Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrics and Operations Research

Advancing cost function design in Distribution Aware Counterfactual Explanation

Mees de Vries (458809)

Supervisor:	Hakan Akyuz
Second assessor:	Name of your second assessor
Date final version:	3rd July 2023

The views stated in this thesis are those of the author and not necessarily those of the supervisor, second assessor, Erasmus School of Economics or Erasmus University Rotterdam.

Abstract

With the rising use of Machine Learning models for decision making, Distribution Aware Counterfactual Explanation can quickly become an important technique for obtaining possible actions to reach a desired prediction result. However, existing implementations simplify this technique, which saves on compute time at the cost of sub-optimal results. In this thesis, two paths for improving these results are explored: altering the calculation of the statistical distance, and the choice of a different outlier detection method. The first path results in promising improvements when applied on real datasets. In Addition, the effects of using statistical distances based on l_p-norms for different p-values are explored and tested.

1 Introduction

Distribution Aware Counterfactual Explanation, or DACE, is quickly becoming more relevant in light of recent development in Artificial Intelligence. It specifically addresses the need for explainable AI (XAI), and Machine Learning Interpretability (ML-I). However, the current implementation is in more than one way a simplification of its theoretical grounds.

Investment in AI and ML has been ever-increasing in recent years and by now has applications in many different fields (Mou, 2019). With applications in professional fields comes the need for understanding and explaining these models to better guide a change in the variables that provide input to the models, as well as for more ethical issues (Bodria et al., 2021). This has lead to the development of many techniques for XAI and ML-I (Karimi, Barthe, Schölkopf & Valera, 2022), among which are many variants of Counterfactual Explanation (CE). In this thesis, the focus will be on Distribution Aware Counterfactual Explanation (DACE) in particular, as established by Kanamori, Takagi, Kobayashi and Arimura (2020).

1.1 Counterfactual Explanation (CE)

The post-hoc method of Counterfactual Explanation (Wachter, Mittelstadt & Russell, 2017) involves explaining individual predictions made by an ML-model. Generally, CE is applied when an input instance results in an undesired output. The aim is to identify the smallest changes in input features that would lead to a more desired prediction outcome. Generating CE's can enhance our insight into how the model makes decisions and understand the impact of input variables on the results.

In an attempt to minimize required change in input features, most variants of CE fail to sufficiently address the achievability of said input changes, as noted by Kanamori et al. (2020). They address the need for regarding feature-correlation and outlier risk among the input features, and introduce the formulation of DACE.

1.2 Distribution Aware Counterfactual Explanation (DACE)

In their paper, a sophisticated cost function is defined, that when minimized, should theoretically solve both the aforementioned problems of CE. To address feature-correlation, Mahalanobis Distance (Mahalanobis, 1936) between the input instance and the adjusted instance is used as statistical distance to be minimized. Also part of the minimization is the Local Outlier Factor (LOF) of the adjusted instance. The LOF (Breunig, Kriegel, Ng & Sander, 2000) is an outlier score using k-Nearest Neighbours to measure the distance of an instance to the sample. A weighted sum of these two metrics forms a cost function that is minimized as a Mixed-Integer Linear Optimization problem (MILO).

This implementation introduces several challenges, however, as it requires a linear formulation of the problem. Both the Mahalanobis-distance and LOF have non-linear definitions, and therefore require linearization. In their implementation, Kentaro et al. replace the theorized cost function with one that uses an l_1-norm based statistical distance. The LOF based on k-NN is fixed at $k = 1$ in order to keep that linear as well. The implementation produces great experimental results in real datasets compared to other CE methods, whilst spurring the question wether a cost function closer to their theoretic base can improve results further, within reasonable compute times.

1.3 Contributions

This thesis aims to answer the question of whether an improved linearization of the theoretic cost function of Distribution Aware Counterfactual Explanation can result in more desirable experimental outcomes. Three main parts make up this discussion:

- A piecewise linearization of the squared Mahalanobis distance is introduced to replace the l_1-norm based statistical distance. This distance-augmented DACE implementation also sparks a debate about using higher order l_p-norm based statistical distances in potential cost functions.
- A consideration of different outlier detection methods leads to a calculation for outlier cost that considers k-Nearest Neighbours for any desired value of k. This outlier-augmented DACE, however, does not strictly provide a linearization of the Local Outlier Factor that the theoretical cost function of DACE suggests.
- Experimental results of DACE are compared to distance-augmented DACE, outlier-augmented DACE and distance-and-outlier-augmented DACE, in order to assess if any improvements can be observed.

2 Notations and Preliminaries

Let X be the sample of N observations. D is the number of features in the model. $\bar{x} \in X$ is an input instance that produces an undesired outcome. $a \in R^D$ is a perturbation vector

representing an action on $\bar{x}$. The covariance matrix of the sample is represented by Σ.
The theoretic cost function for DACE is defined as

$$C_{DACE}(a|\bar{x}) := d_M^2(\bar{x}, \bar{x}+a|\Sigma^{-1}) + \lambda \cdot q_k(\bar{x}+a|X)$$

Where:

- a is a perturbation vector representing an action
- Σ is the covariance matrix of X
- $d_M^2(\bar{x}, \bar{x}+a|\Sigma^{-1})$ is the squared Mahalanobis-distance between the input instance and the instance perturbated by action a, given covariance matrix Σ
- λ is a tuning parameter that balances the influences of the MD and LOF on the cost
- $q_k(\bar{x}+a|X)$ is the k-LOF of the instance with actions applied, given the sample set X.

Additionally we define:

- w_t is the weight of base of the t^{th} base learner
- I^d is the size of the action set for dimension d
- $c_{d,i}^{(n)}$ is the difference in feature d between $\bar{x}+a$ and $x^{(n)}$
- C_n is a sufficiently large constant
- $N_k(x) \subseteq X$ is the set of k Nearest Neighbours of x
- $d_k = max_{x' \in N_k(x)} \Delta(x, x')$ is the distance between x and k^th nearest neighbour
- $rd_k(x, x') = max(\Delta(x, x'), d_k(x'))$ is the k-reachability distance
- $lrd_k()$ is the local reachability density, defined as $lrd_k(x) = k/\sum_{x' \in N_k(x)} rd_k(x, x')$

3 Existing implementation

In order to minimize the cost function of DACE with a MILO-formulation, this cost function needs linearization. Kanamori et al. define a surrogate objective function using an l_1-norm based statistical distance and a 1-NN Local Outlier Factor:

$$\hat{C}_{DACE}(a|\bar{x}) := \hat{d}_M(\bar{x}, \bar{x}+a|\Sigma^{-1}) + \lambda \cdot q_1(\bar{x}+a|X)$$

Where the l_1-norm based statistical distance $\hat{d}_M$ is defined as

$$\hat{d}_M(x, x'|M) = ||U(x'-x)||_1$$

With matrix U such that $U^T U = M$ and $||.||_p$ denoting the l_p-norm. This is a linear function that is easy to model. However, it is substantially different from the squared Mahalanobis distance it is based on:

$$d_M^2(x, x'|M) = ||U(x'-x)||_2^2$$

The full model for Logistic Regression based DACE can be written as:

$$
\begin{aligned}
min \quad & \sum_{d=1}^{D} \delta_d + \lambda \cdot \sum_{n=1}^{N} lrd_1(x^{(n)}) \cdot \rho_n \\
s.t. \quad & \sum_{i=1}^{I_d} \pi_{d,i} = 1 && \forall d \in \{1, ..., D\} && (1) \\
-\delta_d \leq & \sum_{d'=1}^{D} U_{d,d'} \sum_{i=1}^{I_d} a_{d',i} \pi_{d',i} \leq \delta_d && \forall d \in \{1, ..., D\} && (2) \\
& \sum_{n=1}^{N} \nu_n = 1 && && (3) \\
& \sum_{d=1}^{D} \sum_{i=1}^{I_d} (c_{d,i}^{(n)} - c_{d,i}^{(m)}) \cdot \pi_{d,i} \leq C_n \cdot (1 - \nu_n) && \forall n, m \in \{1, ..., N\} && (4) \\
& d_1(x^{(n)}) \cdot \nu_n \leq \rho_n && \forall n \in \{1, ..., N\} && (5) \\
\sum_{d=1}^{D} \sum_{i=1}^{I_d} & c_{d,i}^{(n)} \pi_{d,i} - C_n \cdot (1 - \nu_n) \leq \rho_n && \forall n \in \{1, ..., N\} && (6) \\
& \sum_{t=1}^{T} w_t \xi_t \geq b && && (7) \\
\bar{x}_d + & \sum_{i=1}^{I_d} a_{d,i} \pi_{d,i} = \xi_d && \forall d \in \{1, ..., D\} && (8)
\end{aligned}
$$

In which the following variables are introduced:

$\pi_{d,i}$ binary, $= 1$ if action i on feature d is chosen;

ν_n binary, $= 1$ if instance n is one of the k-Nearest Neighbours;

δ_d the part of the statistical distance between $\bar{x}$ and $\bar{x} + a$ that is on axis d;

ρ_n the reachability distance of n if n is one of the k-Nearest Neighbours;

ξ_t the t^{th} base learner of $\bar{x} + a$;

Constraint (2) defines the statistical distance, and constraints $(3 - 6)$ implement k-NN Local Outlier Factors for $k = 1$.

4 Alternative Models

In this section, possible approaches to improve upon the model of the previous section are proposed. The extensions are confined to the Logistic Regression formulation of DACE.

4.1 distribution-augmented DACE

The first approach is to model a statistical distance that is more similar to the d_M^2 distance. Instead of the l_1-norm based $\hat{d}_{DACE}$, a piecewise linearized squared Mahalanobis distance is

proposed. This is later referred to as distribution-augmented DACE or daDACE.

The model described in the previous section can be adapted by removing constraint (2) and adding constraints (9) and (10).

$$-\tau_d \leq \sum_{d'=1}^{D} U_{d,d'} \sum_{i=1}^{I_d} a_{d',i}\pi_{d',i} \leq \tau_d \qquad \forall d \in \{1,...,D\} \tag{9}$$

$$b_l\tau_d + r_l \leq \delta_d \qquad \forall d \in \{1,...,D\}, l \in \{1,...,L\} \tag{10}$$

With variable τ_d a positive number. The parameter L is the chosen amount of line segments in the linearization, with $\lim_{L\to\infty} \hat{d}_{daDACE} = d_M^2$. Constraint (10) defines the line segments with slope b_l and intercept r_l. Since we are minimizing on a concave function, we can use these unbounded lines instead of bounded line segments. In general, a line segment of an upper estimation of function $f()$ between points a and b can be computed as

$$slope = \frac{f(b) - f(a)}{b - a}$$
$$intercept = f(a) - a \cdot slope$$

If we choose a lower and higher bound to perform the piecewise linearization between (LB and UB, resp.) and spread the L intervals equally between those bounds, the l^{th} interval ranges from $(l-1)\cdot(UB-LB)/L + LB$ to $l\cdot(UB-LB)/L + LB$. With function $f(x) = x^2$, this becomes:

$$b_l = (2l-1)(UB-LB)/L + 2\cdot LB \tag{11}$$

$$r_l = ((l-1)\cdot(UB-LB)/L + LB)^2 - b_l\cdot((l-1)\cdot(UB-LB)/L + LB) \tag{12}$$

This will only add $\mathcal{O}(D)$ variables and constraints to the model. Because this model for $L > 1$ more closely resembles C_{DACE} than the model in section 3, we expect to see better results, however with slightly longer compute times.

4.1.1 Higher order l_p-norms

Now that both an l_1-norm and l_2-norm based approach for statistical distance have been discussed, it can be of interest to evaluate an implementation of l_p-norms for $p > 2$. The effects of a higher value of p would be that dimensions with larger deviations are relatively "punished harder" in the cost function.
We can generalize the computation of b_l and r_l for any value of $p > 0$, resulting in the formulations:

$$b_l = \frac{(l\cdot(UB-LB) + LB\cdot L)^p - ((l-1)\cdot(UB-LB) + LB\cdot L)^p}{(UB-LB)\cdot L^{p-1}} \tag{13}$$

$$r_l = ((l-1)\cdot(UB-LB) + LB\cdot L)^p - b_l\cdot((l-1)\cdot(UB-LB) + LB\cdot L) \tag{14}$$

However, combined with the previously used objective function, this would result in the $\hat{d}_p^p$-distance. When using $p = 2$ as before, this is the desired result. But when $p > 2$, this might completely overshadow the outlier estimation part of the objective function. We can solve this by adding another linearization, however this does further complicate the model. As the goal is to approximate the $\hat{d}_p^2$-distance, we need to linearize $\left(\sum_{d=1}^{D} \delta_d\right)^{2/p}$. As $p > 2$, this is a concave function and therefore we cannot simply add the linear constraints as before. Instead, some more constraints are needed to ensure the line segments don't affect the objective beyond their bounds. Variables $\phi \in R^+$ and $\chi_l \in \{0, 1\}$ are introduced to assist in this step, as well as sufficiently large number M. We can then adapt the model to:

$$min \quad \phi + \lambda \cdot \sum_{n=1}^{N} lrd_1(x^{(n)}) \cdot \rho_n$$

$$s.t. \quad \sum_{l=1}^{L'} \chi_l = 1 \tag{15}$$

$$l \cdot (UB' - LB')/L + LB' + M(1 - \chi_l) \geq \sum_{d=1}^{D} \delta_d \qquad \forall l \in \{1, ..., L'\} \tag{16}$$

$$(l-1) \cdot (UB' - LB')/L + LB' - M(1 - \chi_l) \leq \sum_{d=1}^{D} \delta_d \qquad \forall l \in \{1, ..., L'\} \tag{17}$$

$$b'_l \cdot \sum_{d=1}^{D} \delta_d + r'_l - M(1 - \chi_l) \leq \phi \qquad \forall l \in \{1, ..., L'\} \tag{18}$$

Combined with constraints (1) and $(3 - 10)$. Note that LB' and UB' are the lower and upper bound of this second linearization, and L' is the amount of line segments in the second linearization. These are not necessarily the same as LB, UB and L from the first linearization.

4.1.2 l_{inf}-norm based distance

If the goal is to minimize the single-dimension statistical distance, an l_{inf}-norm based statistical distance can be used. This d_{inf}^2 can be implemented using using constraints (1) and $(3 - 8)$, combined with the following objective function and constraints:

$$min \quad \delta + \lambda \cdot \sum_{n=1}^{N} lrd_1(x^{(n)}) \cdot \rho_n$$

$$s.t. \quad -\tau \leq \sum_{d'=1}^{D} U_{d,d'} \sum_{i=1}^{I_d} a_{d',i} \pi_{d',i} \leq \tau \qquad \forall d \in \{1, ..., D\} \tag{19}$$

$$b_l \tau + r_l \leq \delta \qquad \forall l \in \{1, ..., L\} \tag{20}$$

Where τ and δ no longer have subscript d, since only the largest value is of interest.

4.2 outlier-augmented DACE

Another approach is to find an outlier detection method that uses k-Nearest Neighbours for $k > 1$ whilst being linear. The Local Outlier Factor used as the outlier detection method in the definition of C_{DACE} does not lend itself well for linearization. In recent years however, many other outlier detection methods have been developed (Boukerche, Zheng & Alfandi, 2020). Some linear outlier detection methods exist that can be implemented without further need for linearization, such as the method by Angiulli and Pizzuti (2002) that uses the sum of distances to the k nearest points in the sample. However, this method might not be the best option for our model, as we are aiming to find local outliers, not global outliers. According to Aggarwal (2015), local density needs to be taken into account in order to reliably detect local outliers.

As an outlier-augmented DACE (oaDACE) implementation, a local outlier detection method can be used that combines Angiulli's method for global outliers with the local density, by dividing the sum of distances to the k-Nearest Neighbours by the average k-NN distance sum of those k Neighbours.

$$\hat{q}_k(x|X) = \left(\sum_{j=1}^{k} c_{d,i}^{(n)} \right) \frac{k}{\sum_{m=1}^{k} \sum_{n=1}^{k} \Delta_d(x_d^{(n)}, x_d^{(n)(m)})}$$

where x^n is the instance in the sample that is the n-th closest instance to x. The fraction that makes up the right part of the equation above can be pre-calculated for all N instances, making linear implementation in our model possible. This fraction will be referred to as $oald_k(x^{(n)})$, standing for oa-local density.

We can now model oaDACE by adapting the model described in section 3. Constraints (1), (2), (7) and (8) will stay unchanged. Instead of Constraints $(3-6)$, we introduce constraints $(21-23)$:

$$\sum_{n=1}^{N} \nu_n = k \tag{21}$$

$$\sum_{d=1}^{D} \sum_{i=1}^{I_d} (c_{d,i}^{(n)} - c_{d,i}^{(m)}) \cdot \pi_{d,i} \leq C_n(1 - \nu_n + \nu_m) \qquad \forall n, m \tag{22}$$

$$\sum_{d=1}^{D} \sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} - C_n(1 - \nu_n) \leq \rho_n \qquad \forall n \tag{23}$$

Furthermore, in the objective function, we replace the $lrd_1(x^{(n)})$ with $oald_k(x^{(n)})$ to obtain the model for oaDACE. This model doesn't add time complexity compared to the model described in section 3. However, because a different outlier detection method is used, it is hard to predict if it will improve results, and if results will be consistently better across different datasets.

4.3 distribution-and-outlier-augmented DACE

After considering both alterations separately, they can be combined into a single model:

$$min \quad \sum_{d=1}^{D} \delta_d + \lambda \cdot \sum_{n=1}^{N} oald_k(x^{(n)}) \cdot \rho_n$$

$$s.t. \quad \sum_{i=1}^{I_d} \pi_{d,i} = 1 \qquad \forall d \in \{1, ..., D\} \tag{1}$$

$$-\tau_d \leq \sum_{d'=1}^{D} U_{d,d'} \sum_{i=1}^{I_d} a_{d',i}\pi_{d',i} \leq \tau_d \qquad \forall d \in \{1, ..., D\} \tag{9}$$

$$b_l\tau_d + r_l \leq \delta_d \qquad \forall d \in \{1, ..., D\}, l \in \{1, ..., L\} \tag{10}$$

$$\sum_{n=1}^{N} \nu_n = k \tag{13}$$

$$\sum_{d=1}^{D}\sum_{i=1}^{I_d}(c_{d,i}^{(n)} - c_{d,i}^{(m)}) \cdot \pi_{d,i} \leq C_n \cdot (1 - \nu_n + \nu_m) \quad \forall n, m \in \{1, ..., N\} \tag{14}$$

$$\sum_{d=1}^{D}\sum_{i=1}^{I_d} c_{d,i}^{(n)}\pi_{d,i} - C_n \cdot (1 - \nu_n) \leq \rho_n \qquad \forall n \in \{1, ..., N\} \tag{15}$$

$$\sum_{t=1}^{T} w_t\xi_t \geq b \tag{7}$$

$$\bar{x}_d + \sum_{i=1}^{I_d} a_{d,i}\pi_{d,i} = \xi_d \qquad \forall d \in \{1, ..., D\} \tag{8}$$

Where we now have 3 parameters instead of 1: λ, k and L. Note that setting $L = 1$ gives the oaDACE model, while setting $k = 1$ results in a model that is very similar to but not entirely equivalent to the daDACE model. Having 3 parameters can be a disadvantage, as they might need to be adjusted for every dataset.

5 Results

For the analysis of daDACE, oaDACE and doaDACE, two datasets were used. One that contains German consumer credit data, where the expectation whether a customer will default is the output of the Machine Learning model, and $D = 61$ variables are present. The other dataset is referred to as FICO and contains data on Home Equity Line of Credit, where we are again interested in the risk of default. This model contains $D = 23$ variables. The code is adapted from the code for DACE by Kanamori et al., written in Python 3.7 and using IBM CPLEX as optimizer. On all models, an extra constraint is applied: $||a||_0 \leq 3$.

In order to compare the models, we evaluate them on their results values of the theoretical C_{DACE}. That is, The actual Mahalanobis Distance between the input instance and the instance with actions applied is calculated, as well as the k-Local Outlier Factor with k=10. For both of these measures, we prefer lower values.

5.1 Reproduction

Before evaluating the extensions, we first look at the reproduction of the experiments from Kanamori et al. For both datasets, the results of DACE are compared to a cost function based on a weighted inverse of Median Absolute Deviation (MAD) (Russell, 2019), as well as one based on the Total Log-Percentile Shift (TLPS) (Ustun, Spangher & Liu, 2019), and a cost function based the Pearson's correlation coefficients (PCC) (Ballet et al., 2019). All methods are applied both on a Logistic Regression model as discussed so far, as well as a Random Forest model. The results can be found in tables 1 and 2.

	LR			RF		
Model	d_M	10-LOF	Time (s)	d_M	10-LOF	Time (s)
MAD	8.21 ± 6.31	1.27 ± 0.38	0.095 ± 0.019	2.38 ± 1.49	1.37 ± 0.66	3.56 ± 5.70
TLPS	6.85 ± 4.59	1.35 ± 0.42	0.079 ± 0.015	2.29 ± 1.46	1.37 ± 0.66	4.39 ± 6.67
PCC	11.82 ± 7.32	1.32 ± 0.48	0.072 ± 0.021	3.53 ± 3.43	1.33 ± 0.44	5.03 ± 5.72
DACE	3.59 ± 2.77	1.29 ± 0.28	40.48 ± 15.53	2.30 ± 1.46	1.30 ± 0.62	34.66 ± 17.45

Table 1: FICO dataset, 100 instances, $\lambda = 1$

	LR			RF		
Model	d_M	10-LOF	Time (s)	d_M	10-LOF	Time (s)
MAD	6.96 ± 3.92	1.28 ± 0.72	0.064 ± 0.025	3.20 ± 3.02	1.33 ± 0.75	5.70 ± 3.51
TLPS	2.75 ± 1.32	1.43 ± 0.56	0.061 ± 0.013	1.45 ± 1.50	1.15 ± 0.46	8.25 ± 7.87
PCC	8.04 ± 2.54	1.28 ± 0.72	0.058 ± 0.024	6.07 ± 2.74	1.34 ± 0.75	2.20 ± 1.63
DACE	2.32 ± 1.25	1.21 ± 0.22	3.23 ± 1.29	1.82 ± 1.39	1.04 ± 0.17	28.47 ± 20.07

Table 2: German dataset, 228 instances, $\lambda = 0.01$

5.2 daDACE

We confine the evaluation of the extensions to the Logistic Regression models. For 228 instances of the German dataset and 100 instances of the FICO dataset, results from DACE and daDACE models are compared in table 3. For these runs, we set $L = 40, LB = 0, UB = 10$. Both models were ran with $\lambda = 0.01$ and $\lambda = 0$. In all experiments, daDACE resulted in a lower average d_M, with a similar 10-LOF value. This did consistently come at a cost in run time, however that difference was smaller than perhaps expected. These results are in line with the expectation that daDACE would result in lower MD-values.

5.3 Different l_p-norm based daDACE

To compare DACE implementations with l_p-norm based statistical distances for different values of p, 93 instances of the German dataset were ran using the the l_1-norm (DACE), l_2-norm (daDACE), l_3-norm (3-daDACE), l_5-norm (5-daDACE) and l_{inf}-norm (inf-daDACE). Besides the average Mahalanobis-distance (d_M), 10-LOF and run time, we now also report the maximum single dimensional statistical distance (max-d). The results can be found in table 4. As expected,

for higher values of p, the d_M is larger, but the value of max-d is lower. The average value of max-d for inf-daDACE is higher than that of 5-daDACE though. However, the median value is much lower, and in 54 instances inf-daDACE does have a lower max-d value, compared to 25 instances the other way around.

	German			FICO		
Model	d_M	10-LOF	Time (s)	d_M	10-LOF	Time (s)
DACE	1.97 ± 1.32	1.21 ± 0.24	2.98 ± 1.34	3.59 ± 2.77	1.29 ± 0.28	40.4 ± 15.5
daDACE	1.91 ± 1.25	1.19 ± 0.23	3.95 ± 1.47	3.09 ± 1.89	1.36 ± 0.44	39.1 ± 18.8
DACE$_{\lambda 0}$	1.72 ± 1.27	1.27 ± 0.44	0.59 ± 0.12	3.33 ± 2.22	1.41 ± 0.61	1.90 ± 0.60
daDACE$_{\lambda 0}$	1.62 ± 1.15	1.28 ± 0.41	1.68 ± 0.67	2.86 ± 1.92	1.50 ± 0.54	6.86 ± 3.08

Table 3: daDACE on German and FICO datasets, with $\lambda = 0.01$ and $\lambda = 0$

Model	d_M	$max - d$	10-LOF	Time (s)
DACE	1.81 ± 1.21	1.24 ± 0.85	1.24 ± 0.27	3.30 ± 1.23
daDACE	1.77 ± 1.15	1.12 ± 0.73	1.20 ± 0.23	4.25 ± 1.29
3-daDACE	1.79 ± 1.13	1.03 ± 0.62	1.20 ± 0.21	12.07 ± 7.53
5-daDACE	1.79 ± 1.09	0.93 ± 0.47	1.19 ± 0.22	8.74 ± 6.12
inf-daDACE	2.25 ± 1.51	1.01 ± 0.70	1.24 ± 0.92	3.33 ± 1.28

Table 4: daDACE with different l_p-norms for 93 instances of the German dataset

Model	MD	10-LOF	Time (s)
DACE	1.97 ± 1.32	1.21 ± 0.25	2.98 ± 1.32
daDACE	1.91 ± 1.25	1.19 ± 0.23	3.95 ± 1.25
oaDACE$_{k=5}$	1.72 ± 1.25	1.24 ± 0.44	3.67 ± 2.11
oaDACE$_{k=10}$	1.71 ± 1.24	1.23 ± 0.39	3.59 ± 2.10
doaDACE	1.68 ± 1.24	1.23 ± 0.36	6.97 ± 1.72

Table 5: oaDACE and doaDACE for 228 instances of the German dataset

5.4 oaDACE and doaDACE

oaDACE was ran with both $k = 5$ and $k = 10$, with almost no difference in run time between the two. A lower value for the 10-LOF at a similar value for MD compared to DACE was expected, however the results show the opposite.

for doaDACE, we see similar results. A k-value of 10 and $L = 40$ was used to run the same 228 instances with the doaDACE-model. We can observe a lower MD than the oaDACE-model in 122 instances, compared to only 30 instances with a higher score. However, the differences are almost negligible. Both the oaDACE and doaDACE models do not succeed in providing significant improvements over the DACE and daDACE models.

6 Conclusion

In this thesis, an existing implementation of Distribution Aware Counterfactual Explanation (DACE) was evaluated that uses a MILO-formulation. Multiple possible improvements were

proposed and tested.

With distribution-augmented DACE (daDACE), a piecewise linearized quadratic Mahalanobis distance was implemented to replace the l_1-norm based statistical distance in the cost function of the previous implementation. This resulted in some promising improvements when applied to real datasets, consistently lowering the observed Mahalanobis distance of the experiments at the cost of slight increases in run time.

With outlier-augmented DACE (oaDACE), a different oulier detection method was implemented to replace the 1-Nearest Neighbour based Local Outlier Factor of the previous implementation. The idea was to implement a simpler outlier detection method that could more easily be implemented in a MILO-formulation for k-Nearest Neighbours with $k > 1$. This failed to produce a significant improvement over the previous implementation. However, as not all existing outlier detection methods were tried, this might be of interest for future work.

Also considered were the effects of using a l_p-norm based statistical distance for $p > 2$. An implementation using two linearization steps is formulated. The l_2-norm minimizes the Mahalanobis distance of the proposed action, the l-infinity norm minimizes the dimension with the largest distance. By choosing a value of p between that, one can balance the importance of the two effects. The implementation succeeded in producing those desired results in a run time that takes two to three times as long as the daDACE-implementation.

References

Aggarwal, C. C. (2015). Outlier analysis: Advanced concepts. In *Data mining: The textbook* (pp. 265–283). Cham: Springer International Publishing.

Angiulli, F. & Pizzuti, C. (2002). Fast outlier detection in high dimensional spaces. In *Principles of data mining and knowledge discovery: 6th european conference, pkdd 2002 helsinki, finland, august 19–23, 2002 proceedings 6* (pp. 15–27).

Ballet, V., Renard, X., Aigrain, J., Laugel, T., Frossard, P. & Detyniecki, M. (2019). Imperceptible adversarial attacks on tabular data. *arXiv preprint arXiv:1911.03274*.

Bodria, F., Giannotti, F., Guidotti, R., Naretto, F., Pedreschi, D. & Rinzivillo, S. (2021). Benchmarking and survey of explanation methods for black box models. *arXiv preprint arXiv:2102.13076*.

Boukerche, A., Zheng, L. & Alfandi, O. (2020). Outlier detection: Methods, models, and classification. *ACM Computing Surveys (CSUR)*, *53*(3), 1–37.

Breunig, M. M., Kriegel, H.-P., Ng, R. T. & Sander, J. (2000). Lof: identifying density-based local outliers. In *Proceedings of the 2000 acm sigmod international conference on management of data* (pp. 93–104).

Kanamori, K., Takagi, T., Kobayashi, K. & Arimura, H. (2020). Dace: Distribution-aware counterfactual explanation by mixed-integer linear optimization. In *Ijcai* (pp. 2855–2862).

Karimi, A.-H., Barthe, G., Schölkopf, B. & Valera, I. (2022). A survey of algorithmic recourse: contrastive explanations and consequential recommendations. *ACM Computing Surveys*, *55*(5), 1–29.

Mahalanobis, P. (1936). Mahalanobis distance. In *Proceedings national institute of science of india* (Vol. 49, pp. 234–256).

Mou, X. (2019). Artificial intelligence: investment trends and selected industry uses. *International Finance Corporation*, *8*.

Russell, C. (2019). Efficient search for diverse coherent explanations. In *Proceedings of the conference on fairness, accountability, and transparency* (pp. 20–28).

Ustun, B., Spangher, A. & Liu, Y. (2019). Actionable recourse in linear classification. In *Proceedings of the conference on fairness, accountability, and transparency* (pp. 10–19).

Wachter, S., Mittelstadt, B. & Russell, C. (2017). Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, *31*, 841.

A Programming code

```
This code is based on the code from Kanamori et al.

## aggregator.py
To execute, run aggregator.py. The amount of instances to run,
the dataset to use and formulation type can be set on lines 8--10
of this script. Other parameters, such as lambda, L and p, can be
set per formulation, as parameters of the "extract" functions.

The script will run every instance for all formulations mentioned
in the thesis. After completion, the results are outputted again,
but this time in CSV-format for use with other data analysis
software.

## linear_ce.py
Contains all formulations and calls the CPLEX-solver for the
Logistic Regression formulations. Contains the models for all
extensions.

## forest_ce.py
Contains formulations for Random Forest formultaions. Only used
for replication, does not contain implementations for daDACE,
p-daDACE, oaDACE or doaDACE.

## utils.py
Contains most functions, including calculations of constants and
distributions. Also contains classes for actions.

Datasets are to be stored in a folder called "data" in the root
of the project.
```