Generating Counterfactual Explanations for Clustering Models Based on Their Equivalence to Classification Models

Antonia Karra, Georgios Vardakas, Evaggelia Pitoura, and Aristidis Likas(✉)

Department of Computer Science and Engineering, University of Ioannina, 45110 Ioannina, GR, Greece
{a.karra,g.vardakas,pitoura}@uoi.gr, arly@cs.uoi.gr

Abstract. Counterfactual explanations are a widely adopted approach for interpreting the decisions of machine learning models, mainly in the context of classification problems. A wide variety of techniques have been proposed for the classification task. In this work, we address the generation of counterfactuals for explaining clustering decisions. We focus on k-means and Gaussian clustering and tackle counterfactual generation by defining equivalent classification problems. More specifically, a linear classifier is defined in the k-means case and a quadratic discriminant classifier is defined in the Gaussian clustering case. In this way, widely used methods developed in the classification context can be employed for clustering. We also propose a way to increase the plausibility of the generated countefactuals by moving the cluster boundary towards the target cluster. Experimental results on synthetic and real datasets demonstrate the feasibility and effectiveness of our approach.

Keyword: Counterfactuals, Clustering, k-means, Gaussian cluster.

1 Introduction

Explainable AI (XAI) is essential for developing trustworthy and transparent machine learning models, enabling users to understand and effectively interact with these systems. This is particularly critical in high-stakes domains such as healthcare, finance, and law, where decisions made by machine learning models have significant consequences. By providing clarity on how and why models make certain predictions, XAI fosters trust and accountability while supporting regulatory compliance. Explainability methods can be broadly categorized into *global* and *local* explanation approaches [14].

Global explanation methods aim to provide insights into the overall behavior of a model, summarizing its decision-making processes across the entire dataset. Techniques such as decision trees, rule-based systems, and feature importance rankings fall into this category. For instance, a decision tree approximation of a complex neural network can offer a simplified view of how the network generally behaves, making it easier to interpret. These methods are particularly useful

Published by Springer Nature Switzerland AG 2025
I. Maglogiannis et al. (Eds.): AIAI 2025, IFIP AICT 755, pp. 85–100, 2025.
https://doi.org/10.1007/978-3-031-96239-4_7

when a high-level understanding of the model's decision making logic is needed, such as identifying which features are most influential in shaping predictions across all samples [12].

Local explanation methods focus on *individual* predictions or decisions, providing insights into specific outcomes. These methods are valuable when users need to understand why a model made a particular decision. Several local explanation approaches have been proposed, such as:

- *LIME (Local Interpretable Model-Agnostic Explanations)*: LIME approximates a complex model locally around a specific prediction using a simpler and interpretable model. For instance, LIME can generate an explanation for why a particular text was classified as "spam" by highlighting the most influential words that contribute to the prediction [14].
- *SHAP (SHapley Additive exPlanations)*: SHAP values are grounded in cooperative game theory and assign each feature a contribution score for a given prediction. For example, SHAP can explain why a certain patient was predicted to have a high risk of a disease by quantifying the influence of specific symptoms or test results [11].
- *Saliency Maps*: Introduced in the context of deep learning, saliency maps highlight the most influential features that contribute to a specific prediction [1]. For instance, in an image classification task, a saliency map can reveal which regions of the image the model focused on to classify it as a 'cat'.
- *Counterfactual Explanations* which constitute the focus of this work and are explained below.

Counterfactual explanations (CFEs) provide an intuitive *local explanation* framework to understand the decisions of machine learning models by exploring "what-if" scenarios [6,20]. Specifically, they identify the minimal changes required in the input features of a model (usually a classifier) to achieve a desired outcome. These explanations are actionable, offering users practical insights into how to modify inputs to change predictions. The widely used loan qualification example offers a clear and intuitive illustration of the counterfactual concept: Consider a scenario with two possible outcomes for a model's decision: "reject" where the applicant does not qualify for the loan, and "approve", where the applicant does qualify. If an individual with feature vector y is denied the loan because of the model's decision, i.e., $f(y)$='reject', an explanation for this decision is crucial for the individual. Such an explanation could illustrate the minimum required changes to y (for example, adjustments to income, education, or other factors) in order for the individual to qualify for the loan in the future.

Formally, given a classifier f that assigns the class c to an instance y, a counterfactual explanation is an instance z that satisfies the following conditions: i) the classifier's decision on z differs from that on y, i.e., $f(z) \neq c$, ii) the difference between y and z is minimized [6,20].

Furthermore, various important properties of counterfactual explanations in classification have been identified, each contributing to their interpretability and practical utility. Two key properties are:

- Actionability/Immutability: Certain features may be restricted from modification due to real-world constraints. For example, attributes such as age or ethnicity in a loan approval model cannot be altered, ensuring that counterfactual explanations remain meaningful and applicable in practice. In addition, constraints in the allowed ranges of feature values could be imposed.
- Plausibility: A counterfactual instance should be realistic and belong to the underlying data distribution. That is, it should represent a feasible observation rather than an arbitrary perturbation, ensuring that the suggested changes align with patterns seen in real-world data.

These properties play a crucial role in generating counterfactual explanations that are not only technically valid but also useful for decision making and for understanding model behavior.

In direct analogy to classification, if we assume a *clustering model* f that provides cluster labels, a counterfactual explanation of a factual instance y assigned to cluster c, consists of an instance z such that:

- the cluster assignment of z provided by f is different from c, i.e., $f(z) \neq c$
- the difference between y and z is minimum.

In this work, we propose a methodology for computing counterfactuals for clustering by employing established tools and methods that have been developed for classification problems. This is achieved by exploiting the equivalence between:

- the k-means cluster assignment rule with a linear classifier and
- the Gaussian clustering assignment rule with a quadratic discriminant analysis classifier.

In other words, given the parameters of a cluster assignment model, the parameters of an equivalent classification model are determined. Consequently, generating counterfactuals for clustering is transformed to an equivalent problem of generating counterfactuals for classification. The latter problem can be tackled using widely used methods developed in the classification context. We also suggest a way to increase the plausibility of the generated countefactuals by moving the cluster boundary towards the target cluster. To assess the feasibility and effectiveness of the proposed methodology, we have employed two well-known techniques for explaining classification decisions applied to synthetic and real datasets. We present comparative results with the commonly used $L1$ norm as a proximity measure.

The rest of the paper is organized as follows. In Sect. 2 we briefly discuss the related work on explainable clustering and counterfactual generation. In Sects. 3 and 4 we define the equivalent classification problems for k-means and Gaussian clustering, respectively. In addition, we discuss how to increase the plausibility of the counterfactuals produced. In Sect. 5 we report the results and conclusions of our experimental study, while Sect. 6 provides concluding remarks and future work suggestions.

2 Related Work

2.1 Explainable Clustering

Explainable methods for clustering provide *global explanations*, often leveraging decision tree models to provide interpretations through decision rules. Various methods have been proposed to construct decision trees for explaining clustering, that can be distinguished in two categories.

Indirect global explanation methods follow a two-step process to provide interpretable insights into clustering results. In the first step, cluster labels are generated using a clustering algorithm such as k-means. These labels, which represent the cluster assignments for the data points, are then treated as target variables for the second step. In this step, a supervised decision tree algorithm is applied to construct a decision tree that explains the cluster assignments. For example, in [9], the cluster labels produced by k-means are utilized as a preliminary input to guide the decision tree construction. Similarly, in [17], the centroids computed by k-means play a key role in the splitting process of decision tree construction.

Direct global explanation methods combine the processes of decision tree construction and cluster partitioning into a unified framework. These methods often adopt the conventional top-down splitting procedure commonly used in supervised learning but adapt it to leverage unsupervised splitting criteria. For instance, they may focus on optimizing the compactness of the resulting subsets, as discussed in [2], or employ criteria such as unimodality to guide the splits, as demonstrated in [4]. By aligning the decision tree construction with clustering objectives, these methods ensure that the resulting explanations more directly reflect the underlying structure of the data.

2.2 Counterfactuals for Classification

Counterfactual explanations are local explanations proposed for classification [6,20,21]. Let f be a classification model and $d(x, y)$ a distance function. Given an example (factual) y, its counterfactual explanation z is a data point close to y whose outcome $f(z)$ differs from the prediction $f(y)$. More formally:

$$z = \arg\min_{x} d(x, y) \quad \text{s.t.} \quad f(z) \neq f(y). \tag{1}$$

Assuming a target class c^*, the typical approach to tackle the above constraint minimization problem is to minimize the following objective function [21]:

$$z = \arg\min_{x} \; d(x, y) + \lambda \cdot L(f(x), c^*), \tag{2}$$

where:

- $L(f(x), c^*)$ is a loss function penalizing the difference between $f(x)$ and c^*
- λ is a regularization parameter that balances between minimal changes and achieving the desired outcome.

Various search methods have been studied to solve the above problem ranging from gradient-based [5,7,10,13] to genetic algorithms [15,16]. Those methods have been implemented in widely used software libraries for generating counterfactuals for classification. They support a variety of distance functions (e.g. L_1, L_2 norms), classification models f as well as loss functions L. In addition they allow to specify feasibility constraints such as defining actionable and immutable features and imposing constraints on the permitted range of feature values. We have employed two of those tools in our experimental study.

In [19] we have introduced a method for computing optimal counterfactuals for k-means and Gaussian clustering specifically targeting the $L2$ norm between factual and counterfactual instances. Instead, in this paper, we propose a methodology that builds on previous work on counterfactuals for classification and it is applicable to any distance measure. In our experiments we demonstrate its effectiveness by presenting results using the $L1$ norm.

3 Counterfactuals for *k*-Means Clustering

We assume a clustering solution with $C_1, ..., C_M$ clusters obtained by applying k-means with M clusters on a given dataset. Let m_k denote the center of cluster C_k. Based on the k-means cluster assignment rule, an example x is assigned to cluster C_ℓ whose center m_ℓ is of minimum distance from x.

In order to compute a counterfactual explanation z for a given example (factual) y, we need to specify a distance function $d(x, y)$. Given a factual y of cluster C_k, its counterfactual explanation z is defined as the solution to the following constrained optimization problem:

$$z = \arg\min_x d(x, y) \tag{3}$$

subject to the constraint:

$$C_\ell \neq C_k \text{ where } C_\ell = \arg\min_{C_j} |z - m_j|^2 \tag{4}$$

In the above formulation, C_ℓ is the cluster label of z, taking into account the cluster assignment rule.

3.1 Equivalence to Linear Classification

We consider counterfactual generation assuming a two-cluster problem: with C_f denoting the cluster of factual y and C_{cf} the target cluster of counterfactual z. The reason is that, if more than one target clusters exist, we can solve separately for each possible target cluster, compute the corresponding counterfactual, and select as final solution the counterfactual with minimum distance from the factual.

Let m_f and m_{cf} the cluster means (centers) provided by k-means algorithm for the two clusters of interest. Based on the k-means assigment rule, in order for the counterfactual z to be assigned to target cluster C_{cf} it should hold that:

$$|z - m_{cf}|^2 < |z - m_f|^2 \tag{5}$$

which leads to the following cluster assignment rule:

$$(m_{cf} - m_f)^\top z - \frac{1}{2}(|m_{cf}|^2 - |m_f|^2) > 0 \tag{6}$$

It is obvious that the above decision function is equivalent to a two-class linear classification model ($w^\top z + w_0 > 0$) with parameters:

$$w = m_{cf} - m_f \tag{7}$$

$$w_0 = -\frac{1}{2}(|m_{cf}|^2 - |m_f|^2). \tag{8}$$

Therefore, in order to compute counterfactuals for k-means clustering, we can resort to methods and tools that provide counterfactuals to explain the decisions of a linear classifier whose parameters are determined by the cluster centers using the above equations.

3.2 Plausibility

Counterfactual plausibility is increased if we force the generated counterfactual to lie in high density regions of the target cluster. This can be achieved by transferring the linear classifier decision boundary away from the k-means boundary and inside the target cluster region. The idea can implemented using a distance parameter $\delta > 0$ specified by the user and assuming that the counterfactual z is assigned to the target cluster if:

$$|z - m_{cf}|^2 < |z - m_f|^2 - \delta \tag{9}$$

This leads to the following decision function:

$$(m_{cf} - m_f)^\top z - \frac{1}{2}(|m_{cf}|^2 - |m_f|^2 + \delta) > 0 \tag{10}$$

It is obvious that the above decision function is equivalent to a two-class linear classification model ($w^\top z + w_0 > 0$) with parameters:

$$w = m_{cf} - m_f \tag{11}$$

$$w_0 = -\frac{1}{2}(|m_{cf}|^2 - |m_f|^2 + \delta). \tag{12}$$

Note that if $\delta = 0$, we resort to the typical k-means case described previously.

4 Counterfactuals for Gaussian Clusters

We now consider the case of Gaussian clustering. We assume a partition of a dataset into M clusters that is generated by training a Gaussian Mixture Model with M Gaussian components [3]:

$$p(x) = \sum_{k=1}^{M} \pi_k N(x; \mu_k, \Sigma_k) \tag{13}$$

where μ_k is the mean and Σ_k is the covariance matrix of the k-th Gaussian distribution

$$N(x; \mu_k, \Sigma_k) = \frac{1}{2\pi|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k)\right) \tag{14}$$

and π_k denotes the mixing weight (prior) of Gaussian component k. We consider the more general case of full covariance matrices Σ_k.

The cluster assignment rule in the Gaussian clustering case assigns a data point x to the cluster k for which $\pi_k N(x; \mu_k, \Sigma_k)$ is maximum. This leads to the following formulation for counterfactual computation. Given a distance function $d(x, y)$ and a factual y of cluster C_k, its counterfactual z is defined as the solution to the following constrained optimization problem:

$$z = \arg\min_x d(x, y) \tag{15}$$

subject to the constraint:

$$C_\ell \neq C_k \text{ where } C_\ell = \arg\max_{C_j} \pi_j N(x; \mu_j, \Sigma_j) \tag{16}$$

In the above formulation, C_ℓ is the cluster label of z, taking into account the cluster assignment rule.

4.1 Equivalence to Quadratic Discriminant Analysis

As with the k-means case, we consider a two-cluster problem, where C_f denotes the cluster of factual y and C_{cf} the target cluster of counterfactual z. If more than one target clusters exist, we can compute a counterfactual for each possible target and select as final solution the one with minimum distance from the factual.

Let π_f, μ_f and Σ_f the parameters of the Gaussian cluster of the factual and π_{cf}, μ_{cf} and Σ_{cf} the parameters of the Gaussian target cluster. Based on the cluster assignment rule, in order for a data point z to be a counterfactual, it should be assigned to the target cluster. This meas that:

$$\pi_{cf} N(x; \mu_{cf}, \Sigma_{cf}) > \pi_f N(x; \mu_f, \Sigma_f) \tag{17}$$

Taking the logarithm on both sides of the above equation and using the Gaussian definition of Eq. 14 we find that:

$$\ln \pi_{cf} - \frac{1}{2} \ln |\Sigma_{cf}| - \frac{1}{2}(x - \mu_{cf})^\top \Sigma_{cf}^{-1}(x - \mu_{cf}) > \\ + \ln \pi_f - \frac{1}{2} \ln |\Sigma_f| - \frac{1}{2}(x - \mu_f)^\top \Sigma_f^{-1}(x - \mu_f) \tag{18}$$

This is equivalent to the decision rule of a two-class discriminant classifier analysis whose discriminant functions $g_{cf}(x)$ and $g_f(x)$ are given by the left and right side, respectively, of the above inequality. Since the discriminant functions are quadratic, the decision rule in Gaussian clustering is equivalent to a quadratic discriminant analysis (qda) classifier where each cluster corresponds to a class and each quadratic discriminant function is determined by the parameters of the corresponding Gaussian cluster.

Therefore, in order to compute counterfactuals for Gaussian clustering, we can resort to methods and tools that provide counterfactuals to explain the decisions of the corresponding quadratic classifier.

4.2 Plausibility

If we wish the generated counterfactual to lie in more dense regions of the target cluster, we can modify the decision rule specified by Eq. 17 as follows:

$$\pi_{cf} N(x; \mu_{cf}, \Sigma_{cf}) > \epsilon \cdot \pi_f N(x; \mu_f, \Sigma_f) \tag{19}$$

where $\epsilon > 1$ is user specified parameter. In this way the quadratic decision boundary is moved in regions of higher target cluster density. Therefore, the generated counterfactual would lie in more dense areas of the target cluster compared to the case where $\epsilon = 1$ presented in the previous subsection.

It can be easily observed that the above cluster assignment rule (Eq. 19), is equivalent to a quadratic decision rule as the one presented in the previous subsection (Eq. 18), with the only difference being that the quantity π_f should be replaced by $\epsilon \cdot \pi_f$.

5 Experimental Study

In this section we provide experimental results using the proposed methodology for generating counterfactuals for k-means and Gaussian clustering.

5.1 Experimental Setup and Methods Used

For a given dataset, we begin by performing clustering using either k-means or a Gaussian mixture model with full covariance. Next, we designate a source cluster C_f and a target cluster C_{cf} and derive the parameters of the corresponding classification models.

Two widely used methods for generating counterfactuals for classification have been employed, namely DiCE (DICE) [13] and GuidedPrototypes (PRT) [18]. For DICE, we use the implementation provided by the authors[1], while for PRT, we use the implementation in Alibi Explain[2]. Both implementations provide the flexibility to use several distance functions between factual and counterfactual as well as to define actionable and immutable features. In our experiments *the L1 distance function has been selected*, which is the most popular in the relevant literature, since it usually provides sparse solutions in terms of the features required to change. It should be noted that both DICE and PRT do not guarantee the successful generation of counterfactuals, i.e., that the counterfactual instance would necessarily belong to the target cluster.

DiCE. In order to generate a set $C = \{c_1, \ldots, c_k\}$ of counterfactuals, DICE minimizes the following objective function assuming a classification model f, a factual x and target class y:

$$C(x) = \arg\min_{c_1,\ldots,c_k} \frac{1}{k}\sum_{i=1}^{k} \text{yloss}(f(c_i), y) + \frac{\lambda_1}{k}\sum_{i=1}^{k} \text{dist}(c_i, x) - \lambda_2 \cdot \text{dpp\_diversity}(c_1, \ldots, c_k) \quad (20)$$

where $yloss(f(c_i), y)$ is the classification error for c_i, $dist(c_i, x)$ is the distance between c_i and x (L_1 in our case) and $dpp\_diversity$ promotes variety of the generated counterfactuals. In our experiments we use the following values, optimized through fine-tuning, to maximize the percentage of counterfactuals successfully generated by the method: $total\_cfs = 1$ (a single counterfactual is required, $k = 1$), $\lambda_1 = 1$, $\lambda_2 = 0$ and $stopping\_threshold = 0.5$, indicating the minimum target class probability.

GuidedPrototypes. The following objective function is minimized in the PRT method:

$$L = c \cdot L_{\text{pred}} + \beta_1 \cdot L_1 + \beta_2 \cdot L_2 + \gamma L_{\text{AE}} + \theta L_{\text{proto}} \quad (21)$$

where the prediction loss L_{pred} enforces the counterfactual to belong to the target class, while L_1 and L_2 denote distance norms between factual and counterfactual. The loss term L_{AE} is used to keep the counterfactual in the data manifold by minimizing an autoencoder reconstruction error, while L_{proto} guides counterfactuals towards the prototype of the target class for distribution alignment. Note that in PRT the dataset should be available. The following parameters and corresponding values of the Alibi Prototypes framework were used: $\beta_1 = 1$, $\beta_2 = 0$, $\theta = 100$, $\gamma = 0$, $max\_iterations = 5000$, the $kdtree$ option is selected and the $feature\_range$ option is used to account for immutable features. The parameter

[1] https://github.com/interpretml/DiCE.
[2] https://docs.seldon.io/projects/alibi/en/stable/.

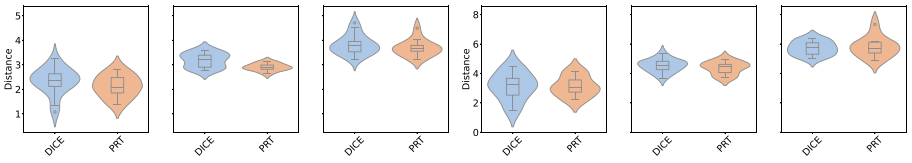
$c_{init} = 1.0$ defines the initial value for c, while c_{steps} determines the number of adjustment steps taken during the search process. In our tests, the c_{steps} value was 3 or 5. We set $c_{steps} = 5$ when one or more features were constrained to be immutable, thus the problem was more difficult to solve.

5.2 Performance Results

We now present results regarding the quality of the generated counterfactuals for each method. We use both synthetic and real datasets. Synthetic datasets are generated using the scikit-learn *make_blobs*[3] function. We report results with a synthetic dataset (3D) with three features and two Gaussian clusters. We also use two well-known real datasets, *Wine* and *Pendigits* [8]. The Wine dataset consists of three wine categories and 13 numerical features representing the chemical properties of the wine. The Pendigits dataset contains ten digit categories with 16 numerical features capturing the pen's trajectory. Class labels were not utilized by the clustering algorithms, and the number of clusters was set equal to the number of classes. For the experiments two of the produced clusters were selected as source (C_f) and target (C_{cf}) clusters. For each dataset, we have studied two cases, one where we impose no constraints on the features and another one where we consider a number of features as immutable, i.e., their value is not allowed to change.

For each cluster pair, 50 factuals were randomly selected partitioned into three groups, namely *near*, *intermediate* and *far*, based on their distance from the center of the target cluster m_{cf}. In general, a factual that is closer to the target center is also expected to be closer to the decision boundary of the classifier.

At first, we report the L_1 distance between each factual and its generated counterfactual. Violin and box plots are presented to illustrate the detailed distribution of L_1 distances. Figure 1 – 3 depict L_1 distributions for k-means clustering, and Fig. 4 – 6 for Gaussian clustering.

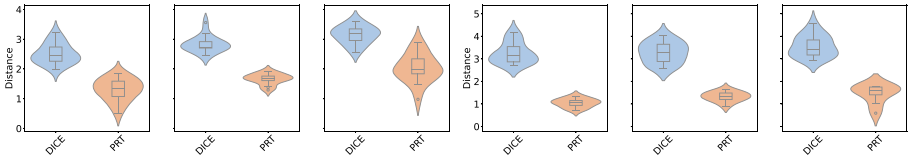


(a) No feature constraint: near, inter, far (b) 2 immutable features: near, inter, far

Fig. 1. k-means clustering: L_1 distribution for the 3D synthetic dataset

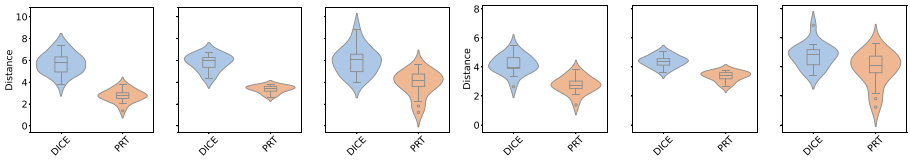
In most cases, the counterfactuals generated by PRT have a shorter L_1 distance from the corresponding factuals than the counterfactuals generated by

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



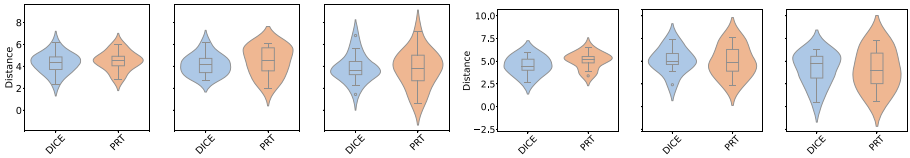
(a) No feature constraint: near, inter, far (b) 7 immutable features: near, inter, far

Fig. 2. k-means clustering: L_1 distribution for the Wine dataset



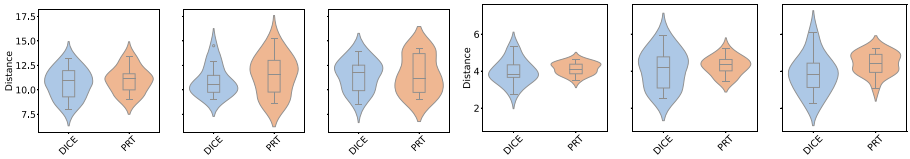
(a) No feature constraint: near, inter, far (b) 3 immutable features: near, inter, far

Fig. 3. k-means clustering: L_1 distribution for the Pendigits dataset.



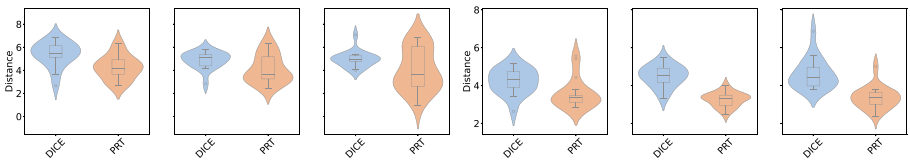
(a) No feature constraint: near, inter, far (b) 2 immutable features: near, inter, far

Fig. 4. Gaussian clustering: L_1 distribution for the 3D dataset.



(a) No feature constraint: near, inter, far (b) 7 immutable features: near, inter, far

Fig. 5. Gaussian clustering: L_1 distribution for the Wine dataset.



(a) No feature constraint: near, inter, far (b) 3 immutable features: near, inter, far

Fig. 6. Gaussian clustering: L_1 distribution for the Pendigits dataset.

DICE, especially in the case of k-means clustering. As expected, in most cases, the L_1 distance is larger for the "intermediate" and "far" groups of factuals, since they are located in further distances from the decision boundary.

Table 1. Experimental results using DICE and PRT for k-means clustering.

Dataset	Time (secs)		Percentage		t-test
	DiCE	PRT	DiCE	PRT	
Synthetic-Near	0.04	114.85	100%	64%	=
Synthetic-Inter	0.05	115.86	100%	62%	PRT
Synthetic-Far	0.05	116.84	100%	76%	=
Synthetic 2 Imm. Feat.-Near	0.08	349.49	93%	64%	=
Synthetic 2 Imm. Feat.-Inter	0.09	348.85	88%	62%	=
Synthetic 2 Imm. Feat.-Far	0.09	392.15	88%	88%	=
Wine-Near	0.10	159.68	100%	100%	PRT
Wine-Inter	0.11	155.84	100%	100%	PRT
Wine-Far	0.10	150.93	100%	100%	PRT
Wine 7 Imm. Feat.-Near	0.06	165.73	100%	70%	PRT
Wine 7 Imm. Feat.-Far	0.08	162.91	100%	68%	PRT
Wine 7 Imm. Feat.-Inter	0.08	166.47	100%	47%	PRT
Pendigits-Near	0.12	125.51	100%	100%	PRT
Pendigits-Inter	0.13	126.29	100%	100%	PRT
Pendigits-Far	0.13	124.42	100%	100%	PRT
Pendigits 3 Imm. Feat.-Near	0.36	130.27	100%	100%	PRT
Pendigits 3 Imm. Feat.-Inter	0.38	127.52	100%	100%	PRT
Pendigits 3 Imm. Feat.-Far	0.38	128.46	100%	100%	PRT

Additionally, we conducted a statistical comparison of the L_1 distances provided by DICE and PRT, using t-test with statistical significance level equal to 0.05. The results are reported in the last column of Table 1 for k-means clustering and Table 2 for Gaussian clustering. In the case where the difference is statistically significant, we report the superior method, i.e., the one that generates counterfactuals at the smaller L_1 distance. Otherwise, we use the "=" symbol.

We have also compared DICE and PRT in terms of execution time. The results in Tables 1 and 2 indicate that DICE is more than three orders of magnitude faster than PRT. The percentage of successfully generated counterfactuals is also reported in those tables. Making some of the features immutable makes the problem of generating counterfactuals harder, resulting in smaller percentages of counterfactuals found. However, the distance does not necessarily increase when some features are immutable. Based on the t-test results, in all cases PRT either generates counterfactuals at shorter L_1 distance from the corresponding factual compared to DICE, or the L_1 distances for both methods are comparable.

Table 2. Experimental results using DICE and PRT for Gaussian clustering.

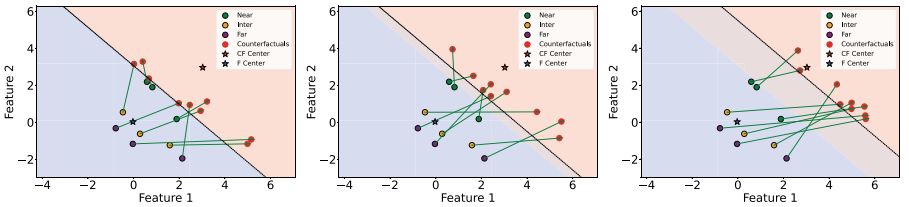
Dataset	Time (secs)		Percentage		t-test
	DiCE	PRT	DiCE	PRT	
Synthetic-Near	0.05	133.35	100%	100%	=
Synthetic-Inter	0.05	133.37	100%	100%	=
Synthetic-Far	0.05	132.70	100%	100%	=
Synthetic 2 Imm. Feat.-Near	0.11	319.08	58%	70%	=
Synthetic 2 Imm. Feat.-Inter	0.14	308.15	62%	62%	=
Synthetic 2 Imm. Feat.-Far	0.09	307.75	76%	70%	=
Wine-Near	0.09	137.05	100%	100%	=
Wine-Inter	0.09	114.14	100%	100%	=
Wine-Far	0.09	117.75	100%	100%	=
Wine 7 Imm. Feat.-Near	0.08	130.71	100%	100%	=
Wine 7 Imm. Feat.-Far	0.08	114.10	100%	100%	=
Wine 7 Imm. Feat.-Inter	0.08	118.98	100%	100%	=
Pendigits-Near	0.09	151.02	100%	100%	PRT
Pendigits-Inter	0.08	142.51	100%	93%	PRT
Pendigits-Far	0.08	147.09	100%	100%	PRT
Pendigits 3 Imm. Feat.-Near	0.38	211.12	100%	100%	PRT
Pendigits 3 Imm. Feat.-Inter	0.38	218.41	100%	100%	PRT
Pendigits 3 Imm. Feat.-Far	0.37	231.89	100%	100%	PRT

5.3 Illustrating Plausibility

We next present a k-means clustering example that illustrates how the increase of the distance parameter value δ (Subsect. 3.2) leads to counterfactuals of increased plausibility. We consider a 2-d synthetic dataset containing two spherical clusters with centers m_f and m_{cf} respectively. We set the parameter δ as a percentage ϵ of the squared distance between the cluster centers: $\delta_\epsilon = \epsilon \cdot |m_f - m_{cf}|^2$.

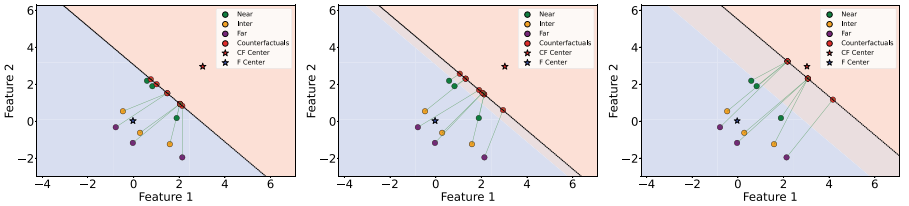
Figure 7 and 8 display the decision boundary and the counterfacuals generated using DICE and PRT for increasing values of ϵ, specifically $\epsilon = 0$, 0.1 and 0.4. It can be observed that as ϵ increases, the decision boundary moves towards the target cluster and, consequently, the distance of the generated counterfactuals from the center of the target cluster decreases. In this way plausibility is increased.

Comparing DICE (Fig. 7) and PRT (Fig. 8), it appears that PRT generates counterfactuals at shorter distances (closer to the boundary) than DICE, suggesting that PRT prioritizes proximity more strongly, even as δ increases.



(a) $\epsilon = 0$ (b) $\epsilon = 0.1$ (c) $\epsilon = 0.4$

Fig. 7. Factual instances and their corresponding counterfactuals generated using the DICE method for increasing values of ϵ.



(a) $\epsilon = 0$ (b) $\epsilon = 0.1$ (c) $\epsilon = 0.4$

Fig. 8. Factual instances and their corresponding counterfactuals generated using the PRT method for increasing values of ϵ.

6 Conclusions

We presented and studied a methodology to generate counterfactual explanations for clustering decisions, specifically targeting k-means and Gaussian clustering. Counterfactual generation for k-means clustering was achieved by defining an equivalent linear classification problem, while for Gaussian clustering a quadratic discriminant classifier was used. This reformulation enabled the use of widely adopted counterfactual explanation techniques originally designed for classification tasks. To enhance the plausibility of the generated counterfactuals, we introduced a strategy that involves shifting the cluster boundary towards the target cluster. We tested our methodology on synthetic and real datasets using two widely used techniques that generate counterfactuals for classification models selecting the L_1 norm as a proximity measure between counterfactual and factual.

Future work could focus on generating a diverse set of counterfactuals for a given factual as well as generating a counterfactual for a group of factuals. It would also be very interesting to explore counterfactual generation in a deep clustering framework.

Acknowledgment. This research project is implemented in the framework of H.F.R.I. call "Basic research Financing (Horizontal support of all Sciences)" under the National

Recovery and Resilience Plan “Greece 2.0” funded by the European Union - NextGenerationEU (H.F.R.I. ProjectNumber: 15940).

References

1. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. Adv. Neural Inf. Process. Syst. **31** (2018)
2. Bertsimas, D., Orfanoudaki, A., Wiberg, H.: Interpretable clustering: an optimization approach. Mach. Learn. **110**(1), 89–138 (2021)
3. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer (2007)
4. Chasani, P., Likas, A.: Unsupervised decision trees for axis unimodal clustering. Information **15**(11), 704 (2024)
5. Dhurandhar, A., et al.: Explanations based on the missing: towards contrastive explanations with pertinent negatives. Adv. Neural Inf. Process. Syst. **31** (2018)
6. Guidotti, R.: Counterfactual explanations and how to find them: literature review and benchmarking. Data Min. Knowl. Disc. **38**(5), 2770–2824 (2024)
7. Jung, H.G., Kang, S.H., Kim, H.D., Won, D.O., Lee, S.W.: Counterfactual explanation based on gradual construction for deep networks. Pattern Recogn. **132**, 108958 (2022)
8. Kelly, M., Longjohn, R., Nottingham, K.: The UCI machine learning repository. https://archive.ics.uci.edu
9. Laber, E., Murtinho, L., Oliveira, F.: Shallow decision trees for explainable k-means clustering. Pattern Recogn. **137**, 109239 (2023)
10. Le, T., Wang, S., Lee, D.: GRACE: generating concise and informative contrastive sample to explain neural network model’s prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 238–248 (2020)
11. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Adv. Neural Inf. Process. Syst. **30** (2017)
12. Molnar, C.: Interpretable machine learning. Lulu. com (2020)
13. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, pp. 607–617 (2020)
14. Ribeiro, M.T., Singh, S., Guestrin, C.: “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)
15. Schleich, M., Geng, Z., Zhang, Y., Suciu, D.: GeCo: Quality counterfactual explanations in real time. arXiv preprint arXiv:2101.01292 (2021)
16. Sharma, S., Henderson, J., Ghosh, J.: CERTIFAI: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models. arXiv preprint arXiv:1905.07857 (2019)
17. Tavallali, P., Tavallali, P., Singhal, M.: K-means tree: an optimal clustering tree for unsupervised learning. J. Supercomput. **77**(5), 5239–5266 (2021)
18. Van Looveren, A., Klaise, J.: Interpretable counterfactual explanations guided by prototypes. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 650–665. Springer (2021)
19. Vardakas, G., Karra, A., Pitoura, E., Likas, A.: Counterfactual explanations for k-means and gaussian clustering (2025). https://arxiv.org/abs/2501.10234

20. Verma, S., Boonsanong, V., Hoang, M., Hines, K., Dickerson, J., Shah, C.: Counterfactual explanations and algorithmic recourses for machine learning: a review. ACM Comput. Surv. **56**(12), 1–42 (2024)
21. Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harv. JL Tech. **31**, 841 (2017)