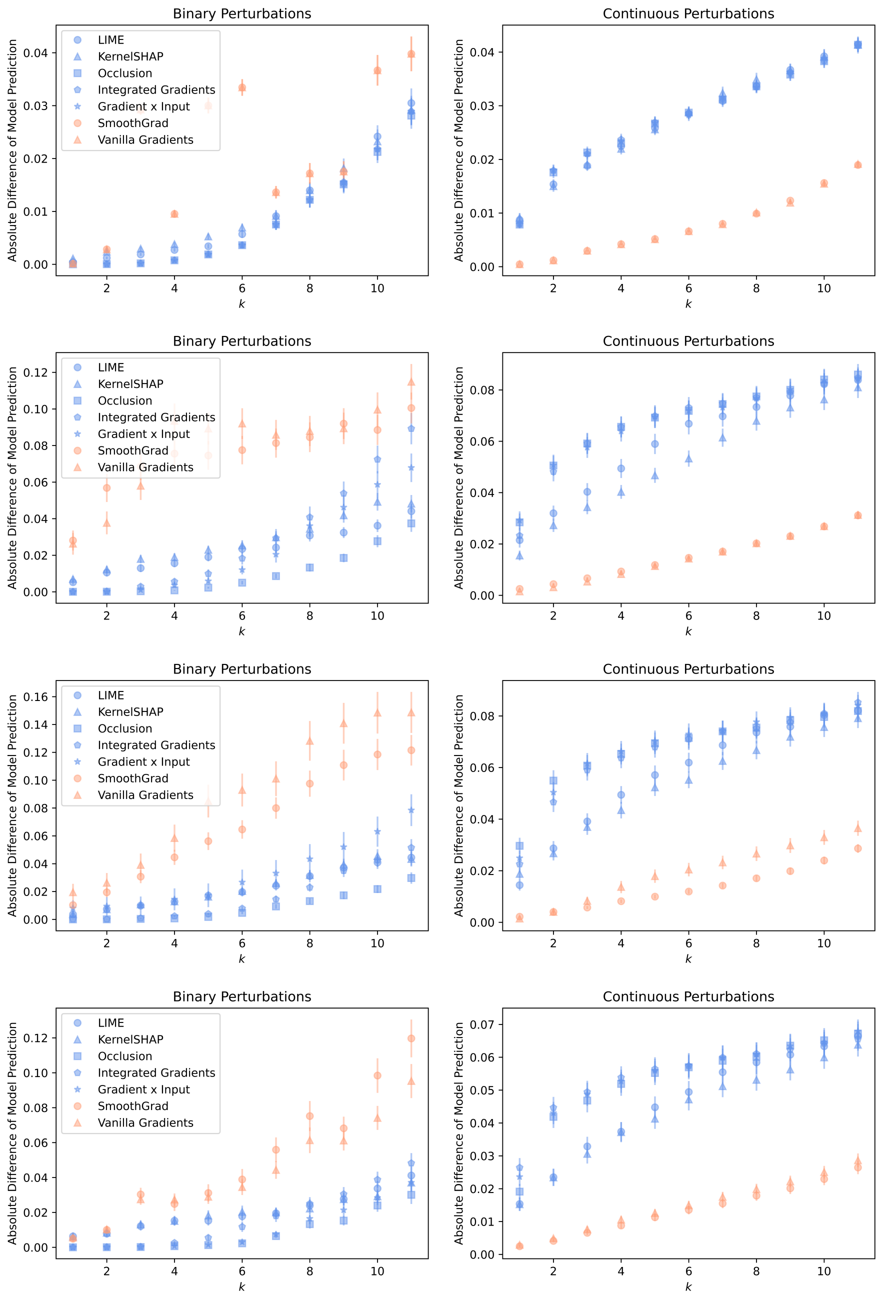

Figure 13: Perturbation tests based on bottom-k features using binary noise (left column) or continuous noise (right column) performed on the HELOC dataset for logistic regression (Row 1), NNA (Row 2), NNB (Row 3), and NNC (Row 4). The lower the curve, the better a method identifies unimportant features.

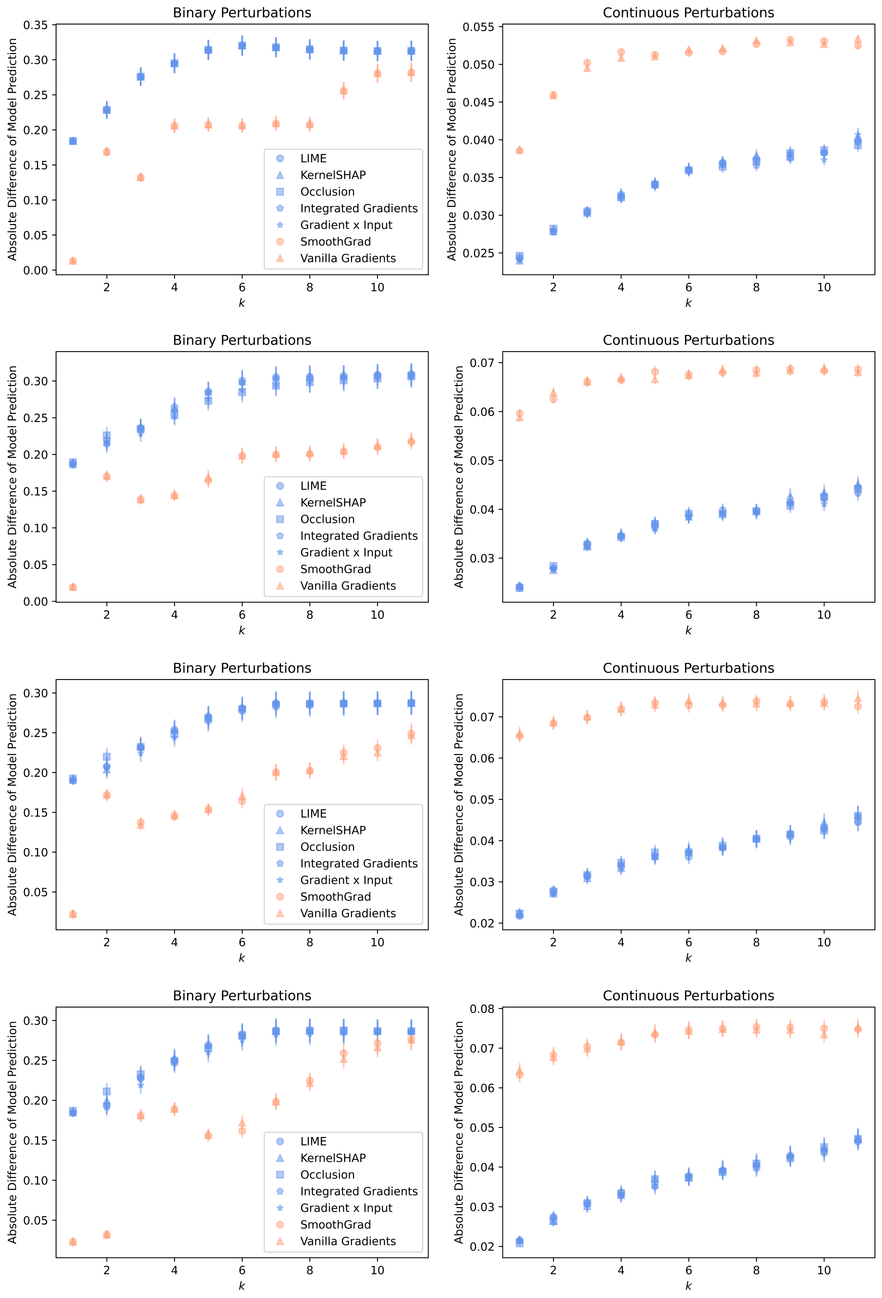

Figure 14: Perturbation tests based on top-k features using binary noise (left column) or continuous noise (right column) performed on the WHO dataset for linear regression (Row 1), NN1 (Row 2), NN2 (Row 3), and NN3 (Row 4). The higher the curve, the better a method identifies important features.

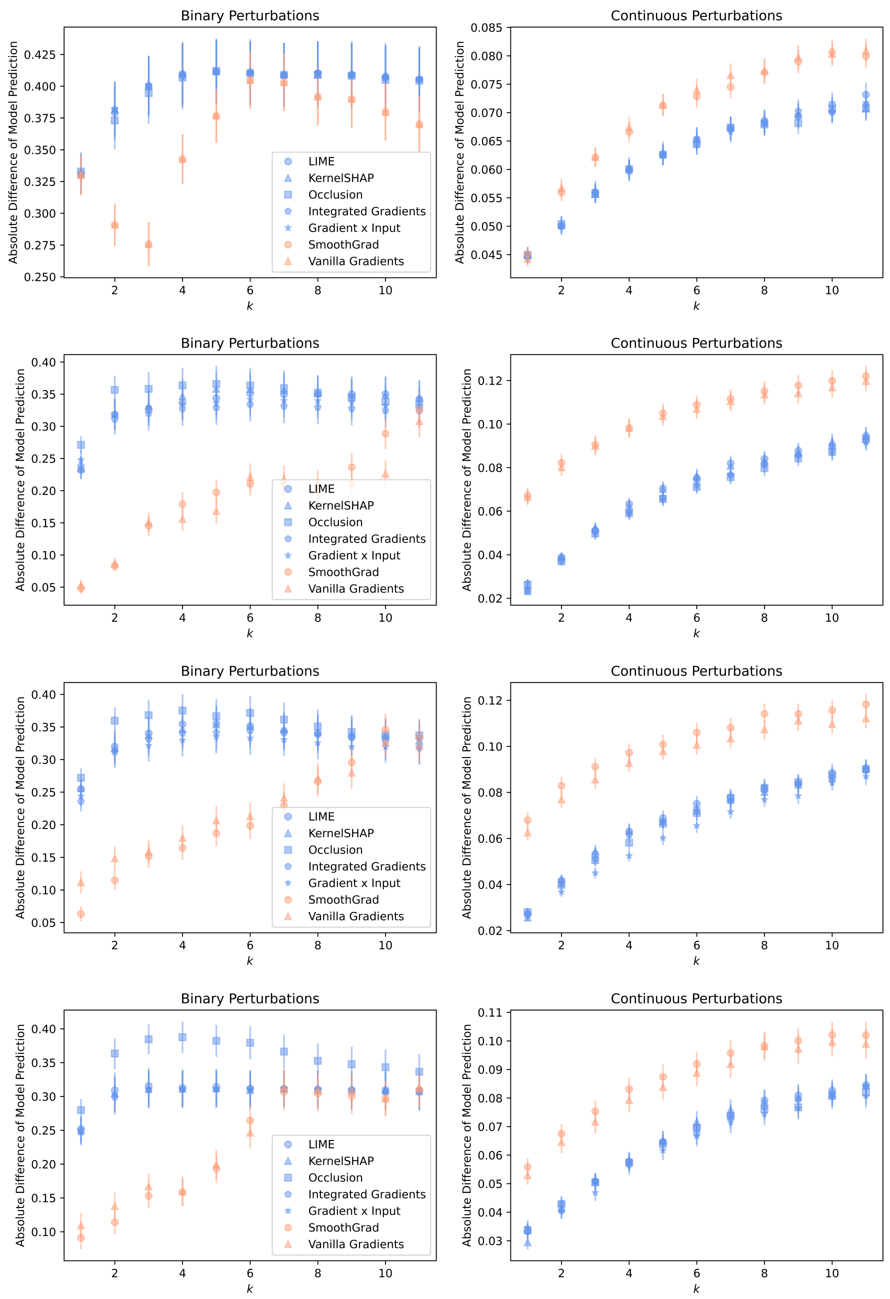

Figure 15: Perturbation tests based on top-k features using binary noise (left column) or continuous noise (right column) performed on the HELOC dataset for logistic regression (Row 1), NNA (Row 2), NNB (Row 3), and NNC (Row 4). The higher the curve, the better a method identifies important features.