Towards Unifying Feature Attribution and Counterfactual Explanations: Different Means to the Same End

Ramaravind K. Mothilal
Microsoft Research India
t-rakom@microsoft.com

Divyat Mahajan
Microsoft Research India
t-dimaha@microsoft.com

Chenhao Tan
University of Colorado Boulder
chenhao.tan@colorado.edu

Amit Sharma
Microsoft Research India
amshar@microsoft.com

ABSTRACT

To explain a machine learning model, there are two main approaches: feature attributions that assign an importance score to each input feature, and counterfactual explanations that provide input examples with minimal changes to alter the model's prediction. We provide two key results towards unifying the approaches in terms of their interpretation and use. First, we present a method to generate feature attribution explanations from a set of counterfactual examples. These feature attributions convey how important a feature is to changing the classification outcome of a model, especially on whether a subset of features is necessary and/or sufficient for that change, which feature attribution methods are unable to provide. Second, we show how counterfactual examples can be used to evaluate the goodness of an attribution-based explanation in terms of its necessity and sufficiency. As a result, we highlight the complementarity of these two approaches and provide an interpretation based on a causal inference framework. Our evaluation on three benchmark datasets—Adult Income, LendingClub, and GermanCredit—confirm the complementarity. Feature attribution methods like LIME and SHAP and counterfactual explanation methods like DiCE often do not agree on feature importance rankings. In addition, by restricting the features that can be modified for generating counterfactual examples, we find that the top-k features from LIME or SHAP are neither necessary nor sufficient explanations of a model's prediction. We conclude by presenting a case study of using different explanation methods on a real-world hospital triage problem.

ACM Reference Format:
Ramaravind K. Mothilal, Divyat Mahajan, Chenhao Tan, and Amit Sharma. 2018. Towards Unifying Feature Attribution and Counterfactual Explanations: Different Means to the Same End. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/1122445.1122456

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

As complex machine learning models are being deployed in high-stakes domains like finance and healthcare, explaining why they make a certain prediction has emerged as a critical task. Explanations of a ML model's prediction have found many uses, including to understand the most important features, discover any unintended bias [39], debug the model [22, 34], increase trust [25], and provide recourse suggestions for unfavorable predictions [30, 38, 46, 47].

There are two main kinds of explanations: attribution-based [12, 27, 34] and counterfactual-based [30, 47]. Attribution-based explanations provide a score or ranking over features, conveying the relative importance of each feature to the model's output. These are generally by local function approximation methods using linear [34] or decision-tree [12] models, or by using game-theoretic attribution such as Shapley values [27]. The second kind, counterfactual-based explanations, instead generate examples that have an alternative model output with minimum changes in the input features, known as counterfactual examples (CF) [47]. Because of the differences in the type of explanation output and how they are generated, attribution-based and counterfactual-based explanations are studied independent of each other.

In this paper we demonstrate the fundamental connections between attribution-based and counterfactual-based explanations. First, we show how counterfactual-based explanations can be used to evaluate attribution-based explanations on key properties. In particular, we consider the necessity (is a feature value necessary for the model's output?) and the sufficiency (is the feature value sufficient for generating the model output?). Second, we propose a simple method by which counterfactual-based explanations can generate an importance ranking for features, just like attribution-based explanations, and study the correlation between these feature importance scores. Therefore, rather than being separate, they are complementary methods towards the same goal.

To provide a formal connection, we introduce the framework of *actual causality* [14] to the explanation literature that reasons about the causes of a particular event, unlike the more common causal inference setting that estimate the effect of a particular cause [32, 36]. Using actual causality, we provide a definition of a model explanation and propose two desirable properties of an explanation: *necessity* and *sufficiency* for generating a given model output. A good explanation should satisfy both, but we find that current explanation methods optimize either one of them. Counterfactual-based methods like DiCE [30] find examples that highlight the necessary feature value for a given model output whereas attribution-based methods like LIME [34] and SHAP [27] focus on the sufficiency of a feature value. Thus, the actual causality framework underscores the complementarity of the two approaches: to achieve a good explanation, we need to provide both necessity and sufficiency.

Using LIME and SHAP as examples of attribute-based and DiCE as an example of a counterfactual-based method, we empirically study this complementarity in two ways. First, we show that examples generated from DiCE can be used to evaluate explanations from LIME and SHAP. By allowing only a specific feature to change in generating CFs, we can evaluate the necessity of the feature's value for the model's predicted output. Similarly, by generating CFs by changing all but a specific feature, we can evaluate the sufficiency of the feature's value for causing the model's outcome. On benchmark datasets related to income or credit predictions (Adult-Income, German-Credit and LendingClub), we find that the top-ranked features from LIME and SHAP are neither necessary nor sufficient. In particular, for Adult-Income and German-Credit, more counterfactuals with alternative predicted outcomes can be generated by using features except the top-3 than using any of the top-3 features, and it is easy to generate counterfactuals even if one of the top-ranked features is not changed at all.

Second, we show that counterfactual examples can be used to generate feature importance scores that can complement the scores from LIME and SHAP. The scores from DiCE and atttribution-based methods do not always agree: compared to other features, DiCE tends to assign relatively higher scores to low-ranked features from LIME and SHAP, likely because it is possible to generate valid CFs using those features as well. Ranks generated by the three methods also disagree: not only does DiCE disagree with LIME and SHAP, but LIME and SHAP also disagree on many features.

Finally, we evaluate the explanation methods on a high-dimensional real-world dataset that has over 200 features and the ML model predicts whether a patient will be admitted to a hospital. The differences observed above are magnified: an analyst may reach widely varying conclusion about the ML model depending on which explanation method they choose. DiCE considers triage features as the most important, LIME considers chief-complaint features as the most important, while SHAP identifies demographic features as the most important. We also find odd results with LIME on necessity: changing the third most important feature provides more valid CFs than changing the most important feature (3rd > 1st).

Our results reveal the importance of considering multiple explanation methods to understand the prediction of an ML model. Different explanation methods have different objectives (and empirical approximations). Hence, a single method may not convey the full picture. To summarize, we make the following contributions.

- Unifying framework for attribute-based and counterfactual examples using actual causality;
- A method to generate feature importance scores from counterfactual examples;
- A method to evaluate attribution-based methods on the necessity and sufficiency of their top-ranked features.
- A systematic empirical investigation using both commonly-used low-dimensional datasets and a high-dimensional dataset.

2 RELATED WORK

We discuss the desirable properties that any explanation method should have, the two main types of explanations, and how different explanation methods compare to each other. There is also important work on building intelligible models by design [6, 26, 37] that we do not discuss here.

2.1 Desirable Properties of an explanation

Explanations serve a variety of purposes, including debugging for the model-developer, evaluating properties for an auditor, and providing recourse and trust for an end individual. Therefore, it is natural that explanations have multiple desirable properties based on the context. Sokol and Flach [42] and Miller [28] list the different properties that an explanation ideally should adhere to. Different works have evaluated the soundness (truthfulness to the ML model), completeness (generalizability to other examples), parsimony, and actionability of explanations. In general, counterfactual-based methods optimize soundness over completeness, while methods that summarize data to produce an attribution score are less sound but optimize for completeness.

In comparison, the notions of necessity and sufficiency of a feature value for a model's output are less studied. In natural language processing (NLP), sufficiency and comprehensiveness have been defined based on the output probability in the context of rationale evaluation (e.g., whether a subset of words leads to the same predicted probability as the full text) [9, 48]. By using a formal framework of actual causality [14], we define the necessity and sufficiency metrics for an explanation, provide a method using counterfactual examples to compute them, and evaluate common explanation methods on them.

2.2 Attribution-based and Counterfactuals

Majority of the work in explainable ML provides attribution-based explanations [40, 44]. These are based on approximating the local decision boundary [34] or by estimating the Shapley value of each feature as its importance in contributing to the model prediction [27]. In contrast, counterfactual explanations [10, 33, 47] search for the smallest change that lead to a change in the outcome. When generating a set of counterfactual explanations, diversity among them is important [30, 38].

We provide a unified view of these two explanations. They need not be considered separate: counterfactuals can be another way to generate feature attributions like model approximation or Shapley values, as suggested by Barocas et al. [5]. We extend their work to present a concrete method to generate such attributions.

2.3 Comparing different explanation methods

Given the number of different explanation methods, it becomes important to reliably compare them and choose suitable methods for an explanation goal. In general, however, comparing explanation methods is difficult because they all produce a feature ranking motivated by a particular theory of importance, and there is no principled way to compare against them. What should one do when two attribution-based methods do not agree?

First, we provide empirical evidence that common explanation methods often disagree, on a wide range of datasets, confirming past work that compares feature attribution methods in NLP [21]. Second, we provide two causality-inspired metrics of necessity and sufficiency that can be used to compare attribution-based methods.

3 UNIFYING EXPLANATION METHODS USING ACTUAL CAUSALITY

Let $f(\boldsymbol{x})$ be a machine learning model and $\boldsymbol{x}$ denote a vector of k features, $x_1, x_2, ...x_k$. Given input $\boldsymbol{x}_0$ and the output $f(\boldsymbol{x}_0)$, a common explanation task is to determine which features are responsible for this particular prediction. There are two main types of explanation: attribution-based and counterfactual-based. Feature attribution methods [27, 34] are local explanation techniques that assign *importance* scores to features based on certain criteria. A feature's score captures its contribution to the predicted value of a test instance. On the other hand, counterfactual explanations are minimally-tweaked version of the original input that leads to a different predicted outcome than the original prediction [47].

Though both methods aim to explain a model's output at a given input, the difference and similarity in their implications are not clear. While feature attributions highlight features that are important in terms of their contributions to the model prediction, it does not imply that changing important features is sufficient or necessary to lead to a different (desired) outcome. Similarly, while CF explanations provide actionable insights required to reach a different outcome, the features changed may not include the most important features of feature attribution methods.

Below we show that while these explanation methods may appear distinct, they are all motivated by the same principle of whether a feature is a "cause" of the model's prediction, and to what extent. We provide a formal framework based on actual causality [14] to interpret these different explanation methods.

3.1 Background: Actual cause and explanation

We first define *actual cause* and how it can be used to explain an event. In our case, the classifier's prediction is an event, and the input features are the potential causes of the event. According to Halpern [14], causes of an event are defined w.r.t to a structural causal model (SCM) that defines the relationship between the potential causes and the event. In our case, the learnt ML model f is the SCM (M) that governs how the prediction output is generated from the input features. The structure of the SCM consists of each feature as a node that causes other intermediate nodes (e.g., different layers of a neural network), and then finally leads to the output node. We assume that the feature values are generated from an unknown process governed by a set of parameters that we collectively denote as $\boldsymbol{u}$, or the *context*. Together, $(M, \boldsymbol{u})$ define a specific configuration of the input $\boldsymbol{x}$ and the output $f(\boldsymbol{x})$ of the model.

Definition 3.1 (Actual Cause, (Original definition) [14]). A subset of features $\boldsymbol{x}_j = a$ is an actual cause of the prediction $f(\boldsymbol{x}_{-j} = b, \boldsymbol{x}_j = a) = y^*$ under the causal setting $(M, \boldsymbol{u})$ if all the following conditions hold:

(1) Given $(M, \boldsymbol{u})$, $\boldsymbol{x}_j = a$ and $f(\boldsymbol{x}_{-j} = b, \boldsymbol{x}_j = a) = y^*$.
(2) There exists a subset of features $W \subseteq \boldsymbol{x}_{-j}$ such that if W is set to w', then $(x_j \leftarrow a, W \leftarrow w') \Rightarrow (y = y^*)$ and $(x_j \leftarrow a', W \leftarrow w') \Rightarrow y \neq y^*$ for some value a'.
(3) $\boldsymbol{x}_j$ is minimal, namely, there is no strict subset $\boldsymbol{x}_s \subset \boldsymbol{x}_j$ such that $\boldsymbol{x}_s = a_s$ satisfies conditions 1 and 2, where $a_s \subset a$.

In the notation above, $\boldsymbol{x}_i \leftarrow v$ denotes that $\boldsymbol{x}_i$ is intervened on and set to the value v, irrespective of its observed value under $(M, \boldsymbol{u})$. Intuitively, a feature is an actual cause if under some value b' of the other features $\boldsymbol{x}_{-j}$, there exists a value $a' \neq a$ such that $f(\boldsymbol{x}_{-j} = b', a') \neq y^*$ and $f(\boldsymbol{x}_{-j} = b', a) = y^*$. For instance, consider a linear model with three binary features $f(x_1, x_2, x_3) = I(0.4x_1 + 0.1x_2 + 0.1x_3 >= 0.5)$ and a prediction of $y = 1$. Here each feature can be considered an actual cause for the model's output, since there is a context where its value becomes necessary to flip the outcome.

To differentiate between the contributions of features, we can use a stronger definition, the *but-for* cause.

Definition 3.2 (But-for Cause). A feature value $x_{0,j} = a$ is a but-for cause of the prediction $f(\boldsymbol{x}_{-j} = b, x_j = a) = y_0$ if it is an actual cause and the empty set $W = \phi$ satisfies condition 2.

That is, changing the value of x_j alone changes the prediction of the model at $\boldsymbol{x}_0$. On the linear model, now it provides a better picture: x_1 is always a but-for cause for $y^* = 1$ but when $y^* = 0$, that is true only in certain special cases. The only context in which x_2 and x_3 are but-for causes for $y^* = 1$ is when $x_1 = 1$.

While the notion of but-for causes captures the necessity of a particular feature subset for the obtained model output, it does not capture sufficiency. Sufficiency means that setting a feature subset $\boldsymbol{x}_j \leftarrow a$ will always lead to the given model output, irrespective of the values of other features. To capture sufficiency, we need an additional condition.

Definition 3.3 (Sufficient Cause). A feature value $\boldsymbol{x}_j = a$ is a sufficient cause of the model output $f(\boldsymbol{x}_{-j} = b, \boldsymbol{x}_j = a) = y^*$ if,

(1) it is an actual cause (satisfies conditions 1-3)
(2) For all contexts $u \in U$, $x_j \leftarrow a \Rightarrow y = y^*$.

Based on the above definitions, we are now ready to define an ideal explanation.

*Definition 3.4 (**Ideal Model Explanation**).* A feature value $\boldsymbol{x}_j = a$ is an explanation for a model output y^* if it is a sufficient cause of the output, under all contexts.

This definition captures the intuitive meaning of explanation: the feature affects the output (output changes if the feature is changed under certain conditions), and as long as the feature is unchanged, the output cannot be changed. In practice, however, it is rare to find clean explanations of a ML model's output. Even in our simple linear model above, no feature is sufficient to cause the output.

3.2 Partial Explanation for Model Output

For most realistic ML models, an ideal explanation is impractical. Therefore, we now describe the concept of *partial* explanations [14] that relaxes the necessity and sufficiency conditions to consider the fraction of contexts over which these conditions are valid. Partial explanations are characterized by two metrics.

The first metric captures the extent to which a feature value is *necessary* to cause the model's (original) output.

$$\alpha = P(x_j \text{ is a cause of } y^* | x_j = a, y = y^*) \quad (1)$$

where 'is a cause' means that $x_j = a$ satisfies Definition 3.1. The second metric captures *sufficiency* using conditional probability of outcome given the feature's value.

$$\beta = \Pr(y = y^* | x_j \leftarrow a) \tag{2}$$

where $x_j \leftarrow a$ denotes an intervention to set x_j to a. Both probabilities are over the set of contexts. Combined, they can be called (α, β) goodness of an explanation. When both $\alpha = 1$ and $\beta = 1$, $\alpha = 1$ captures that $x_j = a$ is a necessary cause of $y = y^*$ and $\beta = 1$ captures that $x_j = a$ is a sufficient cause of $y{=}y^*$. In other words, a feature value $x_j = a$ is a good explanation for a model's output y^* if the feature value is an actual cause of the outcome and $y = y^*$ with high probability whenever $x_j = a$.

3.3 Unifying different local explanations

Armed with the (α, β) goodness of explanation metrics, we now show how common explanation methods can be considered as special cases of the above framework. We first consider the definition of (α, β) when only but-for causes are allowed, and then consider β for general actual causes.

3.3.1 Only but-for causes. When only but-for causes are allowed, α and β are related to the idea of counterfactual-based explanations. Given y^* and a candidate feature subset $\boldsymbol{x}_j$, α corresponds to fraction of contexts where x_j is a but-for cause. That means, keeping everything else constant and only changing $\boldsymbol{x}_j$, how often does the classifier's outcome change? Eqn. 1 reduces to,

$$\alpha_{CF} = \Pr((\boldsymbol{x}_j \leftarrow a' \Rightarrow y \neq y^*) | \boldsymbol{x}_j = a, y = y^*) \tag{3}$$

where the above probability is over a reasonable set of contexts (e.g., all possible values for discrete features and a bounded region around the original feature value for continuous features). By definition, each of the perturbed inputs above that change the value of y can be considered as a counterfactual example [47]. Counterfactual explanation methods aim to find the smallest perturbation in the feature values that change the output, and correspondingly the modified feature subset x_j is a but-for cause of the output. α_{CF} provides a metric to summarize the outcomes of all such perturbations and to rank any feature subset for their necessity in generating the original model output. In practice, however, computing α is computationally prohibitive and therefore explanation methods empirically find a set of counterfactual examples and allow (manual) analysis on the found counterfactuals. In §4, we will see how we can develop a feature importance score using counterfactuals that is inspired from the α_{CF} formulation.

β corresponds to the fraction of contexts where $x_j = a$ is sufficient to keep $y = y^*$. That corresponds to the degree of sufficiency of the feature subset. That means, keep $\boldsymbol{x}_j$ constant but change everything else and check how often the outcome remains the same. While not common, such a perturbation can be considered as a special case of the counterfactual generation process, where we specifically restrict change in the given feature set. A similar idea is explored in (local) anchor explanations [35]. It is also related to pertinent positives and pertinent negatives [10].

3.3.2 General actual causes. When all actual causes are allowed, α can be defined as the fraction of all contexts where setting $x_j \neq a$ leads to a change in y and in the same context, $x_j = a$ obtains the same y. Depending on how we define the set of *all* contexts, we obtain different local attribute-based explanation methods. For a binary features dataset, the total number of contexts is 2^m. For continuous features, the number of contexts is infinite. For ease of exposition, we consider binary features below.

LIME can be interpreted as estimating β for a restricted set of contexts (random samples) near the input point. Rather than checking Definition 3.3 (condition 2) for each of the random sampled points and estimating the probability of its satisfaction, it uses linear regression to estimate $\beta(a, y^*) - \beta(a', y^*)$. Note that linear regression estimates $z : \mathbb{E}[Y|x_j = a'] - \mathbb{E}[Y|x_j = a]$ which is equivalent to $\Pr(Y = 1|x_j = a') - \Pr[Y = 1|x_j = a]$ for a binary y. It then estimates effects for all features at once using linear regression, assuming that the importance of each feature is independent.

Shapley value-based methods take a different approach, based on estimating β over a restricted set of contexts. Shapley values are defined as the number of times including a feature leads to the positive outcome, averaged over all possible configurations of other input features. That is, they define the valid contexts for a feature value as all valid configurations of the other features (size 2^{m-1}). The intuition is to see, at different values of other features, whether the given feature value is sufficient to cause the desired model output y^*. The goal of estimating Shapley values directly corresponds to the equation for β described above.

Note how selection of the contexts effectively defines the type of attribution-based explanation method [20, 43]. For example, we may weigh the contexts based on their likelihood to obtain a probability distribution over contexts, that leads to feasible attribute explanations [2]. These different contexts also mean that explanation methods based on them will have different properties.

The above analysis indicates that different explanation methods optimize for either α or β: counterfactual explanations are inspired from the α_{CF} metric and attribution-based methods like LIME and SHAP are approximations for computing β. Since a good explanation ideally needs both high α and β, our framework suggests that there is value in evaluating both these parameters for an explanation method, and in general considering both types of explanations for their complementary value in understanding a model's output. In the following sections, we propose methods for evaluating necessity (α_{CF}) and sufficiency (β) of an explanation and study their implications for multiple real-world datasets.

4 PROPOSED METHODS

To connect attribution-based methods with counterfactual explanation, we propose two methods. The first measures the necessity and sufficiency of any attribution-based explanation using counterfactuals, and the second creates feature importance scores using counterfactual examples.

4.1 Explanation methods

For our empirical evaluation, we use two attribute-based methods, LIME [34] and SHAP [27], and a counterfactual method, DiCE [30]. **Attribution-based methods.** We use LIME and SHAP as our feature attribution methods because of their widespread usage and open-source implementations. For a given test instance $\boldsymbol{x}$ and a ML model $f(.)$, LIME perturbs its feature values and uses the perturbed samples to build a local linear model g of complexity $\Omega(g)$ [34]. In general, g can be any simple model from a family of explainable models G. The coefficients of the linear model are used as explanations ζ and larger coefficients imply higher importance. Formally, LIME generates explanations by optimizing the following loss where L measures how close g is in approximating f in the neighborhood of $\boldsymbol{x}$, $\pi_{\boldsymbol{x}}$.

$$\zeta(\boldsymbol{x}) = \underset{g \in G}{\arg\min}\, \mathrm{L}(f, g, \pi_{\boldsymbol{x}}) + \Omega(g) \tag{4}$$

SHAP, on the other hand, assigns importance score to a feature based on Shapley values, which are computed using that feature's average marginal contribution across different coalitions of all features [27]. SHAP borrows axiomatic principles from game theory to guarantee certain desirable properties such as consistency and local accuracy. However, the challenge is in reliably estimating the shapley values that often need approximation methods.

Counterfactual generation method. For counterfactual explanations, we use an open-source tool, DiCE, that solves an optimization problem to generate a diverse set of CFs close to the original input that yield a different model output. DiCE generates a diverse counterfactual set C of length nCF for a test instance $\boldsymbol{x}$ by optimizing the following loss function, where $\boldsymbol{c}_i$ is a counterfactual example:

$$C(\boldsymbol{x}) = \underset{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_{nCF}}{\arg\min} \frac{1}{nCF} \sum_{i=1}^{nCF} \mathrm{yloss}(f(\boldsymbol{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^{nCF} dist(\boldsymbol{c}_i, \boldsymbol{x}) - \lambda_2 \,\mathrm{dpp\_diversity}(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{nCF}) \tag{5}$$

The three additive components in the loss function optimizes (1) $yloss(.)$ between ML model $f(.)$'s prediction and the desired outcome y, (2) distance between $\boldsymbol{c}_i$ and test instance $\boldsymbol{x}$, and (3) diversity of CFs using determinantal point processes based method [19].

4.2 Measuring Necessity and Sufficiency

Suppose $y^* = f(\boldsymbol{x}_j = a, \boldsymbol{x}_{-j} = b)$ is the output of a classifier f to input $\boldsymbol{x}$. To measure necessity of a feature value $\boldsymbol{x}_j = a$ for the model output y^*, we would like to operationalize Equation 3. A simple way to do so is to use a method for generating counterfactual explanations, but restrict it such that only $\boldsymbol{x}_j$ can be changed. To implement it, we use Equation 5 from DiCE but restrict the set of modifiable variables to $\boldsymbol{x}_j$. The fraction of times that changing $\boldsymbol{x}_j$ leads to a valid counterfactual example indicates that the extent to which $\boldsymbol{x}_j = a$ is necessary for the current model output y^*.

$$\texttt{Necessity} = \frac{\sum_{i, \boldsymbol{x}_j \neq a} \mathbb{1}(CF_i)}{\mathrm{nCFs} * N} \tag{6}$$

where N is the total number of test instances for which nCF counterfactuals are generated each.

For the sufficiency condition from Equation 2, we adopt the reverse approach. Rather than changing $\boldsymbol{x}_j$, we fix it to its original value and let all other features vary their values, which can similarly be implemented with DiCE. If no unique valid counterfactual examples are generated, then it implies that $\boldsymbol{x}_j = a$ is sufficient for causing the model output y^*. If not, then (1- fraction of times that a counterfactual example is generated) tells us about the extent of sufficiency of $\boldsymbol{x}_j = a$. In practice, even when using all the features, we may not obtain 100% success in generating valid counterfactuals. Therefore, we modify the sufficiency metric to compare the fraction of CFs generated using all features to the fraction of CFs generated while keeping $\boldsymbol{x}_j$ constant.

$$\texttt{Sufficiency} = \frac{\sum_i \mathbb{1}(CF_i)}{\mathrm{nCFs} * N} - \frac{\sum_{i, \boldsymbol{x}_j \leftarrow a} \mathbb{1}(CF_i)}{\mathrm{nCFs} * N} \tag{7}$$

4.3 Feature Importance using counterfactuals

In addition to evaluating properties of attribution-based explainers, counterfactual explanations offer a natural way of generating feature attribution scores based on the extent to which a feature value is necessary for the outcome. The intuition comes from Equation 3: a feature that is changed more often when generating counterfactual examples must be an important feature. Below we describe the method $DiCE_{FA}$ to generate attribution scores from a set of counterfactual examples.

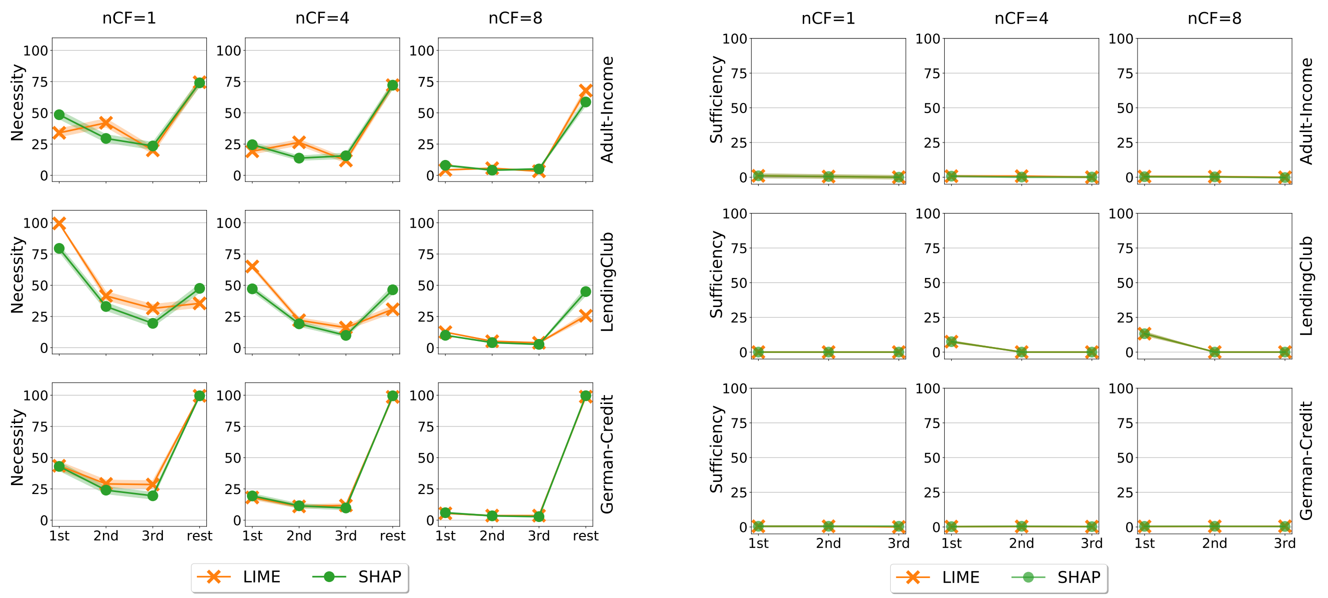
To explain the output $y^* = f(\boldsymbol{x})$, the $DiCE_{FA}$ algorithm proceeds by generating a diverse set of nCF counterfactual examples for the input $\boldsymbol{x}$, where nCF is the number of counterfactuals. A feature x_j that is important in changing a predicted outcome, is more likely to be changed frequently in nCF diverse CFs than a feature x_k that is less important. For each feature, therefore, the attribution score is the fraction of CF examples that have a modified value of the feature. To generate a local explanation, the attribution score is averaged over multiple values of nCF, typically going from 1 to 10. To obtain a global explanation, this attribution score can be averaged over many inputs for each feature.

4.4 Datasets and Implementation Details

We use three common datasets in explainable ML literature.

- `Adult-Income`. This dataset [18] is based on the 1994 Census database and contains information like Age, Gender, Martial Status, Race, Education Level, Occupation, Work Class and Weekly Work Hours. It is available online as part of the UCI machine learning repository. The task is to determine if the income of a person would be higher than $50,000 (1) or not (0). We process the dataset using techinques proposed by prior work [50] and obtain a total of 8 features.
- `LendingClub`. Lending Club is a peer-to-peer lending company, which helps in linking borrowers and investors. We use the data about the loans from LendingClub for the duration (2007-2011) and use techniques proposed works [7, 17, 45] for processing the data. We arrive at 8 features, with the task to classify the payment of the loan by a person (1) versus no payment of the loan (0).
- `German-Credit`. German Credit [1] consists of various features like Credit Amount, Credit History, Savings, etc regarding people who took loans from a bank. We utilize all the features present in the dataset for the task of credit risk prediction, whether a person has good credit risk (1) or bad credit risk (0).

Implementation Details. We trained ML models for different datasets in PyTorch and use the default parameters of LIME and DiCE in all our experiments unless specified otherwise. For SHAP, we used its KernelExplainer interface with median value of features as background dataset. We understand SHAP's KernelExplainer is slow with a large background dataset, so we used median instead. However, the choice of KernelExplainer and our background



(a) Necessity

(b) Sufficiency

Figure 1: The y-axis represents the *necessity* and *sufficiency* measures at a particular nCF, as defined in §4.2. Fig. 1a shows the results when we are only allowed to change the k-th most important features ($k = 1, 2, 3$) or the other features, while Fig. 1b shows the results when we fix the k-th most important features ($k = 1, 2, 3$) but are allowed to change other features. While necessity is generally aligned with feature ranking derived from LIME and SHAP, the most important features cannot lead to changes in the model output on their own. In almost all cases, "rest" achieves better success in producing CFs. For sufficiency, none of these top features are sufficient for preserving the original model output.

dataset setting can limit the strength of SHAP[1], and we leave further exploration of different configurations of SHAP to future work.

Note that DiCE's hyperparameters for proximity and diversity in CFs are important. For instance, the diversity term enforces that different features change their values in different counterfactuals. Otherwise we may obtain multiple duplicate counterfactual examples that change the same feature. Results in the main paper are based on the default hyperparameters in DiCE, but our results are robust to different choices of these hyperparameters (see Suppl. A.1).

5 EVALUATING NECESSITY & SUFFICIENCY

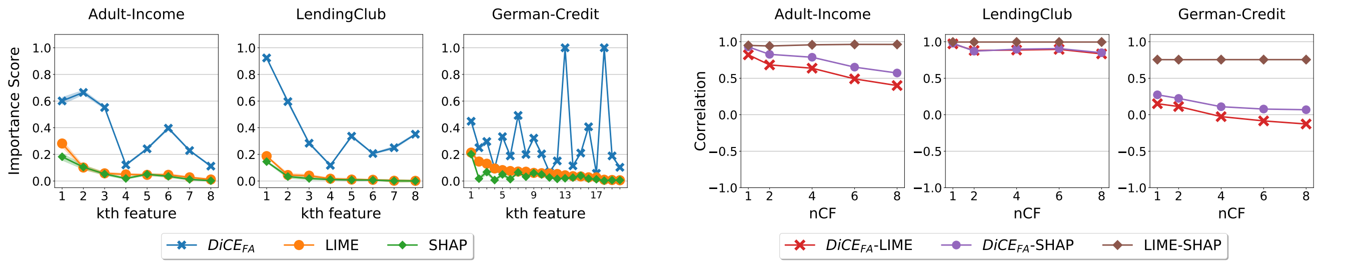
We start by examining the necessity and sufficiency of top features derived with feature attribution methods through counterfactual generation. Namely, we measure whether we can generate valid CFs by changing only the k-th most important feature (necessity) or changing other features except the k-th most important feature (sufficiency). Remember that necessity and sufficiency are defined with respect to the original output. For example, if changing a feature can vary the predicted outcome, then it means that this feature is necessary for the original prediction.

Are important features necessary? Given top features identified based on feature attribution methods (LIME and SHAP), we investigate whether we can change the prediction outcomes by using *only* the k-th most important feature, where $k \in \{1,2,3\}$, We choose small k since the number of features is small in these datasets. Specifically, we measure the average percentage of unique and valid counterfactuals generated using DiCE for 200 random test instances by fixing other features and allowing DiCE to only change the k-th most important feature. This analysis helps us understand if the top features from LIME or SHAP are *necessary* to produce the current model output. Fig. 1a shows the results for different datasets when asked to generate different numbers of CFs. While we produced CFs for nCFs $\in$ 1,2,4,6,8, we show results for 1, 4, and 8 only for brevity. To provide a benchmark, we also consider the case where we use all the other features that are not in the top three.

Our results in Fig. 1a suggest that the top features are mostly unnecessary for the original prediction: changing them does not alternate the predicted outcome. For instance, in `German-Credit`, none of the top features have a validity of above 50%, in fact often below 20%. In comparison, features outside the top three can always achieve almost 100%. This is likely related to the fact that there are 20 features in `German-Credit`, but it still highlights the limited utility in explanation by focusing on the top features from feature attribution methods. Similar results also show up in `Adult-Income`, but not as salient as in `German-Credit`.

A relatively more necessary case is the top feature for `LendingClub`. Upon investigation, we find this dataset has a categorical feature *grade* of seven levels, which is assigned by the lending company as an indicator of loan repayment. The loan grade is designed based

[1] See issues 391 and 451 on SHAP's GitHub repository: https://github.com/slundberg/shap/issues



(a) Average feature importance scores (nCFs=4).

(b) Correlation between feature importance scores.

Figure 2: In Fig. 2a, feature indexes in x-axis are based on ranking from LIME. SHAP mostly agrees with LIME, but less important features based on LIME can have high feature importance based on $DiCE_{FA}$. Fig. 2b directly compares feature importance score from different methods: LIME and SHAP are more similar to each other than to $DiCE_{FA}$.

on a combination of factors including credit score. Since the quality of loan grade is highly correlated with loan repayment status, both LIME and SHAP give high importance score to this feature for most test instances – they assign highest score for 98% and 73% of the test instances respectively. As a result, changing LIME's top-1 feature is enough to get almost perfect unique valid CFs when generating one counterfactual. However, the necessity of a single feature quickly reduces as we generate more CFs. Even in this dataset where there is a dominant feature, the other features are more necessary than the top feature (grade) when $nCF = 8$.

Despite the limited nature, necessity is generally aligned with the feature ranking from LIME and SHAP: the higher the feature importance score, the greater the necessity. The only exception is the second most important feature in adult based on LIME. For most instances, this feature is a person's education level. We repeat the above analysis by allowing *all* features upto top-k to be changed (Suppl. A.3) and find that necessity of the top-k subset increases, but is still less than 1 for nCF> 1. That is, changing all top-3 ranked features is also not enough to generate counterfactuals for all input examples, especially for higher-dimensional `German-Credit`.

Are important features sufficient? Similar to necessity, we measure the sufficiency of top features from attribution-based methods by fixing the k-th most important features and allowing DiCE to change the other features. If the k-th most important feature is sufficient for the original prediction, we would expect a low success rate in generating valid CFs with the other features.

Fig. 1b shows the opposite. We find that the validity is 100% till $nCF = 8$ even *without* changing the k-th most important feature based on LIME or SHAP in `Adult-Income` and `German-Credit`. In comparison, for `LendingClub`, while no change in the top-2 or top-3 does not affect the perfect validity, however, no change in the most important feature does decrease the validity when generating more than one CFs. This result again highlights the dominance of grade in `LendingClub`. However, even in this case, the sufficiency metric is still below 20%. We also repeat the above analysis by fixing *all* the top-k features and get identical results (see Suppl. A.3).

Given the low sufficiency and necessity of top features from LIME and SHAP, we examine how often these important features get changed by DiCE if we do not fix any features, in Suppl. A.2.

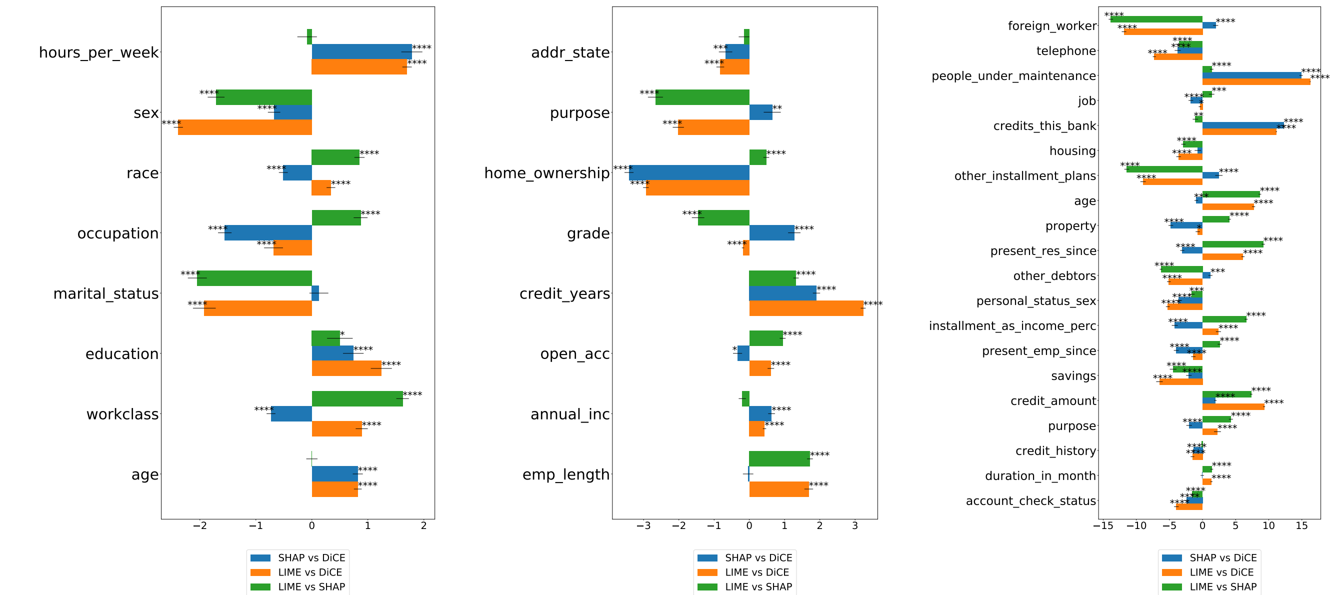
6 FEATURE IMPORTANCE BY CF EXAMPLES

As discussed in §4, we can obtain feature importance score from DiCE based on how often a feature is changed in the generated CFs. In this section, we compare the feature importance score from $DiCE_{FA}$ to that from LIME and SHAP, and investigate how it can provide additional, complementary information about a ML model.

Correlation with LIME or SHAP feature importance. We start by examining how the importance scores from different methods vary for different features and datasets. Fig. 2a shows the average feature importance score across 200 random test instances when nCFs = 4. For LIME and SHAP, we take the absolute value of feature importance score to indicate contribution. LIME and SHAP agree very well on for `Adult-Income` and `LendingClub`. While they mostly agree in `German-Credit`, there are some bumps indicating disagreements. In comparison, $DiCE_{FA}$ is less similar to LIME than SHAP. This is especially salient in `German-Credit`. The features that are ranked 13th and 18th by LIME– the number of existing credits a person holds at the bank and the number of people being liable to provide maintenance for – are the top two important features by $DiCE_{FA}$'s scores. They are ranked first and second, respectively, by $DiCE_{FA}$ in 98% of the test instances.

We then compute the Pearson correlation between these average feature importance scores derived with different explanation methods in Figure 2b for different nCFs. We find that LIME and SHAP agree on the feature importance scores on average for all the three datasets, similar to what was observed in Figure 2a at nCFs = 4. The correlation is especially strong for `Adult-Income` and `LendingClub` datasets each of which have only 8 features.

Comparing $DiCE_{FA}$ and the two feature attribution methods, we find that they are well correlated in `LendingClub`. This, again, can be attributed to the dominance of *grade*. All methods choose to consider grade as an important feature. In `Adult-Income`, the correlation of $DiCE_{FA}$ with SHAP and LIME decreases as the number of counterfactuals (*nCF*) increases. This is not surprising since at higher *nCF*, DiCE changes a diverse set of features of different importance levels (according to LIME or SHAP) to get counterfactuals. For instance, in Figure 2a at $nCF = 4$, the feature that is ranked sixth on average by LIME, *hours-per-week*, is changed by DiCE almost twice than another feature, *sex*, which is ranked fourth on average. Hence, we can expect that the average frequency of changing the



(a) Adult Dataset **(b) Lending Club Dataset** **(c) German Credit Dataset**

Figure 3: Ranking correlation of different explanation methods on all the datasets. The x-axis denotes the mean difference in the rankings for each feature over all the test inputs. Stars denote significance of the mean difference by using p-values.

most important feature would decrease with increasing *nCF* and less important features would start to vary more (see §5). By highlighting the less-important features as per LIME or SHAP, $DiCE_{FA}$ focuses on finding different subsets of *necessary* features that can change the model output. In comparison, LIME and SHAP tend to prefer *sufficiency* of features in contributing to tht original model output.

Further, the correlation between $DiCE_{FA}$ and LIME or SHAP is very low in `German-Credit`: it is below 0.25 for all values of nCFs and can also be negative as the number of counterfactuals increases. We hypothesize that this is due to the number of features. `German-Credit` has 20 features and in general with increasing feature set size, we find that DiCE is able to generate CFs even with less important features of LIME or SHAP. In `German-Credit`, LIME and SHAP also agree less compared to other datasets. These results suggest that datasets with few features such as `Adult-Income` and `LendingClub` may provide limited insights into understanding explanation methods in practice, especially as many real-world datasets tend to have tens of features.

Differences in feature ranking. Feature importance scores can be difficult to compare and interpret, therefore many visualization tools show the ranking of features based on importance. Fig. 3 shows the mean difference in the rankings induced by feature importance scores from different explanation methods for each feature, computed over 200 test inputs. We also perform paired *t*-tests to test if there is a significant difference between rankings from different methods for the same feature. This analysis allows us to see the local differences in feature rankings beyond average feature importance score.

For most features across all datasets, we find that the feature rankings on individual inputs can be significantly different. This is true even when comparing LIME and SHAP (green bars), which otherwise show high positive correlation in average (global) feature importance score. For instance, in `Adult-Income`, LIME consistently ranks marital status and sex higher than SHAP, while SHAP tend to rank work class, race, and occupation higher. Interestingly, they tend to agree on the ranking of continuous features, i.e., hours per week and age. As expected, LIME and DiCE provide different rankings for all features, while SHAP and DiCE differs in all except marital status. We see similar, large difference in feature rankings for `German-Credit` and `LendingClub` datasets.

While we expected significant differences between $DiCE_{FA}$ and the two attribution-based methods based on global feature importance scores from Fig. 2, we find significant differences between LIME and SHAP too on individual inputs. In general, our results demonstrate the difficulty in building a single, ideal explanation method. On the one hand, explanations capture different theoretical notions such as necessity and sufficiency, which is why $DiCE_{FA}$ disagrees in its ranking on almost all features with LIME and SHAP. On the other hand, even for explanation methods that capture the same notion (sufficiency), their specific implementation of the sufficiency notion leads to varying feature rankings in practice.

7 CASE STUDY: HOSPITAL ADMISSION

To understand the complementarities between different explanation methods on realistic datasets beyond the toy settings of `Adult-Income` and `LendingClub`, we now present a case study using a real-world hospital admission prediction problem. Predicting patients who are likely to get admitted during emergency visits helps hospitals to better allocate their resources, provide appropriate medical interventions, and improving patient treatment rates [4, 8, 11, 13, 16, 23, 29, 31]. Given the importance of the decision, it is critical that the predictions from an ML model be explainable to doctors in the emergency department. We leverage the dataset and models by Hong et al. [15] who uses a variety of ML models including XGBoost and deep neural networks to predict hospital admission at emergency department (ED) from "triage" and demographic information, and other data collected during previous ED visits.

Data and model training. We use the ML model based on triage features, demographic features and chief complaints information from Hong et al. [15]. Triage features consist of 13 variables to indicate the severity of ailments when a patient arrives at the ED. This model also uses demographics, including including race, gender, and religion, and 200 binary features indicating the presence of various chief complaints. As a result, this dataset has many more features than `Adult-Income`, `LendingClub`, and `German-Credit`. We refer to this dataset as `HospitalTriage`. We reproduce the deep neural network used by Hong et al. which has two hidden layers with 300 and 100 neurons respectively. The model achieves a precision and recall of 0.81 each and an AUC of 0.87 on the test set. Further, we used a 50% sample of the original data, consisting of 252K data points, for model training as the authors show that the accuracy saturates beyond this point. We sample 200 instances from the test set over which we evaluate the attribution methods.

In-depth look at the feature ranking. We start with the feature ranking produced by different methods to help familiarize with this real-world dataset. We then replicate the experiments in §5 and §6.

We rank the features of `HospitalTriage` based on $DiCE_{FA}$, LIME, and SHAP using the same method as in §6. Fig. 4 shows the distribution of mean rankings of different types of features in `HospitalTriage` according to our feature attribution methods[2]. This dataset has three category of features — demographics, triage and chief complaints. We find that SHAP ranks binary chief-complaints features much lower on average than $DiCE_{FA}$ and LIME. Though $DiCE_{FA}$ and LIME disagree on demographics and triage features rankings, they both have similar mean rankings on chief-complaints features which constitutes 90% of the features. Hence, $DiCE_{FA}$ and LIME has a relatively higher correlation (see Fig. 6b) compared to any other methods.

Furthermore, $DiCE_{FA}$ considers demographics and triage features more important as compared to the chief-complaints features, since the former features have smaller rank (<80) on average. In contrast, LIME assigns them a larger rank. This has implications in fairness: when the ML model is evaluated based on LIME alone, the model would be seen as fair since chief-complaints features contribute more to the prediction on average. However, $DiCE_{FA}$ shows that demographic features can also be changed to alter a prediction, raising questions about making decisions based on sensitive features. Indeed, the authors of the study present a low-dimensional XGBoost model[3] by identifying features using information gain as metric[15]. They find that 5 out of 9 demographic details – insurance status, marital status, employment status, race, and gender, and 6 out of 13 triage features are identified as important in their refined model. On the other hand, only 8 out of 200 chief-complaints features are found important.

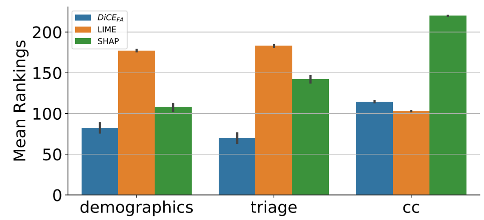


Figure 4: Mean rankings of different types of feature groups by $DiCE_{FA}$, LIME, and SHAP. The lower the ranking, the more important the features.

Necessity and sufficiency. Next, we replicate the experiments from §5 for `HospitalTriage` to understand the necessity and sufficiency of the important features of LIME and SHAP in generating CFs. The trend for SHAP in Fig. 5 is similar to what was observed in Fig. 1a— changing the more important features is more likely to generate valid CFs (the orange line decreases as *nCF* increases). However, in the case of LIME, we observe that the third important feature leads to more CFs, almost double than that of the first or second feature only. The reason is that in around 26% of the test instances, LIME rates Emergency Severity Index (ESI) as the third most important feature. ESI is a categorical feature indicating the level of severity assigned by the triage nurse [15]. $DiCE_{FA}$ considers this feature important to change the outcome prediction and ranks it among the top-10 features for more than 60% of the test

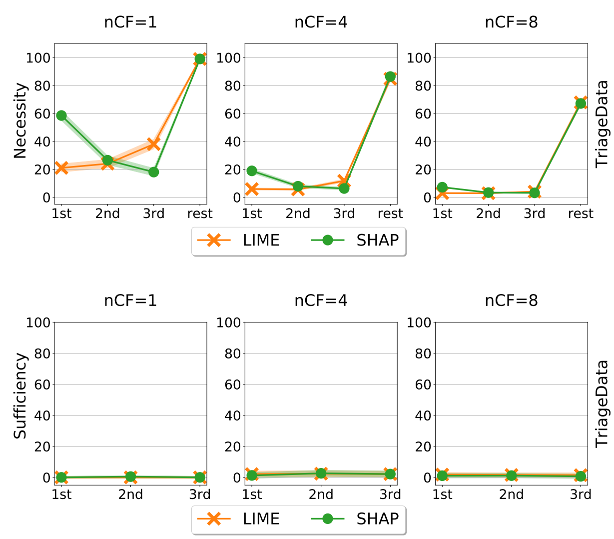
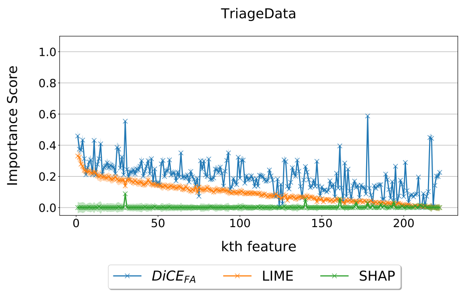


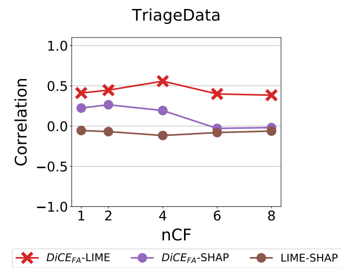
Figure 5: *Necessity* and *Sufficiency* measures at a particular k, as defined in §4.2 for the `HospitalTriage` data.

[2] We assign features the maximum of the ranks when there is a tie. $DiCE_{FA}$'s and LIME's rankings are invariant to the treatment of ties whereas SHAP's is. We choose the maximum to better distinguish different methods' rankings

[3] The authors of the `HospitalTriage` study built this refined model using all features they collected while we use the one type of data they refer to as triage features.



(a) Average feature importance scores (nCFs=4).



(b) Correlation between feature importance scores.

Figure 6: In Fig. 6a, feature indexes in x-axis are based on ranking from LIME. SHAP presents very different outcomes from LIME, and the feature importance shows much smaller variation than $DiCE_{FA}$. Fig. 6b directly compares feature importance score from different methods: the correlation between LIME and SHAP is much weaker than in Fig. 2b.

instances. ESI is also one of the top-3 features by the information gain metric in the refined XGBoost model from Hong et al. [15].

The sufficiency results (Fig. 5) are similar to Fig. 1b. Any of the top-3 features are not necessary for generating CFs. At nCFs = 1, the same number of valid counterfactuals (100%) can be generated while keeping the 1st, 2nd or the 3rd feature fixed, as when changing all features. Similarly, the same number of valid counterfactuals (68%) can be generated at nCFs = 8, irrespective of whether the top-k features are changed or not. Note that the overall fraction of valid counterfactuals generated decreases as nCFs increase, indicating that it is harder to generate diverse counterfactuals for this dataset. We expect the lack of sufficiency of top-ranked features to hold in many datasets, as the number of features increases.

Similarity between feature importance from different methods. Fig. 6b shows the correlation of feature importance score derived from different methods. Different from what was observed for other datasets in Fig. 2b, LIME and SHAP have almost zero correlation between the feature rankings in `HospitalTriage`. This observation resonates with prior work demonstrating the instability and lack of robustness of these feature attribution methods, i.e., they can significantly differ when used to explain complex nonlinear models with high dimensional data [3, 21, 41, 49]. In the case of `HospitalTriage`, the importance scores given by LIME and SHAP are indeed very different for most of the features. For instance, SHAP assigns close to zero weights for many binary "chief-complaint" features of `HospitalTriage` data in most of the test instances, while LIME assigns diverse importance scores. For instance, Figure 6a shows the absolute feature attribution scores of different methods at $nCF = 4$ and it can be observed that SHAP's scores are close to zero, on average, for most of the features. Indeed, we find that the average entropy of the importance scores of LIME is 3.2 points higher than that of SHAP on average. On the other other hand, the differences for `LendingClub`, `Adult-Income`, and `German-Credit` were only 0.37, 0.48, and 0.84 respectively.

In addition, while $DiCE_{FA}$ agrees more with SHAP than with LIME for other datasets (except `LendingClub` where all methods agreed due to a dominating feature), here we obtain the reverse trend. $DiCE_{FA}$ has relatively weaker correlation with SHAP in the case of `HospitalTriage`, echoing the difference observed for chief complaints in Fig. 4. In particular, at nCFs = 6 and nCFs = 8, they both have no correlation on average feature rankings. At higher nCF, DiCE varies more number of binary features most of which are assigned very low weights by SHAP and hence the disagreement.

To summarize, we show how analyzing the feature attribution methods on a real-world problem highlights the complementarity and the differences in these methods. First, the highest ranked features by attribution-based methods like LIME are not sufficient, and are not always the most necessary for causing the original model output; more valid counterfactuals can be generated by varying a feature with larger rank compared to those with smaller rank. Second, there are substantial differences in feature importance scores from the different methods, to the extent that they can completely change the interpretation of a model wrt. properties like fairness. Unlike the previous low-dimensional datasets, even LIME and SHAP demonstrate substantial differences in global feature importance scores. $DiCE_{FA}$ rankings somehow strike a balance between the two methods in importance: $DiCE_{FA}$ agrees with SHAP on demographics features and with LIME on chief complaint features. Finally, similar to what we have observed before, $DiCE_{FA}$ distributes feature importance more equally, especially for the features with larger rank from LIME and SHAP.

8 CONCLUDING DISCUSSION

Our work represents the first empirical attempt to unify explanation methods based on feature attribution and counterfactual generation. We provide a framework based on actual causes to interpret these two approaches. Through an empirical investigation on a variety of datasets, we demonstrate intriguing similarities and differences between these methods. Our results show that it is not enough to focus on only the top features identified by feature attribution methods such as LIME and SHAP. They are neither sufficient nor necessary. Other features are (sometimes more) meaningful and can potentially provide actionable changes.

We also find significant differences in feature importance induced from different explanation methods. While feature importance induced from DiCE can be highly correlated with LIME and SHAP on low-dimensional datasets such as `Adult-Income` and `LendingClub`, they become more different as the feature dimension grows. Even in German-Credit with 20 features, they can show no or even negative correlation when generating multiple counterfactuals.

Our study highlights the importance of using different explanation methods and of future work to find which explanation methods are more appropriate for a given question. There can be many valid questions that motivate a user to look for explanations [24]. Even for the specific question of which features are important, the definition of importance can still vary, for example, actual causes vs. but-for causes. It is important for our research community to avoid the one-size-fits-all temptation that there exists a uniquely best way to explain a model. Overall, while it is a significant challenge to leverage the complementarity of different explanation methods, we believe that the existence of different explanation methods provides exciting opportunities for combining these explanations.

REFERENCES

[1] Accessed 2019. UCI Machine Learning Repository. German credit dataset. https://archive.ics.uci.edu/ml/support/statlog+(german+credit+data)

[2] Kjersti Aas, Martin Jullum, and Anders Løland. 2019. Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *arXiv preprint arXiv:1903.10464* (2019).

[3] David Alvarez-Melis and Tommi S Jaakkola. 2018. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049* (2018).

[4] Rajiv Arya, Grant Wei, Jonathan V McCoy, Jody Crane, Pamela Ohman-Strickland, and Robert M Eisenstein. 2013. Decreasing length of stay in the emergency department with a split emergency severity index 3 patient flow model. *Academic Emergency Medicine* 20, 11 (2013), 1171–1179.

[5] Solon Barocas, Andrew D Selbst, and Manish Raghavan. 2020. The hidden assumptions behind counterfactual explanations and principal reasons. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 80–89.

[6] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of KDD*.

[7] Kevin Davenport. 2015. Lending Club Data Analysis Revisited with Python. https://kldavenport.com/lending-club-data-analysis-revisted-with-python/

[8] Thomas Desautels, Jacob Calvert, Jana Hoffman, Melissa Jay, Yaniv Kerem, Lisa Shieh, David Shimabukuro, Uli Chettipally, Mitchell D Feldman, Chris Barton, et al. 2016. Prediction of sepsis in the intensive care unit with minimal electronic health record data: a machine learning approach. *JMIR medical informatics* 4, 3 (2016), e28.

[9] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2019. ERASER: A Benchmark to Evaluate Rationalized NLP Models. In *Proceedings of ACL*.

[10] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems*. 592–603.

[11] Andrea Freyer Dugas, Thomas D Kirsch, Matthew Toerper, Fred Korley, Gayane Yenokyan, Daniel France, David Hager, and Scott Levin. 2016. An electronic emergency triage system to improve patient distribution by critical outcomes. *The Journal of emergency medicine* 50, 6 (2016), 910–918.

[12] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820* (2018).

[13] Julian S Haimovich, Arjun K Venkatesh, Abbas Shojaee, Andreas Coppi, Frederick Warner, Shu-Xia Li, and Harlan M Krumholz. 2017. Discovery of temporal and disease association patterns in condition-specific hospital utilization rates. *PloS one* 12, 3 (2017), e0172049.

[14] Joseph Y Halpern. 2016. *Actual causality.* MiT Press.

[15] Woo Suk Hong, Adrian Daniel Haimovich, and R Andrew Taylor. 2018. Predicting hospital admission at emergency department triage using machine learning. *PloS one* 13, 7 (2018), e0201016.

[16] Steven Horng, David A Sontag, Yoni Halpern, Yacine Jernite, Nathan I Shapiro, and Larry A Nathanson. 2017. Creating an automated trigger for sepsis clinical decision support at emergency department triage using machine learning. *PloS one* 12, 4 (2017), e0174708.

[17] JFdarre. 2015. Project 1: Lending Club's data. https://rpubs.com/jfdarre/119147

[18] Ronny Kohavi and Barry Becker. 1996. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/adult

[19] Alex Kulesza and Ben Taskar. 2012. Determinantal point processes for machine learning. *arXiv preprint arXiv:1207.6083* (2012).

[20] I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. 2020. Problems with Shapley-value-based explanations as feature importance measures. *arXiv preprint arXiv:2002.11097* (2020).

[21] Vivian Lai, Jon Z Cai, and Chenhao Tan. 2019. Many Faces of Feature Importance: Comparing Built-in and Post-hoc Feature Importance in Text Classification. *arXiv preprint arXiv:1910.08534* (2019).

[22] Himabindu Lakkaraju, Stephen H Bach, and Jure Leskovec. 2016. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1675–1684.

[23] Scott Levin, Matthew Toerper, Eric Hamrock, Jeremiah S Hinson, Sean Barnes, Heather Gardner, Andrea Dugas, Bob Linton, Tom Kirsch, and Gabor Kelen. 2018. Machine-learning-based electronic triage more accurately differentiates patients with respect to clinical outcomes compared with the emergency severity index. *Annals of emergency medicine* 71, 5 (2018), 565–574.

[24] Q Vera Liao, Daniel Gruen, and Sarah Miller. 2020. Questioning the AI: Informing Design Practices for Explainable AI User Experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–15.

[25] Zachary C Lipton. 2018. The mythos of model interpretability. *Queue* 16, 3 (2018), 31–57.

[26] Yin Lou, Rich Caruana, and Johannes Gehrke. 2012. Intelligible models for classification and regression. In *Proceedings of KDD*.

[27] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*. 4765–4774.

[28] Tim Miller. 2018. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* (2018).

[29] Karel GM Moons, Andre Pascal Kengne, Mark Woodward, Patrick Royston, Yvonne Vergouwe, Douglas G Altman, and Diederick E Grobbee. 2012. Risk prediction models: I. Development, internal validation, and assessing the incremental value of a new (bio) marker. *Heart* 98, 9 (2012), 683–690.

[30] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 607–617.

[31] Ziad Obermeyer and Ezekiel J Emanuel. 2016. Predicting the future—big data, machine learning, and clinical medicine. *The New England journal of medicine* 375, 13 (2016), 1216.

[32] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer.* John Wiley & Sons.

[33] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. FACE: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 344–350.

[34] 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.

[35] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations.. In *AAAI*, Vol. 18. 1527–1535.

[36] Donald B Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *J. Amer. Statist. Assoc.* 100, 469 (2005), 322–331.

[37] 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.

[38] Chris Russell. 2019. Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 20–28.

[39] Shubham Sharma, Jette Henderson, and Joydeep Ghosh. 2019. Certifai: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models. *arXiv preprint arXiv:1905.07857* (2019).

[40] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685* (2017).

[41] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2019. How can we fool LIME and SHAP? Adversarial Attacks on Post hoc Explanation Methods. *arXiv preprint arXiv:1911.02508* (2019).

[42] Kacper Sokol and Peter Flach. 2020. Explainability fact sheets: a framework for systematic assessment of explainable approaches. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 56–67.

[43] Mukund Sundararajan and Amir Najmi. 2019. The many Shapley values for model explanation. *arXiv preprint arXiv:1908.08474* (2019).

[44] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365* (2017).

[45] Sarah Tan, Rich Caruana, Giles Hooker, and Yin Lou. 2017. Detecting bias in black-box models using transparent model distillation. *arXiv preprint arXiv:1710.06169* (2017).

[46] Berk Ustun, Alexander Spangher, and Yang Liu. 2019. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 10–19.

[47] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.* 31 (2017), 841.
[48] Mo Yu, Shiyu Chang, Yang Zhang, and Tommi S Jaakkola. 2019. Rethinking cooperative rationalization: Introspective extraction and complement control. *arXiv preprint arXiv:1910.13294* (2019).
[49] 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).
[50] Haojun Zhu. 2016. Predicting Earning Potential using the Adult Dataset. https://rpubs.com/H_Zhu/235617

A SUPPLEMENTARY MATERIALS

A.1 Validity and Stability of DiCE

Table 1 shows the mean percentage validity of DiCE with its default hyperparameters. DiCE has two main hyperparamaters, namely *proximity_weight* and *diversity_weight*, controlling the closeness of counterfactuals to the test instance and the diversity of counterfactuals respectively. *proximity_weight* takes 0.5 and *diversity_weight* takes 1.0 as the default values respectively. These two parameters have an inherent trade-off [30] and hence we change only the *diversity_weight* to examine the sensitivity of hyperparameters to the feature importance scores derived from $DiCE_{FA}$. Figure 7 shows the results. We find that $DiCE_{FA}$ is not sensitive to these hyperparameters and different hyperparameter versions have a correlation of above 0.96 on all datasets.

Data	avg %valid CFs	#instances
`Adult-Income`	[96,99,98,98,98]	[192,196,188,184,185]
`German-Credit`	[100,100,100,100,100]	[199,198,199,199,198]
`LendingClub`	[100,100,100,100,100]	[200,200,200,200,200]
`HospitalTriage`	[99,92,86,72,68]	[198,187,134,65,53]

Table 1: The second column shows the mean percentage of unique and valid CFs found at each $nCF \in 1,2,4,6,8$ for different datasets given in the first column. The mean validity is computed over a random sample of 200 test instances for each dataset. The third column shows the number of test instances for which all the CFs found are unique and valid at different nCF.

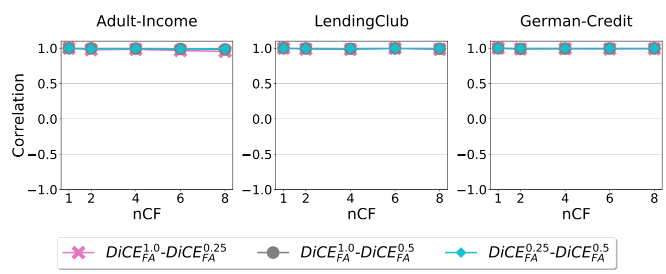


Figure 7: Correlation between different versions of $DiCE_{FA}$ at different hyperparameters. The pink line corresponds to correlation between feature importance derived from DiCE with 1.0 and 0.25 *diversity-weight*. Similarly, the gray and blue lines correspond to 1.0 and 0.5, and 0.25 and 0.5 *diversity-weights* respectively. All $DiCE_{FA}$ methods exhibit high pairwise correlation (> 0.96) on all datasets.

A.2 How are locally important features changed in CFs?

Given the low sufficiency and comprehensiveness of top features from LIME and SHAP, we examine how often these important features get changed by DiCE if we do not fix any features. Specifically, for each test instance, we have local rankings from different feature

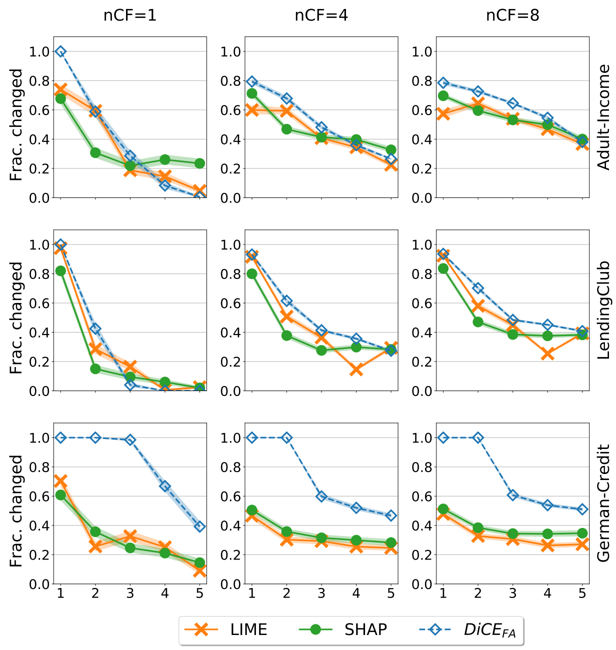


Figure 8: The y-axis shows the fraction of *nCF* counterfactuals in which a feature is changed compared to the original input, averaged across 200 random test instances. x-axis is based on the ranking of different feature attribution methods, which varies from instance to instance. In other words, it shows how frequently the k-th most important feature for a given instance is changed to generate counterfactuals for this instance.

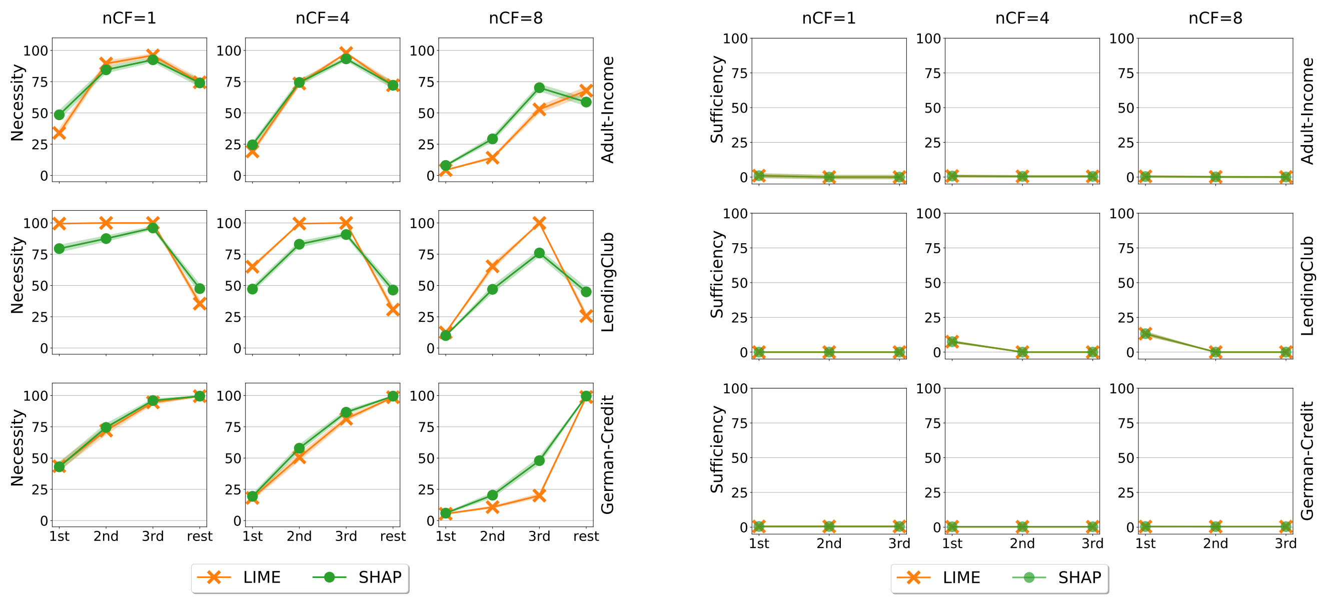
attribution methods and examine what fraction of top-5 features from each method is varied in $\boldsymbol{x}$ to get counterfactuals at different *nCF*. Further, we use the feature rankings based on $DiCE_{FA}$'s feature importance scores, N^{f_i}, as a comparison point. Note that the rankings of LIME, SHAP, and $DiCE_{FA}$ vary from instance to instance, and the fraction of changes based on the $DiCE_{FA}$ ranking can be seen as an oracle towards getting counterfactuals with DiCE.

Fig. 8 shows the comparison between attribution methods for our three datasets at different *nCF*. First, the more important features based on LIME or SHAP tend to be changed more frequently by DiCE to generate CFs, suggesting that the important features from LIME and SHAP reasonate with DiCE as well. The frequency of change in the most important features (the top-1 especially) of LIME/SHAP are very close to that of $DiCE_{FA}$ for `LendingClub` while it is much lower with `German-Credit`. Corroborating our previous inference, we note that in the presence of a dominating feature, like *grade* in `LendingClub`, different attribution methods tend to agree well with each other. However, in the case of `German-Credit` where there are relatively many features, DiCE generates more CFs with features that lead to a different prediction class than those that contribute more only to $\boldsymbol{x}$'s prediction but does not help in changing the class. This result confirms the theoretical understanding from §3 that DiCE optimizes for the necessity of a feature value (α_{CF}) whereas LIME and SHAP optimize for the sufficiency of a feature value (β) in causing the model output.

Second, the frequency of change for less important features increases as *nCF* increases for almost all the datasets considered. The plots corresponding to LIME and SHAP becomes almost flat for `German-Credit` at higher *nCF*s. This indicates that DiCE is able to generate unique valid CFs even with features that are not in the top 5 based on LIME and SHAP, reinforcing our observation made from Fig. 1b. Further, an important point to note here is that for `Adult-Income`, while the most important feature of $DiCE_{FA}$ is always changed when $nCF = 1$, it is changed only 0.8 times on average at other *nCF*s, unlike in other datasets. This observation indicates that there are more than one features that are equally important in changing the predicted outcome.

A.3 Necessity and Sufficiency

Figure 9 shows the necessity and sufficiency metrics when we allow *all* features upto top-k features to change (for necessity) or remain fixed (sufficiency). Necessity increases for the top-k features, but sufficiency remains identical to the setting in the main paper.



(a) Necessity **(b) Sufficiency**

Figure 9: The y-axis represents the *necessity* and *sufficiency* measures at a particular nCF, as defined in §4.2. Fig. (a) shows the results when we are only allowed to change *until* the k-th most important features ($k = 1, 2, 3$) or the other features, while Fig. (b) shows the results when we fix the *until* k-th most important features ($k = 1, 2, 3$) but are allowed to change other features.