

DisCERN: Discovering Counterfactual Explanations using Relevance Features from Neighbourhoods

Nirmalie Wiratunga, Anjana Wijekoon, Ikechukwu Nkisi-Orji, Kyle Martin, Chamath Palihawadana, David Corsar

School of Computing
Robert Gordon University
Aberdeen, Scotland

{n.wiratunga, a.wijekoon1, i.nkisi-orji, k.martin3, c.palihawadana, d.corsar1}@rgu.ac.uk

Abstract—Counterfactual explanations focus on “actionable knowledge” to help end-users understand how a machine learning outcome could be changed to a more desirable outcome. For this purpose a counterfactual explainer needs to discover input dependencies that relate to outcome changes. Identifying the minimum subset of feature changes needed to action an output change in the decision is an interesting challenge for counterfactual explainers. The DisCERN algorithm introduced in this paper is a case-based counter-factual explainer. Here counterfactuals are formed by replacing feature values from a nearest unlike neighbour (NUN) until an actionable change is observed. We show how widely adopted feature relevance-based explainers (i.e. LIME, SHAP), can inform DisCERN to identify the minimum subset of “actionable features”. We demonstrate our DisCERN algorithm on five datasets in a comparative study with the widely used optimisation-based counterfactual approach DiCE. Our results demonstrate that DisCERN is an effective strategy to minimise actionable changes necessary to create good counterfactual explanations.

Index Terms—Explainable AI, Counterfactuals, Case-based Reasoning

I. INTRODUCTION

Understanding a user’s explanation need is central to a system’s capability of provisioning an explanation which satisfies that need [1]. Typically an explanation generated by an AI system is considered to convey the internal state or workings of an algorithm that resulted in the system’s decision [2]. In machine learning (ML) the decision tends to be a discrete label or class (or in the case of regression tasks a numeric value). Although explanations focused on the internal state or logic of the algorithm is helpful to ML researchers it is arguably less useful to an end-user who may be more interested in how their current circumstances could be changed to receive a desired (better) outcome in the future. This calls for explainable AI (XAI) methods that focus on discovering relationships between the input dependencies that led to the system’s decision.

Case-based reasoning (CBR) is a widely accepted methodology for problem-solving, where the use of “similar problems to solve similar solutions” promotes an inherently interpretable reasoning strategy [3]. The neighbourhood of similar problems

is commonly harnessed in explainable AI [4]. A Nearest-Like Neighbours (NLNs) based explainer, focuses on input dependencies by identifying similarity relationships between the current problem and the retrieved nearest neighbour [5]. Research has shown that when similarity computation is focused on feature selection [6] and weighting [7], it can significantly improve retrieval of NLNs. The prevailing CBR approach to counterfactual generation harnesses similarity in neighbourhoods to identify similar cases with different class labels to the NLNs. Such neighbours, referred to as nearest unlike neighbours (NUNs), forms the basis for generating counterfactuals from neighbourhoods. Accordingly it would be reasonable to expect that NUN-based explanation generation would also benefit from the knowledge of feature importance. Certainly having to focus on a few key “actionable” features in domains with large numbers of features will be more desirable from a practical standpoint, as well as reducing the cognitive burden of understanding neighbourhood-based explanations.

Unlike NLN based explanations, a counterfactual explanation focuses on identifying “actionable knowledge”; which is knowledge about important causal dependencies between the input and the ML decision. Such knowledge helps to understand what could be changed in the input to achieve a preferred (desired) decision outcome. Typically a NUN is used to identify the number of differences between the input and its neighbour, that when changed can lead to a change in the system’s decision [8]. A key challenge that we address in this paper is to identify the minimum subset of feature value changes to achieve that needed decision flip - the “actionable features”. Accordingly the paper has the following contributions:

- Discover the minimum actionable feature changes using feature relevance-based explainer methods like LIME [9] or SHAP [10];
- Advances the case-based counterfactual generation research by introducing our DisCERN algorithm; and
- Evidences the utility of DisCERN with results from a comparative study with multiple datasets and the optimisation-based DiCE [11] counterfactual approach.

The rest of the paper is organised as follows. Section II investigates the importance of counterfactual XAI and dis-

cuss two key counterfactual methods and their evaluation methodologies. Feature Relevance Explainers are discussed and compared in Section III and Section IV presents our DisCERN algorithm to improve the discovery of actionable features in counterfactuals. Section V presents the evaluation methodologies, datasets and performance metrics followed by results in Section VI. Finally we draw conclusions and discuss future work in Section VII.

II. RELATED WORK

Like many explanation methods, counterfactual explanations are rooted within the study of human psychology. Counterfactual thinking is a mental exercise where we attempt to isolate the specific actions which contributed to a (usually undesirable) outcome, with the goal of identifying how we could alter these facts to reach an alternative (and often more desirable) outcome [12]. In this manner we derive a form of causal explanation for the outcome that was actually achieved, allowing us to reason about how a different outcome could be achieved in future [13]. To clarify, consider the following fictitious example of a runner who was placed third in a race: “I won the bronze medal for my race (actual outcome), but I would have won the gold medal (desired outcome) if I hadn’t tripped (causal action which changed outcome)”. Through this thought process, the runner has derived what they believe to be a causal action that led to receiving the bronze medal. With that knowledge inferred, the runner can then reason that in order to achieve a better outcome, they should run a similar race again, but avoid tripping. Likewise with a counterfactual explanation we aim to explain why the system generated a specific outcome instead of another, by inferring important relationships (causal or otherwise) between input features [2].

Case-based Reasoning (CBR) [8] and optimisation techniques [11], [14] have been the pillars of discovering counterfactuals from data. Recent work in CBR has shown how counterfactual case generation can be conveniently supported through the case adaptation stage, where query-retrieval pairs of successful counterfactual explanation experiences are used to create an explanation case-base [8]. Unlike the CBR approach to counterfactual generation, DiCE [11] trains a generative model using gradient descent optimisation to output multiple perturbed diverse counterfactuals. This optimisation upholds two key requirements of a good counterfactual which are: maximising the probability of obtaining the desired class (i.e. discovered counterfactual class is different from query class); and minimising the distance to the query (similar to discovering NUN). Additionally the DiCE optimisation also maximises the distance between multiple counterfactuals to ensure that they are diverse. With the CBR approach additional counterfactuals can be identified by increasing the neighbourhood. This ability to provide multiple counterfactuals to end-users has been found to improve end-user’s mental model. In our work we also adopt CBR’s NUN method to find counterfactuals but instead of the adaptation CBR step or the DiCE optimisation, we opt for feature relevance explainers to inform us about actionable features. In doing so we avoid

the need to create similarity-based explanation case-bases, yet maintain the advantage of locality-based explanations which ensure valid counterfactuals that are often harder to guarantee with optimisation methods.

Discussions regarding user acceptability and satisfiability of explanations has dominated XAI research in recent years [15]. Quantitative evaluation of counterfactual explanations focus on measures that can ascertain properties of good counterfactuals. In case-based counterfactual research, an explainer CBR system’s ability to solve future “explanation queries” is measured by explanatory competency [8]. This measures the fraction of queries that are currently explained by the explainer CBR system - coverage of the counterfactual cases. Authors claim that a good counterfactual will have at most two feature-differences (although this is not guaranteed by the explainer CBR system) and thereby maintain minimum plausible changes by operating within a local neighbourhood. Other related work also confirm the importance of measuring nearness and minimal changes to evaluate counterfactual explanations through proximity and sparsity measures [11]. Measures such as validity and diversity are also used with generative counterfactual XAI methods. However these are not applicable to our work, as case-based counterfactuals have the advantage of formulating plausible feature changes using valid cases in the case-base. Accordingly in our evaluation strategy, we use proximity and sparsity to compare counterfactual methods and also measure efficiency of actionable feature discovery.

III. FEATURE RELEVANCE EXPLAINERS

Feature relevance weights conveys the extent to which a feature contributes to a model’s output - i.e. greater weight indicates greater relevance of that feature to model decision-making. Features with the largest weights can therefore be used to explain the contributory input values that resulted in the output decision. In DisCERN, relevance weights are used to partially order features for actionable feature discovery. In the rest of this section we discuss and compare two widely used feature relevance explainers; LIME and SHAP.

A. Local Interpretable Model-agnostic Explanations (LIME)

LIME [9] is a model-agnostic feature relevance explainer which creates an interpretable model around a data instance to estimate how each feature contributed to the black-box model outcome. LIME creates a set of perturbations within the instance’s neighbourhood and labels them using the black-box model. This newly labelled dataset is used to create a linear interpretable model (e.g. a weighted linear regression model). The resulting surrogate model is interpretable and only locally faithful to the black-box model (i.e. correctly classifies the input instance, but not all data instances outside its immediate neighbourhood). The new interpretable model is used to predict the classification outcome of for the data instance. Thereafter an explanation of the predicted class is formed by obtaining the weights that indicate how each feature contributed to the outcome.

TABLE I: Comparison of Feature Relevance Explainers

Property	LIME	SHAP
Explainer Type	Post-hoc	Post-hoc
Model dependency	Model-agnostic	Model-agnostic
Explainability	Linear	Game Theory
Principal	Approximation	Inspired
Local Accuracy	✓	✓
Missingness	✓	✓
Consistency	-	✓

B. SHapley Additive exPlanation (SHAP)

SHAP [10] is a model-agnostic feature relevance explainer with theoretical guarantees about consistency and local accuracy from game theory and based on the shapley regression values [16]. Shapley values are calculated by creating linear models using subsets of features present in a case-base, X (i.e. a set of data instances). More specifically, a model is trained with a subset of features of size, m' , and another model is trained with a subset of features of size, $m' + \hat{m}$. Here, $m' + \hat{m} \leq m$, and the second model additionally includes a set of features, \hat{m} , selected from the set of features that were left out in the first model. A set of such model pairs are created for all possible feature combinations. For a given data instance that needs to be explained, the prediction differences of these model pairs are averaged to find the explainable feature relevance weights.

C. Feature Relevance Explainer Properties

LIME and SHAP are Post-hoc, model-agnostic feature relevance explainers (see Table I). Shapley values are considered to be *consistent* (i.e. the same query results in the same relevance explanation), unlike LIME which discovers relevance weights using perturbed data. Both SHAP and Lime guarantees local *accuracy* (i.e. the surrogate model and black-box model predicts the same outcome for a data instance); and *missingness* (i.e. ensures that there there is no weight contribution from a missing feature). Both provide a feature relevance weights vector, for any given query. In our counterfactual work the magnitude of the relevance weights, focus the search for the minimal subset of feature value changes that are likely to bring about a class change for a given query.

IV. METHODS

DisCERN algorithm uses feature relevance to identify the minimum subset of changes needed to form a counterfactual explanation from a retrieved NUN. Here we formalise the NUN counterfactual approach and thereafter discuss how weights from, feature relevance explainers, can be used in DisCERN to discover minimal number of actionable features.

A. Nearest-Unlike-Neighbour Counterfactual

The goal of a counterfactual explanation is to guide the end-user to achieve class change (i.e. actionable), with a focus on minimising the number of changes needed to flip the decision to a more desirable outcome. Given a query instance, $x = \{x_1, x_2, \dots, x_m\}$, with m features, its counterfactual,

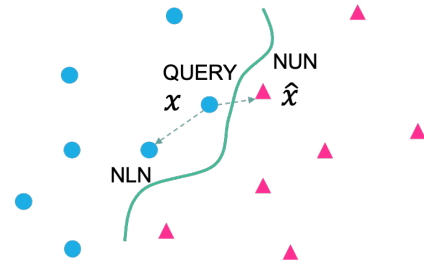


Fig. 1: Nearest Like and Unlike Neighbours in 2D space

$\hat{x} = \{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m\}$, is identified as the NUN in the case-base, X [2], as follows:

$$\hat{x} \leftarrow \arg \min_{x' \in X} d(x, x'); x, x', \hat{x} \in X; y \neq y' \quad (1)$$

Figure 1 illustrates a neighbourhood for a binary classed problem in two dimensional feature space ($m = 2$). It shows for a given query, how a NUN appears close to its decision boundary. In theory the closer it is to the boundary the fewer actionable changes are likely to be needed to flip the decision. However in practice certain types of feature changes (even if small) may be harder, or in some cases, even impossible to action (e.g. features related to demographics). Such discoveries may still be useful to unearth, because they can point to unethical or unfair practices.

Paired distances between the query and candidate NUNs (i.e. other data instances with a different class to that of the query) are computed using Euclidean distance (Equation 2).

$$d(x, \hat{x}) = \sqrt{\sum_{i=1}^m (x_i - \hat{x}_i)^2} \quad (2)$$

The optimal NUN will have the smallest distance to the query. Both the query's and NUN's feature values are used by DisCERN to formulate the counterfactual explanation. Here we identify two challenges: discovering the minimal number of actionable features (from a maximum of m potential feature changes) and minimising the amount of change required for each feature. We address these challenges using relevance weights.

B. Feature weights from a Feature Relevance Explainer

For an arbitrary data instance, p , the output of a feature relevance explainer, Rel , is a list of weights, $\mathbf{w} = (w_1, w_2, \dots, w_m)$, where w_i is a real-valued weight assigned to feature, p_i :

$$Rel(p) \rightarrow \{\mathbf{w} \mid \mathbf{w} \in \mathbb{R}^m\} \quad (3)$$

A positive weight ($w_i \geq 0$) indicates that the corresponding feature contributes positively and a negative weight ($w_i < 0$) contributes negatively towards the predicted outcome. These relevance weights are used to define an ordering constraint \mathcal{R} on features. Given a weights vector, \mathbf{w} , the overall value, $w(\cdot)$,

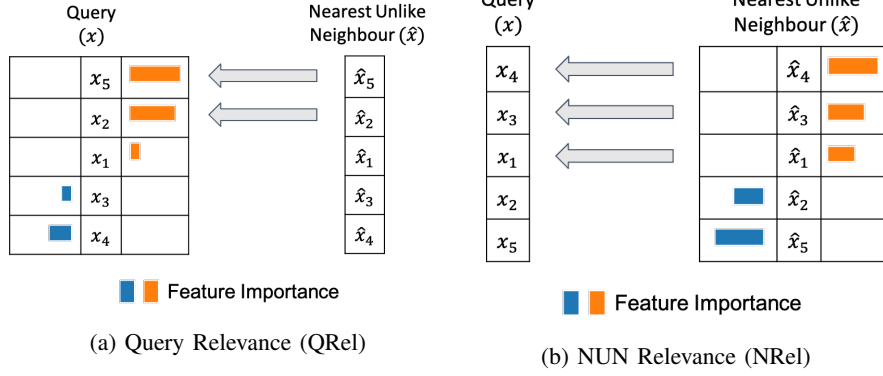


Fig. 2: Actionable Feature Discovery with QRel and NRel

is the weight lookup of a feature included in p . This is used to define the ordering \preceq on features:

$$p_i \preceq_{\mathcal{R}} p_j \iff \mathcal{R} :: w(p_i) \geq w(p_j) \quad (4)$$

C. Actionable Feature Discovery with Feature Relevance Explainers

The number of feature changes, n , required to achieve a class change, can range from 1 to m ($1 \leq n \leq m$). We propose two methods to discover actionable features, with the goal of minimising the number of feature changes (n) needed to form a counterfactual explanation. The first method replaces values of the most important features in the query with the corresponding feature values from its NUN; or a second alternative is to identify the most relevant features of the NUN and copy those feature values to modify the query instance.

Figure 2 shows a query with five features being adapted to form a counterfactual using two alternative value replacement methods: QRel in Figure 2a; and NRel in Figure 2b. With both methods, values are copied from the NUN, but the exact features selected for replacement change based on the instance parameter used in $Rel(\cdot)$. In QRel, the query features are ordered by their feature weights and the most important features are replaced by the respective NUN feature values. That is, QRel, uses $Rel(p = x)$ to obtain feature weights. With NRel, the NUN features are ordered by their feature weights returned by $Rel(p = \hat{x})$, and the most important features are reused by the query. Since each method imposes a different feature ordering, the number of changes needed to form the final counterfactual is likely to be different. In our example QRel and NRel achieve class change with 2 and 3 feature replacements respectively in Figure 2.

D. DisCERN Algorithm

Algorithm 1 brings together methods from Sections IV-A through IV-C to form NUN counterfactuals. Here, $y = \mathcal{F}(x)$, is the (black-box) classifier prediction for the query and, $\hat{y} = \mathcal{F}(\hat{x})$, is the class prediction for the NUN. Relevance weights from Section IV-C are identified in reference to either the query or the NUN, which we have denoted as p ; and the *Order* method (from Equation 4) provides a list of feature

Algorithm 1 DisCERN [Rel, p]

Require: (x, y) : query and label pair

Require: (\hat{x}, \hat{y}) : NUN and label pair

Require: \mathcal{F} : classification model

Require: p : either x or \hat{x} \triangleright as described in Section IV-C

Require: Rel : Feature Relevance Explainer

```

1:  $\mathbf{w} = Rel(p)$   $\triangleright$  see Equation 3
2:  $\hat{\mathbf{w}} = Order(\mathbf{w})$   $\triangleright$  see partial order Equation 4
3: Initialise  $y' = y; x' = x$   $\triangleright$  init counterfactual as query
4: for  $w_i \in \hat{\mathbf{w}}$  do
5:   if  $x'_i \neq \hat{x}_i$  then  $\triangleright i$  is the index of  $w_i$ 
6:      $x'_i \leftarrow \hat{x}_i$   $\triangleright$  copy NUN feature value
7:      $y' = \mathcal{F}(x')$   $\triangleright$  class prediction for counterfactual
8:     if  $y' \neq y$  then
9:       Break
10:    end if
11:  end if
12: end for
13: return  $x'$ 

```

indices ranked by the relevance weights. Feature values are iteratively replaced until the actionable change condition is met (i.e., $\mathcal{F}(x') = y' \wedge y \neq y'$) or the query is completely changed to the NUN (which is guaranteed to result in class change). Here the fewer the iterations needed the better the actionable features discovered. Once class change is achieved DisCERN returns x' as the counterfactual.

V. EVALUATION

The goal of this evaluation is twofold. Firstly to investigate different relevance feature explainers discussed in Section III with a view to finding which is best for feature weighting with DisCERN. Secondly to conduct a comparative study to determine the effectiveness of the DisCERN counterfactual explanation method (in Section IV-D) with the widely-used DiCE [11] and the baseline random feature ordered counterfactual creation methods.

Since DisCERN is a combination of a feature relevance explainer, Rel , and an actionable feature discovery method based

TABLE II: Dataset Properties

Datasets	Moodle	Loan-2015	Alcohol	Income	Credit
Features	95	77	5	15	15
Continuous Features	95	68	3	7	6
Categorical Features	0	9	2	8	9
Classes	2	2	2	2	2
Classification Accuracy	83%	97%	99%	84%	77%

on $Rel(p = x)$ or $Rel(p = \hat{x})$; we use the abbreviated notation DisCERN [Rel, p] to refer to these configuration combinations, where $Rel \leftarrow (\text{RND}|\text{LIME}|\text{SHAP}|\text{LIME}_C|\text{SHAP}_C)$; and $p \leftarrow (\text{QRel}|\text{NRel})$. For example DisCERN [LIME, QRel] denotes the combination of LIME weights with the QRel actionable feature discovery method for counterfactual creation. Note that RND refers to random selection of actionable features and therefore does not depend on either the query or NUN for parameter p , and is denoted as DisCERN [RND, Null].

A. Datasets

Datasets used in this paper are summarised in Table II, and used to carry out experiments as follows:

- comparative study of relevance feature explainers for feature weighting uses the Moodle dataset with results analysed in Section VI-A; and
- the counterfactual creation experiment is conducted using the Moodle dataset and four further datasets: Loan-2015, Alcohol, Income and Credit datasets. These results appear in Section VI-B.

Accuracy for each dataset is reported using a RandomForest classifier of 500 trees, which we found had better performance over several other black box models (including neural nets) in an initial set of fine-tuning experiments (with stratified 3 fold validation) For convenience with explanation experiments, we assume that the black box model has correctly predicted the outcome, and hence it made sense to work with the model with highest dataset accuracy. In this paper all features are considered candidates for actionable features¹. Details of how each dataset is preprocessed and used in a counterfactual explanation scenario are discussed next.

1) *Moodle Dataset*: is constructed from records of student footprints on the Moodle Virtual Learning Environment (VLE) for a single class delivered within Robert Gordon University, UK (RGU). VLE interactions help to capture vital touchpoints that can be used as proxy measures of student engagement. The dataset consists of 74 students enrolled on a Computer Science class during Semester 1 of 2020/2021 at RGU. It contains 95 features, where each feature is a learning resource stored on the Moodle VLE and the feature value is the number of times it was accessed by a student.

The ML task is to predict if a student gets a higher or a lower grade based on their Moodle footprint. This task is

¹In practice this is unlikely to be always possible. For instance features such as income, home ownership and length of employment in the Loan-2015 dataset are naturally non-actionable; and so are demographic features.

based on the assumption that there is some (causal or other) relationship between the Moodle access and the final grade of a student. We consider grades *A* and *B* as *Higher* grades and *C*, *D*, *E* and *F* as *Lower* grades. Grades were consolidated as *Higher* and *Lower* to mitigate the comparably lower number of data instances and class imbalance. This formed a dataset of 74 instances for a binary grade classification task, based on the Moodle footprint. The explanation intent relevant to this dataset is of the type *Why* student *A* did *not* receive a higher grade *X*? instead of *Why* did student *A* receive grade *Y*? The latter can be explained using a feature relevance explainer presenting the contribution of the most important features for the predicted grade; and the former *Why not* type questions are better explained through a counterfactual explanation with actionable features to guide the student to achieve a more desirable outcome in the future.

2) *Loan-2015 dataset*: is the subset of 2015 records from the Lending Club loan dataset on kaggle². We limit the dataset to records from 2015 to create the loan-2015 dataset of 421,095 data instances with 151 features. The ML task is to predict if a loan will be paid in full or not, and this outcome is used to accept or reject future loan requests. We apply data pre-processing steps recommended by the data providers to obtain a dataset with 342,865 instances and 115 features to perform binary classification. The desirable outcome for an end-user is that his/her loan request is accepted (i.e. similar users successfully re-paid their loans). For example an explanation intent here can trigger a question such as *Why* person *A* did *not* receive the loan? with a counterfactual pointing to those features in need of adjustments, before a desirable outcome is possible in the future.

3) *Alcohol Dataset*: is the Blood Alcohol Concentration (BAC) dataset which consists of 127,800 data instances with 7 features [17]. It includes features such as gender, if a meal was taken, the duration between the meal and BAC test. The ML task for this dataset is to predict if the BAC is over a regulatory limit. Accordingly, in a pre-processing step the dataset is converted in to a binary classification task by recognising the two classes with the BAC regulatory limit as the decision threshold. In common with the previous two datasets, one of the outcomes is more desirable than the other; which is of having the BAC below the acceptable threshold for driving. Accordingly, counterfactual explanations are sought by individuals who have a BAC above the threshold and are looking to understand how they might keep their BAC below the threshold by better managing one or more actionable features (e.g. such as periods between meal and alcohol consumption).

4) *Income Dataset*: contains 45222 data instances of personal data based on US 1994 Census database³. There are 15 features including demographic, educational, and other personal properties to predict their yearly income in US Dollars. The ML task for this data set is to predict, if the income

²<https://www.kaggle.com/wordsforthewise/lending-club>

³<https://archive.ics.uci.edu/ml/datasets/adult>

of a person is above or below 50k per year. Accordingly, in a pre-processing step the dataset is converted in to a binary classification task by setting the two classes to be less than or equal to 50k, and above 50k; with the latter being the desirable class. A counterfactual explanation is sought out by an individual with a salary below 50k and seeking to change one or more of their circumstances (such as their educational or demographic attributes) to acheive the higher salary class.

5) *Credit Dataset*: is a credit card application approval dataset with 653 data instances⁴. There are 15 anonymised features describing an applicant with the class indicating if the credit card application was approved or not. In this binary classification task the the model predicts if an applicant should be approved or not. For an applicant the desirable outcome is an “approved” credit card application. Accordingly, applicants who seek counterfactual explanations would have typically “failed” their credit card application and are looking to change this to an “approved” outcome by identifying necessary feature changes.

B. Performance Measures

Four quantitative performance measures (validity, proximity, sparsity and diversity) for evaluating counterfactuals were introduced in [11]. Proximity measures the mean feature-wise distance between a query and its counterfactual explanation. Sparsity refers to the number of features that are different between a query and its counterfactual explanation. Validity measures if the counterfactuals presented by the method actually belongs to the desirable class (i.e. not the same class as the query). Diversity measures the heterogeneity between multiple counterfactuals. Both validity and diversity are not relevant for our work because: 1) by selecting a NUN we ensure 100% validity; and 2) by selecting the most similar NUN counterfactual there is no requirement to measure diversity. In this paper we present two measures which correspond to sparsity and proximity respectively, but are not identical to the measures in [11].

1) *Mean number of feature changes (#F)*: required to achieve class change is calculated as follows:

$$\#F = \frac{1}{N \times m} \sum_{j=1}^N \sum_{i=1}^m 1_{[\hat{x}_i \neq x_i]} \quad (5)$$

Here the number of features with different values between the counterfactual (\hat{x}) and the query (x) are calculated and averaged; where N refers to the number of query instances, and m is the number of features.

2) *Mean amount of feature changes (\$F)*: required to achieve class change is calculated as follows:

$$\$F = \frac{1}{N \times \#F} \sum_{j=1}^N \sum_{i=1}^m (|\hat{x}_i - x_i|) \quad (6)$$

Here the sum of feature differences are average over $\#F$ and the number of query instances (N). All continuous features

TABLE III: Comparison of feature ordering strategies

DisCERN [Rel↓, p→]	#F		\$F	
	QRel	NRel	QRel	NRel
LIME	8.14	8.61	0.2642	0.2726
LIME _C	11.41	10.38	0.2308	0.2524
SHAP	7.69	8.32	0.2660	0.2454
SHAP _C	10.28	10.18	0.2085	0.2068
Chi2	12.27		0.2700	

are min/max normalised and therefore, continuous feature differences are between 0 and 1 whereas categorical feature differences are always 1 (using the overlap distance). Accordingly, datasets with more categorical features will have higher $\$F$, which means that the $\$F$ measure is not comparable across datasets. Note that smaller values of $\#F$ and $\$F$ are desirable.

VI. EXPERIMENTAL RESULTS

A. Comparison of Weights from Relevance Feature Explainers

For the first study we compare the following relevance feature explainers for DisCERN:

- 1) LIME: feature weights from the local feature relevance explainer discussed in Section III-A.
- 2) SHAP: feature weights provided by Shapley values discussed in Section III-B.
- 3) Chi2: global feature weights from the Chi-Squared test for feature selection.
- 4) LIME_C and SHAP_C: these are two class level feature relevance explainer versions of LIME and SHAP weights respectively, where for each class, the aggregated feature relevance is the mean feature relevance weights over all train data instances for that class.

A comparison of DisCERN settings with 5 alternative options for *Rel*; and 2 alternatives for *p* appear in Table III for the Moodle dataset. Each alternative’s performance is compared on $\#F$ and $\$F$ performance measures. Note that there is no difference between QRel and NRel when using Chi2 because feature weightings are global and not determined by a local data instance (be that the query or the NUN). Results show that SHAP achieves the best performance over LIME and Chi2 with both QRel and NRel methods (see bold font). SHAP using the QRel feature ordering method has achieved lowest $\#F$, whilst SHAP_C has the lowest $\$F$. However, since SHAP_C requires additional feature changes to achieve class change, we consider SHAP to be a preferable strategy. Moreover, we observe that LIME also achieves comparable performances for both $\#F$ and $\$F$. Notably, Chi2 failed to outperform both LIME and SHAP strategies in both minimising number of features and amount of change. Overall, these results emphasise the importance of feature relevance explainers as a proxy to identifying features important to achieving class change.

B. Evaluation of Actionable Feature Discovery

Table IV provides a comparison of five DisCERN configurations with the DiCE counterfactual algorithm. Here RND

⁴<https://archive.ics.uci.edu/ml/datasets/Credit+Approval>

TABLE IV: Comparison of counterfactual methods on $\#F$

Datasets	Moodle	Loan-2015	Alcohol	Income	Credit
DiCE	10.21	2.59	2.53	2.95	2.81
DisCERN [RND, Null]	21.62	21.91	2.16	2.92	3.18
DisCERN [LIME, QRel]	8.14	6.93	2.11	2.59	2.42
DisCERN [LIME, NRel]	8.61	6.69	2.15	2.64	2.50
DisCERN [SHAP, QRel]	7.69	6.86	2.11	2.70	2.48
DisCERN [SHAP, NRel]	8.32	5.51	2.12	2.68	2.42

TABLE V: Comparison of counterfactual methods on $\$F$

Datasets	Moodle	Loan-2015	Alcohol	Income	Credit
DiCE	0.6344	0.7763	0.6707	0.8497	0.8179
DisCERN [RND, Null]	0.2924	0.0569	0.0909	0.3545	0.2573
DisCERN [LIME, QRel]	0.2642	0.0660	0.0927	0.3643	0.2365
DisCERN [LIME, NRel]	0.2726	0.0675	0.0910	0.3570	0.2662
DisCERN [SHAP, QRel]	0.2660	0.0760	0.0929	0.3604	0.2343
DisCERN [SHAP, NRel]	0.2454	0.0711	0.0925	0.3258	0.2210

is a baseline counterfactual explanation comparator where actionable features are selected randomly instead of being informed by feature relevance weights. Results suggests that DisCERN in the QRel setting achieves the best performance on the Moodle, Alcohol and Income datasets and DiCE achieves best performance on the Loan-2015 dataset (see bold font). DisCERN using LIME relevance explainer achieves best performance on the Alcohol, Credit and Income datasets while SHAP matches the performance on the Alcohol and Credit Datasets. SHAP achieved best performance on the Moodle dataset. DisCERN with QRel and NRel achieve comparable performances across all datasets which resembles findings in Table III. Interestingly, DiCE failed to outperform Random feature ordering on the Alcohol and Income dataset which could be due to the limited amount of features available.

Overall DisCERN with SHAP and NRel strategy achieves class changes with lowest $\$F$ values (see bold font in Table V) on the Moodle, Income and Credit datasets. $\$F$ performance of DisCERN strategies are better compared to DiCE on all five datasets. For instance, for a query in the Loan-2015 dataset, the total amount of change ($\#F \times \$F$) with the DiCE method is $2.01(2.59 \times 0.7763)$ and with DisCERN [LIME,NRel] is $0.39(5.51 \times 0.0711)$. In situations where actionable features are not “easy to change”, it is more feasible to use DisCERN over DiCE. With DisCERN there was no single configuration (choices for Rel and p) that had out performed the rest across datasets. It is unusual that DisCERN with RND resulted in lowest $\$F$ on the Loan-2015 and Alcohol datasets. Accordingly, the choice of relevance explainer (i.e. LIME or SHAP) and ordering strategy (i.e. QRel or NRel) are seemingly dataset dependent.

C. DisCERN Counterfactuals

Figure 3 shows how DisCERN can be used to form counterfactual textual explanations using an example query from each dataset. For purposes of illustration, we selected queries that had a negative outcome and are good candidates to demonstrate actionable feature changes to achieve a desirable

class change. Here only those actionable features discovered using DisCERN are shown (and the other features including those with identical values are not). A template-based textual explanation generated from the counterfactual is also presented for each example. It is interesting to note that with all datasets, DisCERN is identifying important relationships such as that between a person’s weight, the time since the last meal, and the BAC level in the Alcohol Dataset or the relationship between working hours and salary in the income Dataset. Although some of these examples are genuine causal relationships, others are relationships which do not directly affect each others values. For instance in the Moodle dataset we observe increased access to learning materials is being related to a positive outcome; however this does not naturally translate to a causal relationship. Understanding the types of relationships that are discovered and using those to guide language generation templates remains an important open-area of research for the future.

VII. CONCLUSION

In this paper, we presented a novel approach to finding actionable knowledge when constructing an explanation using a counterfactual. We used feature relevance explainers as a strategy to discover features that are most significant to a predicted class and then used that knowledge to discover the actionable features to achieve class change with minimal change. We demonstrated our approach DisCERN using five datasets one of which (Moodle Dataset) is an original contribution.

Our empirical results showed that SHAP is the most optimal feature relevance explainer for ordering actionable features. DisCERN with QRel and NRel counterfactual methods introduced in this paper have either outperformed or achieved comparable performance over DiCE. The results have also highlighted the need to find balance between the number of feature changes and amount of feature change based on the selected actionable features. Accordingly, we find that there is conclusive evidence that feature relevance explainers are

Moodle Dataset	External tool: BbC	File: CM4107 CW1 Submission Template	File: CW2 Solution template stylefile	File: Coursework - Part I	File: Demo code: How to add hidden layers?	File: Lab Activities_2	File: Lecture Notes (draft)	late_coursework_1
Query	7.0	2.0	3.0	8.0	0.0	0.0	0.0	1.0
Counterfactual	23.0	1.0	1.0	11.0	4.0	2.0	1.0	0.0

If the student can increase engagement with online course content then a higher grade is likely. For instance consider increasing access to: the BbC tool (16x), Coursework Part 1 file (3x) lab activities_2 (2x), Lecture notes (draft) (1x) and demo on how to add hidden layers (4x); decrease access to: CW2 Solution template (2x) and CW1 submission template (1x); and ensure the coursework is submitted on time.

Loan-2015 Dataset	loan_amnt	recoveries	last_pymnt_amnt	last_fico_range_high
Query	6000.0	119.14	197.95	599.0
Counterfactual	5000.0	0.00	2052.39	779.0

The Loan Applicant will be successful, if the loan amount is reduced by \$1000, with no recoveries, and managing an increase in both the last payment by \$1854, and the upper boundary range for the borrower's last FICO by \$180.

Alcohol Dataset	wt	dp_mins	
Query	80.0	90.0	If the person weighs 81kg and their last meal was 92 minutes ago their BAC level will be below threshold.
Counterfactual	81.0	92.0	

Income Dataset	hours-per-week	
Query	40.0	If the Person worked 5 more hours per week, their income will be >\$50K per year.
Counterfactual	45.0	

Credit Dataset	A2	A3	A14	
Query	29.221513	1.293931	280.0	For this credit application to be approved, the applicant should consider the following changes: A2, A3 and A14 by approximately 11, -5 and -200 respectively.
Counterfactual	18.216110	6.108687	80.0	

Fig. 3: Example Counterfactuals

an important proxy to discovering actionable features and minimising the changes required. Future work will expand upon our evaluation to include additional real-world datasets and the use of qualitative evaluation through crowd-sourcing techniques.

REFERENCES

- [1] S. Mohseni, N. Zarei, and E. D. Ragan, "A survey of evaluation methods and measures for interpretable machine learning," *arXiv preprint arXiv:1811.11839*, 2018.
- [2] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the gdpr," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.
- [3] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," in *The Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 3530–3537, 2018.
- [4] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.
- [5] E. M. Kenny and M. T. Keane, "Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ann-cbr twins for xai," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 2708–2715, International Joint Conferences on Artificial Intelligence Organization, 7 2019.
- [6] N. Wiratunga, I. Koychev, and S. Massie, "Feature selection and generalisation for retrieval of textual cases," in *European Conference on Case-Based Reasoning*, pp. 806–820, Springer, 2004.
- [7] D. Wetschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, no. 1, pp. 273–314, 1997.
- [8] M. T. Keane and B. Smyth, "Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai)," in *International Conference on Case-Based Reasoning*, pp. 163–178, Springer, 2020.
- [9] M. T. Ribeiro, S. Singh, and C. Guestrin, "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*, pp. 1135–1144, 2016.
- [10] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, vol. 30, pp. 4765–4774, 2017.
- [11] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pp. 607–617, 2020.
- [12] N. J. Roese, "Counterfactual thinking.," *Psychological bulletin*, vol. 121, no. 1, p. 133, 1997.
- [13] P. L. Harris, T. German, and P. Mills, "Children's use of counterfactual thinking in causal reasoning," *Cognition*, vol. 61, no. 3, pp. 233–259, 1996.
- [14] J. Timmis and C. Edmonds, "A comment on opt-ainet: An immune network algorithm for optimisation," in *Genetic and Evolutionary Computation Conference*, pp. 308–317, Springer, 2004.
- [15] R. Hoffman, S. T. Mueller, G. Klein, and J. Litman, "Metrics for explainable ai: Challenges and prospects," *ArXiv*, vol. abs/1812.04608, 2018.
- [16] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games*, pp. 307–317, 1953.
- [17] P. Cunningham, D. Doyle, and J. Loughrey, "An evaluation of the usefulness of case-based explanation," in *International conference on case-based reasoning*, pp. 122–130, Springer, 2003.