Learning from Revealed Algorithmic Recourse Preferences

Rory Creedon
University College London
Department of Computer Science

Submitted to University College London (UCL) in partial fulfilment of the requirements for the award of the degree of

Master of Science in Data Science and Machine Learning.

Industry supervisor: **Colin Rowat**

Academic supervisor: **Matthew Caldwell**

Submission date: **11$^{\text{th}}$ September 2023**

Abstract

- Most works in algorithmic recourse/strategic classification assume a simple, pre-specified cost function for changing feature values.
- Understanding *individual* cost functions is important for generating recourse and understanding *global* cost functions is important for strategic classification.
- Whilst there has been research into generating individual recourse through preference elicitation, there has not been research into learning *global* cost functions.
- Learning algorithms are proposed to learn cost function from the users' revealed preferences - their responses to a series of pairwise comparisons of different recourse options.
- The algorithms are evaluated on synthetic and semi-synthetic data.
- Recourse costs are compared for users with different protected attributes, showing if learning costs functions aids or exacerbates fairness of recourse.

Contents

1 Introduction

Introduction chapter.

2 Literature Review

2.1 Algorithmic Recourse

2.1.1 Motivation

Description of what algorithmic recourse is and why it is important - use of automatic decision making, GDPR (Voigt and Bussche, 2017). Mention psychological factors causing humans to prefer recourse to explanations (**to find paper(s)**: was mentioned by Ruth Byrne in ICML panel session, from 25 minutes onwards).

2.1.2 Problem Set-up

- Description of the original set-up and problem - i.e.,

$$\begin{aligned} \boldsymbol{\delta}^* = & \operatorname*{argmin}_{\boldsymbol{\delta}} c(\mathbf{x}, \boldsymbol{\delta}) \\ \text{s.t. } & h(\mathbf{x}') = 1, \\ & \mathbf{x}' = \mathbf{x} + \boldsymbol{\delta}, \\ & \boldsymbol{\delta} \in \mathcal{F} \end{aligned} \tag{2.1.1}$$

- Description of the causal recourse set-up and problem (Karimi, Schölkopf, et al., 2021)
- Cost and distance functions, actionability of features

2.1.3 Recourse methods

Run through methods mentioned in survey paper (Karimi, Barthe, et al., 2022) and also those implemented in `CARLA`.

2.2 Strategic Classification

2.2.1 Standard Strategic Classification

- Begin with Hardt et al. (2016) and explain the set-up as a Stackelberg game with an example.
- Algorithms proposed for this task include Levanon and Rosenfeld (2021), Chen et al. (2020) and Ahmadi et al. (2022).
- Mention extensions such as:
- Where the cost function is completely unknown to the lender (Dong et al., 2018)
- Where the response of lenders to the classifier is noisy (Jagadeesan et al., 2021).
- Where borrowers do not know the decision rule (Ghalme et al., 2021; Bechavod et al., 2022).

- Where the incentives of lender and borrower align (e.g., recommender systems) (Levanon and Rosenfeld, 2022).
- Where the cost functions are linked by graphs for the borrowers (Eilat et al., 2023).
- Where the borrowers act first (Nair et al., 2022).
- Where the borrowers and lenders update at different rates (Zrnic et al., 2021).

2.2.2 Causal Strategic Classification

A review of the *causal* strategic classification literature, which focuses more on causal identification of features which are strategically manipulated (without causing an improvement in underlying credit 'worthiness') and features which causally affect credit 'worthiness'.

2.3 Revealed Preferences

A brief primer on axioms of revealed preferences, and on the literature of *learning from revealed preferences*. To briefly discuss:

- Original paper by Beigman and Vohra (2006), where principal issues a list of prices and the agent purchases different quantities of each good. Over time, the principal learns from the different purchase amounts (which are the revealed preferences).
- When prices are of goods and budget of the agent are drawn from an unknown distribution (Zadimoghaddam and Roth, 2012; Balcan et al., 2014).
- Where the principal is maximising profit (Amin et al., 2015; Roth et al., 2016).
- Move onto a more detailed discussion of Dong et al. (2018).

2.4 Pairwise Metric Learning

- Start with an introduction of what pairwise metric learning is and key papers.
- Move onto specific proposed adaptation/simplification of the learning algorithm proposed in Canal et al. (2022).

2.5 Canonical Datasets

The canonical datasets used in the algorithmic recourse and strategic classification literature include:

- `Adult` - dataset to predict whether someone earns over $50,000 or more.
- `German Credit` - dataset identifies people as either good or bad credit risks.
- `FICO-HELOC` - dataset of HELOC applications, where applicants have applied for a credit line between $5,000 and $150,000. Outcome variable is whether they are a good or bad credit risk.
- `Finance` - dataset to predict financial distress for a number of companies. There are several over different time periods for each company.

3 Causal Algorithmic Recourse

As discussed in section 2, the cost of changing features from $\mathbf{x}$ to $\mathbf{x}'$ in the strategic classification literature is typically a quadratic cost function of the form $c(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^2$, or occasionally a quadratic form cost function $c(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{M}(\mathbf{x} - \mathbf{x}')$ where $\mathbf{M}$ is a fixed, known, positive semi-definite square matrix (Bechavod et al., 2022).

Add citations

However, these do not necessarily represent the true complexities of the cost of moving from $\mathbf{x}$ to $\mathbf{x}'$, for a number of (non-exhaustive) reasons. Consider the case of an individual applying for a line of credit.

1. **Changing one feature can change the cost of changing another feature**. If an individual decides not to inquire about a loan for a number of months (which will change the feature "number of inquiries in the last 6 months", the cost of decreasing the feature "number of inquiries in the last 6 months, excluding the last 7 days" will be very low or zero. However, if a quadratic cost function (or any L_p norm cost function) is used, this will be interpreted as two separate feature changes and the costs of each will be summed. Whilst this simple case can likely be handled by domain expertise, more complex causal relations will exist. Consider an individual obtaining two more credit cards. Whilst this may reduce the cost of increasing "number of credit cards", this may also increase the cost of "monthly credit card payments" and may have less clear effects (which need not be linear) on other features.
2. **Changing feature costs can be different for different individuals**. For example, increasing the number of credit cards from 1 to 5 may be much easier for someone with a higher income or increasing income from £25,000 to £30,000 may be much easier for someone with a higher level of education. These are all modelled as the same in typical cost functions used in the literature.

To address through clustering, or add in note saying that this is out of scope.

3.1 Structural Causal Models

To address the first issue, that changing one feature can change the cost of changing another feature, we must model the effect of changing the feature $\mathbf{x}_1$ to $\mathbf{x'}_1$ has on changing the feature $\mathbf{x}_2$ to $\mathbf{x'}_2$. Following Karimi, Schölkopf, et al. (2021), we can model the world using a structural causal model (SCM) to account for downstream effects. The SCM can be defined formally as M $\mathcal{M} = \langle \mathbb{U}, \mathbb{V}, \mathbb{F} \rangle$ capture all the causal relations in the world, where $\mathbb{U}$ represents set of exogenous variables, $\mathbb{V}$ represents the set of endogenous variables (all are a descendant of at least one of the variables in $\mathbb{U}$) and $\mathbb{F}$ represents the set of structural equations which describe how the endogenous variables can be determined from the exogenous variables (Pearl et al., 2016).

We can illustrate a simple SCM with a four variable SCM $\mathcal{M}$ with $\mathbb{U} = \{x_1\}$, $\mathbb{V} = \{x_2, y\}$ and structural equations $\mathbb{F}$ are defined in equation 3.1.1, where σ is the sigmoid function.

$$
\begin{aligned}
x_1 &= u_1; \quad u_1 \sim N(30,000, 5,000) \\
x_2 &= u_2 + 0.5x_1; \quad u_2 \sim N(20,000, 10,000) \\
y &= \sigma\left(\frac{x_1 + x_2 - 60,000}{10,000}\right)
\end{aligned} \tag{3.1.1}
$$

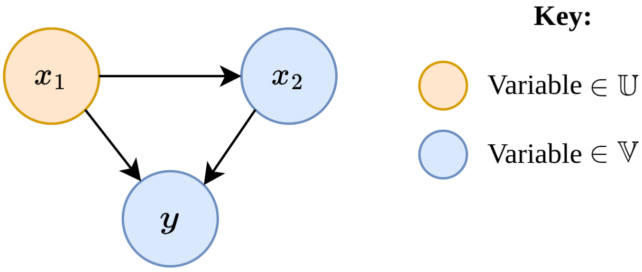

Figure 3.1: Causal graph $\mathcal{G}$ of SCM $\mathcal{M}$.

The SCM can also be presented with a causal graph $\mathcal{G}$, which is shown in Figure 3.1. Let x_1 represent salary, x_2 represent savings and y represent the probability of individual being approved for a mortgage. If an individual for whom the SCM $\mathcal{M}$ holds receives an increase in salary, this will lead to an increase in savings, and both the increase in savings and income will increase the probability of mortgage approval. If the individual's initial features were [£30,000, £25,000]^T and their salary increases to £35,000, then it should then be easier to increase their savings than when their salary was £30,000. This toy example shows how the SCM $\mathcal{M}$ encodes the downstream effect of increased salary on both savings and probability of mortgage approval.

We can use the Abduction-Action-Prediction steps (Pearl et al., 2016) to obtain a *structural counterfactual* of an increase in salary (x_1) from £30,000 to £35,000, given that the existing features are [£30,000, £25,000, 0.27]^T

1. Abduction. Calculate the value of exogenous variables before intervention, given the evidence (the current values of the features).

$$
\begin{aligned}
u_1 &= x_1 = 30,000 \\
u_2 &= x_2 - 0.5x_1 = 25,000 - 0.5(30,00) = 10,000 \\
u_3 &= 0
\end{aligned} \tag{3.1.2}
$$

2. Action. Modify the model $\mathcal{M}$ by implementing the intervention on x_1 (i.e; $do(x_1 = 35,000)$). This leads to a new SCM M_1 where all incoming edges to the intervened upon variable x_1 are severed and the value of the intervened upon variable x_1 is set to the intervention value £35,000. In this case, there are no incoming edges to x_1, so $\mathcal{M} = \mathcal{M}_1$.

$$
\begin{aligned}
x_1 &= 35,000 \\
x_2 &= u_2 - 0.5x_1 \\
x_3 &= \sigma\left(\frac{x_1 + x_2 - 60,000}{10,000}\right)
\end{aligned} \tag{3.1.3}
$$

3. Prediction. Using the updated model $\mathcal{M}_1$ and values of exogenous variables $\mathbf{u}$, calculate the values of the endogenous variables.

$$\begin{aligned}
x_1^{\text{SCF}} &= 35,000 \\
x_2^{\text{SCF}} &= 0.5x_1^{\text{SCF}} + u_2 = 0.5(35,000) + 10,000 = 27,500 \\
y^{\text{SCF}} &= \sigma\left(\frac{x_1^{\text{SCF}} + x_2^{\text{SCF}} - 60,000}{10,000}\right) = \sigma\left(\frac{35,000 + 27,500 - 60,000}{10,000}\right) = 0.56
\end{aligned} \tag{3.1.4}$$

Mathematically, we can denote the Action-Abduction-Prediction steps as shown in equation 3.1.5, where I is the set of indices of intervened on variables, δ_i is the action on variable i, $f_i \in \mathbb{F}$ is structural equation of the variable i, pa_i are the parents of variable i and [`condition`] is 1 if the condition is true and 0 is it is not.

Introduce δ_i notation

$$\begin{aligned}
x_i^{\text{SCF}} =& [i \in I](x_i + \delta_i) \\
&+ [i \notin I]\Big(x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)\Big)
\end{aligned} \tag{3.1.5}$$

To calculate recourse $\mathbf{x}^*$ with an SCM over all the features within $\mathbf{x}$, we need to reformulate the original algorithmic recourse optimisation problem presented in equation 2.1.1. Following Karimi, Schölkopf, et al. (2021), we replace the minimising the cost of changing features from $\mathbf{x}$ to $\mathbf{x}'$ with minimising the cost of *interventions* $A = \text{do}\{\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D}$ where D is the number of features. The updated recourse problem is shown in equation 3.1.6.

To consider whether better to use a_i notation instead of $x_i + \delta_i$

$$\begin{aligned}
A^* =& \operatorname*{argmin}_A \texttt{cost}(\mathbf{x}, A) \\
\text{s.t. } & h(\mathbf{x}^{\text{SCF}}) = 1, \\
& \mathbf{x}_i^{\text{SCF}} = [i \in I](\mathbf{x}_i + \boldsymbol{\delta}_i) + [i \notin I]\Big(x_i + f_i(\mathbf{pa}_i^{\text{SCF}}) - f_i(\mathbf{pa}_i)\Big), \\
& A \in \mathcal{F}
\end{aligned} \tag{3.1.6}$$

We define the cost of interventions A on original features $\mathbf{x}$ as the sum of the cost of the individual actions. The cost of each individual action is left to be flexible, and can represent a variety of cost functions, such as the ℓ_p-norm of $\boldsymbol{\delta}_i$ or a percentile shift based cost function such as that used by Ustun et al. (2019).

$$\texttt{cost}(\mathbf{x}, A) = \sum_{i=1}^{D} c\Big(\text{do}(\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i)\Big) \tag{3.1.7}$$

This formulation relies on two key assumptions.

Assumption 1. The interventions are structural (hard) interventions, where after intervening on a variable, all incoming edges to its corresponding node in the causal graph are severed. If an individual were to intervene on savings (x_2) (perhaps by selling their car or borrowing from family), then we assume that they then stop saving a proportion of their income (severing the edge between x_1 and x_2).

Assumption 2. Interventions on multiple variables are carried out simultaneously. Given an intervention A = [£32,000, £27,500]^T, it is assumed that salary as increased at the same time as savings, as opposed to taking place sequentially.

3.1.1 Soft (Parametric) Interventions

In many cases where recourse is provided, such as credit scoring, it is unlikely that intervening on a variable leads to all incoming edges to its corresponding node in the causal graph being severed - a violation of assumption 1. Intervening on savings is unlikely to lead to an individual stopping saving their salary. Likewise, intervening on body fat levels through liposuction does not lead to diet having no causal effect on body fat levels.

Explain when hard interventions are used - i.e., RCTs/-experiments

In these cases, we can then represent the interventions as soft (or parametric) interventions, which do not result in severing of incoming edges (Eberhardt and Scheines, 2007). We can represent the resulting value of x^{SCF} in equation 3.1.8. Compared to hard (structural) interventions in equation 3.1.5, the formula for x_i^{SCF} is the same for variables that are not intervened upon, and the effects of incoming edges are present for intervened upon variables through $f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)$.

$$x_i^{\text{SCF}} = [i \in I]\delta_i + \Big(x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)\Big) \tag{3.1.8}$$

As the majority of interventions which take place in the context of algorithmic recourse, such as increasing savings and re-taking an test such as the GMAT/GRE (in the case of recourse for postgraduate admissions) do not tend to result in the severing of incoming edges such as the proportion of income saved and the effect of additional revision on test scores, soft interventions are implemented in this thesis. Using soft interventions results in an updated causal recourse problem shown below in equation 3.1.9.

$$\begin{aligned} A^* = & \operatorname*{argmin}_A \texttt{cost}(\mathbf{x}, A) \qquad (3.1.9)\\ \text{s.t. } & h(\mathbf{x}^{\text{SCF}}) = 1, \\ & \mathbf{x}^{\text{SCF}} = [i \in I]\boldsymbol{\delta}_i + \Big(\mathbf{x}_i + f_i(\mathbf{pa}_i^{\text{SCF}}) - f_i(\mathbf{pa}_i)\Big), \\ & A \in \mathcal{F} \end{aligned}$$

3.1.2 Sequential Interventions

The second assumption of the causal recourse problem formulation in 3.1.5 is that all interventions occur simultaneously. Picture a scenario where an individual is rejected from a PhD program, and the recourse interventions are to gain more research experience (potentially through a pre-doctoral fellowship or research assistant position) and obtain a more favourable letter of recommendation. In the real world, it is likely that these actions will be carried out sequentially, where research experience is obtained first and the letter of recommendation is second (as the professor for whom the applicant is conducting their research under will likely be the author of the letter of recommendation), as opposed to occurring simultaneously.

Using equation 3.1.8, the order of the intervention does not affect the counterfactual values x_i^{SCF}, but can affect the cost of actions A.

Potentially worth showing how $f_i(\text{pa}_i^{\text{SCF}})$ in one intervention is cancelled out by $f_i(\text{pa}_i)$ in the next intervention

Proposition 1. In a sequential intervention setting with n separate interventions where x_i and x_j are intervened upon, if the cost of individual interventions $c(\text{do}(x_i := x_i + \delta_i))$

depends on the value of x_i and x_i is a descendent of x_j, then the ordering of the sequential interventions affects the total cost of the n sequential interventions.

Proof. The values of x_i after an intervention on x_i and intervening on x_j are shown below.

$$x_i^{\text{SCF}_i} = x_i + \delta_i \tag{3.1.10}$$

$$x_i^{\text{SCF}_j} = x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i) \tag{3.1.11}$$

As the cost of individual interventions depends on the value of x_i before intervention, the cost of intervening on x_i first depends on the value x_i whereas the cost of intervening on x_i second depends on the value $x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)$ (the value after intervening on x_j, seen in equation 3.1.11). This results in different costs for the intervention on x_i for the different orderings.

As x_j is a descendant of x_i, the intervention on x_i has no effect on $x_j^{\text{SCF}_i}$ and both intervening on x_j first or second leads to the same cost for the intervention on x_j.

As the total cost is the sum of the costs of each intervention and the costs for the interventions on x_i are different for each ordering and the intervention on x_j are the same for each ordering, then the total cost for the two orderings of sequential interventions are different. □

Make this more maths-y and potentially add a worked example

To take into account the potential effects of different orderings on the costs, we denote interventions as an ordered set $A = \{(\mathbf{S}, do\{\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i\}_{i=1,\ldots,D})\}$ where $\mathbf{S}$ is a permutation of the set $\{1, \ldots, D\}^N$ and represents the ordering of the intervention. Given this updated definition of A, the causal recourse formulation stays the same as shown in 3.1.9.

To replace o_i with a permutation set?

3.2 Differentiable Sorting

In order to solve equation 3.1.9 with sequential interventions, we need to optimise for an ordering of interventions. With D features, there are $D!$ different orderings (i.e. permutations of the set $\{1, \ldots, D\}$) and equation 3.1.9 becomes a combinatorial optimisation problem.

Is this NP-hard?

I think this is true, to check

In order to transform the combinatorial optimisation problem to a continuous optimisation problem, we can first define a vector $O \in \mathbb{R}^D$, which can be optimised using continuous heuristics such as gradient descent. From O, we can recover S through the transformation $S = \texttt{argsort}(O)$.

Not sure if this para is clear enough - trying to say we make a vector O from which we can reconstruct the ordering S. The vector $O \in \mathbb{R}^D$ can be optimised by gradient descent, so we optimise O and therefore indirectly optimise the ordering S

However, the operation `argsort` is not differentiable. As a solution, we replace the `argsort` operator with the `SoftSort` operator, a continuous relaxation of the `argsort` operator (Prillo and Eisenschlos, 2020).

Is it worth explaining this or just take as given?

For a given permutation (i.e., ordering) $\pi \in \{1, \ldots, D\}^D$, we can also express the permutation π as a permutation matrix $P_\pi \in \{0,1\}^{D \times D}$. We can represent P_π mathematically as a square matrix with values as shown in equation 3.2.1. For example, the permutation matrix of the permutation $\pi = [3, 1, 2]^T$ is shown in equation 3.2.2.

$$P_\pi[i,j] = \begin{cases} 1 & \text{if } j = \pi_i \\ 0 & \text{otherwise} \end{cases} \tag{3.2.1}$$

$$\pi = [3, 1, 2]^T \implies P_\pi = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.2.2}$$

`SoftSort` defines a continuous relaxation for $P_{\texttt{argsort}(O)}$, defined in equation 3.2.3, where d is a differentiable (almost everywhere) semi-metric function and $\tau > 0$ is a temperature parameter that controls the degree of approximation. For the experiments in this thesis, $d(x, y) = |x - y|$ has been used.

$$\texttt{SoftSort}^d_\tau(O) = \texttt{softmax}\left(\frac{-d(\texttt{sort}(O)\mathbb{1}^T, \mathbb{1}O^T)}{\tau}\right) \tag{3.2.3}$$

The value of the semi-metric function d is larger when $\texttt{sort}(O)[i]$ is close to $O[j]$ and smaller when $\texttt{sort}(O)[i]$ is far from $O[j]$. The `softmax` function is applied row-wise, meaning that the larger the value of semi-metric function d compared to other values, the larger the value of $\texttt{SoftSort}[i, j]$. A larger temperature parameter $\tau > 0$ leads to the values of d moving closer together and, after the `softmax`, the values of $\texttt{SoftSort}[i, j]$ become more evenly distributed, compared to the true $P_{\texttt{argsort}(O)}$, which is binary (i.e., very unevenly distributed). Therefore, the larger the value of τ, the more approximate `SoftSort` becomes. As $\tau \to 0$, $\texttt{SoftSort}^d_\tau(O) \to P_{\texttt{argsort}(O)}$.

A visual representation can be seen below of $P_{\texttt{argsort}(O)}$ and $\texttt{SoftSort}^{|\cdot|}_1(O)$ can be seen below for $O = [2, 5, 4]^T$.

$$O = \begin{bmatrix} 2 \\ 5 \\ 4 \end{bmatrix} \implies P_{\texttt{argsort}(O)} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{3.2.4}$$

$$\texttt{SoftSort}^{|\cdot|}_1(O) = \texttt{softmax}\left(- \begin{bmatrix} |5-2| & |5-5| & |5-4| \\ |4-2| & |4-5| & |4-4| \\ |2-2| & |2-5| & |2-4| \end{bmatrix} \right) = \begin{bmatrix} 0.04 & \mathbf{0.70} & 0.26 \\ 0.09 & 0.24 & \mathbf{0.67} \\ \mathbf{0.85} & 0.04 & 0.11 \end{bmatrix} \tag{3.2.5}$$

As `SoftSort` is the combination of the `softmax` function (which is differentiable), the semi-metric d (which is differentiable almost everywhere) and the `sort` function (which is differentiable almost everywhere), this leads to `SoftSort` being differentiable (almost everywhere).

The values of the matrix that `SoftSort` produces, such as in equation 3.2.5, can be interpreted *loosely* as the probability that the i^{th} element of $\pi_{\texttt{argsort}(O)}$ is j.

When incorporating `SoftSort` into the the causal recourse problem defined in equation 3.1.9, we take the maximum 'probability' in each row as the ordering S, as opposed to weighting the costs of differing orderings by their 'probabilities'.

To work out if and why this is correct - was previously having a problem were converged to O was [0.25, 0.25, 0.25, 0.25] - although potentially because in a setting where ordering didn't matter

==

We denote the interventions as a set $\mathcal{A} = \{(a_1, o_1), (a_2, o_2), \ldots, (a_D, o_D)\}$ where (a_i, o_i) represents that the $o_i{}^{\text{th}}$ intervention will be an intervention of a_i on $\mathbf{x}'_i$, the current value of feature i after any previous interventions. The intervention may have downstream effects on other features, as described by the causal graph.

The causal graph can be expressed as a weighted adjacency matrix W, where W_{ij} is the marginal effect on $\mathbf{x}_j$ of intervening on $\mathbf{x}_i$. Different orderings can result in different values of $\mathbf{x}'$.

Using the same causal graph in Figure 3.1 and weighted adjacency W

The cost of the interventions $\mathcal{A}$ is therefore the sum of a cost function over each individual intervention a_i (for example $c(a_i) = a_i^2$). The cost of the intervention can be expressed as follows, where D is the number of features.

$$c(\mathcal{A}) = \sum_{i=1}^{D} c(\mathbf{a}_i) \tag{3.2.6}$$

The post-intervention features $\mathbf{a}^*$ can be calculated as shown in Algorithm 1.

Algorithm 1 Intervention Evaluation Function

function EVALINTERVENTIONS$(\mathbf{a}, \mathbf{x}, \mathbf{o}, W)$
    Input: size of actions for each feature $\mathbf{a}$, original features $\mathbf{x}$, order of actions $\mathbf{o}$, adjacency matrix W
    Output: new features $\mathbf{x}^*$
    $\mathbf{x}^* \leftarrow \mathbf{x}$
    $W \leftarrow W + I$ ▷ where I is the identity matrix
    $\mathbf{s} \leftarrow \texttt{argsort}(\mathbf{o})$
    for i in $\mathbf{s}$ **do** ▷ Loop through each feature in order of action
        $\mathbf{x}^* \leftarrow \mathbf{x}^* + \mathbf{a}_i \times W[:, i]^T$ ▷ Update for downstream effects
    end for
    return $\mathbf{x}^*$
end function

3.3 Generating recourse

Using the new formulation of the cost function as the cost of interventions, we can also re-write the recourse generation problem, given a classifier h, original features x and weighted adjacency matrix W. We denote the intervention evaluation as described in Algorithm 1 as `eval`. For negatively classified individuals, the task of recourse involves solving equation 3.3.1, where we minimise the cost of intervention subject to the new features $\mathbf{x}'$ leading to a positive classification (which corresponds to a classification score of greater than 0.5).

$$\begin{aligned} \mathbf{a}', \mathbf{o}' = & \operatorname*{argmin}_{\mathbf{a},\mathbf{o}} \sum_{i=1}^{D} c(\mathbf{a}_i) \\ & \text{s.t. } \; h(\texttt{eval}(\mathbf{a}, \mathbf{x}, \mathbf{o}, W)) \geq 0.5 \end{aligned} \tag{3.3.1}$$

As `eval` is a non-convex function, we cannot solve this constrained optimisation problem using convex optimisation. Instead, is converted into a unconstrained problem using Lagrange multipliers as shown in equation 3.3.2. This is solved using gradient descent, where at each iteration, a step is first taken to maximise λ and then a step is then taken to minimise $\mathbf{a}$ and $\mathbf{o}$.

$$\min_{\mathbf{a},\mathbf{o}} \max_{\lambda} \sum_{i=1}^{D} c(\mathbf{a}_i) - \lambda(h(\texttt{eval}(\mathbf{a}, \mathbf{x}, \mathbf{o}, W)) - 0.5) \tag{3.3.2}$$

This expression is possible to optimise using gradient descent when only optimising for $\mathbf{a}$, the function `eval` contains the line $S \leftarrow \texttt{argsort}(\mathbf{o})$, which is non-differentiable. Therefore, when optimising for the ordering $\mathbf{o}$, we must find an alternative to the `argsort` operator.

3.3.1 Differentiable sorting

The key point to make in this section is that technically, as long as a function is differentiable and maps from $f : \mathbb{R}^D \rightarrow \texttt{ordering}^D$, we should be able to take derivatives of f and use gradient descent to find the value of the input to f which minimises the objective defined in equation 3.3.2. The smoother the function is in its mapping from a vector of real numbers to the ordering (i.e., vectors close to each other map to similar orderings), the fewer local minima it should have, making optimisation easier.

4 Cost Learning

4.1 Learning from revealed preferences

We do not observe the cost function itself, but one way to approximate it is to *learn from revealed preferences* (see section 2.3). We propose that each individual who is negatively classified is presented with N pairs of recourse options $((\mathbf{a}_n^1, \mathbf{o}_n^1), (\mathbf{a}_n^2, \mathbf{o}_n^2))$. Each of the recourse options corresponds to a list of actions and associated orderings. The individuals, for each of the N pairs of recourse options, then select which one is preferable.

Need to rephrase actions as the total action (even if you get part of it for 'free'.

If, for a single pair of recourse options $((\mathbf{a}_n^1, \mathbf{o}_n^1), (\mathbf{a}_n^2, \mathbf{o}_n^2))$, option 1 is selected, then we assume that $\sum_{i=1}^D c(\mathbf{a}_{ni}^1) \leq \sum_{i=1}^D c(\mathbf{a}_{ni}^2)$. If option 2 is selected, we assume the opposite, that $\sum_{i=1}^D c(\mathbf{a}_{ni}^1) > \sum_{i=1}^D c(\mathbf{a}_{ni}^2)$. The responses to the pairs of recourse options presented (the pairwise comparisons) reveal information about the individuals' preferences over recourse options, i.e., their cost functions over individual features.

User preference over cost functions involves knowledge of the causal graph, should this then be optimised for in learning from revealed preferences.)

Once a cost function is learned, we need to solve the optimisation problem mentioned in equation 3.3.2 to generate the recourse $(\mathbf{a}', \mathbf{o}')$.

4.1.1 Weighted Squared Costs

Let the individual cost function take the form $c(\mathbf{a}, \beta) = \sum_{i=1}^D \beta_i \mathbf{a}_i^2$, where $\beta \in \mathbb{R}^D$ is a vector which expresses the mutability of each feature i. To learn the cost function, we need to learn β.

Given a fixed weighted adjacency matrix W, we can denote the response of the nth paired comparison as follows.

Seems much more efficient to assume a fixed ordering

$$y_n = \begin{cases} -1 & \text{if } \sum_{i=1}^D c(\mathbf{a}_{ni}^1) \leq \sum_{i=1}^D c(\mathbf{a}_{ni}^2) \\ +1 & \text{if } \sum_{i=1}^D c(\mathbf{a}_{ni}^1) > \sum_{i=1}^D c(\mathbf{a}_{ni}^2) \end{cases} \tag{4.1.1}$$

We optimise for when y_n and $\hat{y}_n$ (the predicted value of y_n, using our initial guess of β) are similar. A optimisation to do this is shown below, where there are K individuals and N pairwise comparisons and $\ell(y, \hat{y}) = \max[0, 1 - y\hat{y}]$ represents the hinge loss.

$$\underset{\beta}{\operatorname{argmin}} = \frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \ell(y_{kn}, \hat{y}_{kn}(\beta)) + \underbrace{\lambda ||\beta||_2}_{\text{L2 regularisation}} \tag{4.1.2}$$

This is an unconstrained optimised problem that can be optimised via gradient descent. However, operation in equation 4.1.1 is non, differentiable, so instead it is approximated with the below expression, which fits $\hat{y}_n$ into [-1,1] where λ is a hyperparameter regularising for 'confidence' of the predictions.

$$\hat{y}_n = \tanh\left(\lambda\Big(\sum_{i=1}^{D} c(\mathbf{a}_{ni}^1) - \sum_{i=1}^{D} c(\mathbf{a}_{ni}^2)\Big)\right) \tag{4.1.3}$$

4.2 Mahalanobis distance

The Mahalanobis distance between the vector $\mathbf{x}$ and the vector $\mathbf{y}$ is defined in equation 4.2.1, where M is a positive semi-definite matrix.

$$||\mathbf{x}-\mathbf{y}||_{\mathbf{M}}=\sqrt{(\mathbf{x}-\mathbf{y})^T\mathbf{M}^{-1}(\mathbf{x}-\mathbf{y})} \tag{4.2.1}$$

The matrix $\mathbf{M}$ captures different relationships between the features within $\mathbf{x}$ and $\mathbf{y}$ in the off-diagonal elements of $\mathbf{M}$. If $\mathbf{M}$ is set to the identity matrix, then the Mahalanobis distance then becomes equal to the Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$.

4.2.1 Learning the Mahalanobis distance

In order to use the Mahalanobis distance as a cost function, we must learn the matrix $\mathbf{M}$. In this set-up, each individual k with original features $\mathbf{x}_k$ is presented with N recourse options $(\mathbf{x}^a_{kn}, \mathbf{x}^b_{kn})$. The response y_{kn} is defined in equation 4.2.2, where c^G_k represents the ground truth cost function of individual k.

$$y_{kn}=\begin{cases}-1 & \text{if } c^G_k(\mathbf{x}_{kn},\mathbf{x}^a_{kn})\leq c^G_k(\mathbf{x}_{kn},\mathbf{x}^b_{kn})\\ +1 & \text{if } c^G_k(\mathbf{x}_{kn},\mathbf{x}^a_{kn})> c^G_k(\mathbf{x}_{kn},\mathbf{x}^b_{kn})\end{cases} \tag{4.2.2}$$

To optimise for $\mathbf{M}$, we compare the squared Mahalanobis distances between $\mathbf{x}_k$ and $\mathbf{x}^a_{kn}$ and between $\mathbf{x}_k$ and $\mathbf{x}^b_{kn}$. The optimisation problem to learn $\mathbf{M}$ is shown in equation 4.2.3, where ℓ represents either the hinge or logistic loss function. The optimisation is an adaptation of the optimisation problem presented in Canal et al. (2022).

[**TO ADD EXPLANATION ON WHY THIS IS A GOOD OPTIMISATION**]

$$\begin{aligned}&\min_{\mathbf{M}}\frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\ell\Big(y_{kn}(||\mathbf{x}_k-\mathbf{x}^a_{kn}||^2_{\mathbf{M}}-||\mathbf{x}_k-\mathbf{x}^b_{kn}||^2_{\mathbf{M}})\Big)\\&\text{s.t. }\ \mathbf{M}\succeq 0,\end{aligned} \tag{4.2.3}$$

4.3 Convex layers

To look into convex neural networks using `cvxpylayers`, which is based on Agrawal et al. (2019).

5 Experiments

5.1 Synthetic data

The experiments that follow use synthetic data generated from a structural causal model, which is shown in Figure 5.1 .

To simplify, removing x_4, y and $\hat{y}$.

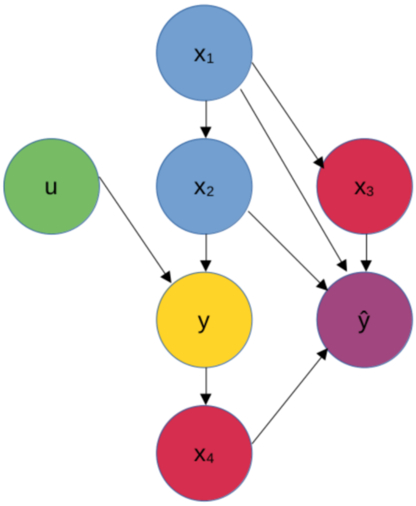

Figure 5.1: The Structural Causal Model used for synthetic data generation

The structural causal model contains 2 unobserved variables

- $\mathbf{u}$ - which is sampled from a normal distribution
- $\mathbf{y}$ - the true binary outcome which is a linear combination of $\mathbf{u}$ and $\mathbf{x}_2$

And 4 observed variables

- $\mathbf{x}_1$ - which is sampled from a normal distribution
- $\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ - which are linear combinations of other variables
- $\hat{\mathbf{y}}$ - The predicted value of $\mathbf{y}$

5.2 Simulation process

[TO CONVERT TO AN ALGORITHM/PSEUDOCODE]

The process for simulation is as follows. We assume that the individuals do *not* have any knowledge of the classifier.

1. Split the data into test and train sets.
2. Fit a classifier using just the train set and measure accuracy against the *true* labels.
3. Predict labels for *all* data points (both train and test)

4. Calculate recourse actions for all negatively classified data points, by minimising the current approximation of the cost function.

5. Perturb the recourse actions to create pairwise comparisons for the negatively classified individuals to evaluate.

6. Learn cost function from evaluated pairwise comparisons from *all* previous iterations.

7. Generate updated recourse with the current approximation of the cost function.

8. Calculate the 'ground truth cost' of the recourse with learned cost.

5.3 Mahalanobis distance

References

Agrawal, Akshay et al. (2019). "Differentiable Convex Optimization Layers". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 9558–9570. URL: https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html (Cited on page 15).

Ahmadi, Saba et al. (2022). "On Classification of Strategic Agents Who Can Both Game and Improve". In: *3rd Symposium on Foundations of Responsible Computing (FORC 2022)*. Vol. 218. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 3:1–3:22. DOI: 10.4230/LIPIcs.FORC.2022.3 (Cited on page 4).

Amin, Kareem et al. (2015). "Online Learning and Profit Maximization from Revealed Preferences". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, pp. 770–776. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9984 (Cited on page 5).

Balcan, Maria-Florina et al. (2014). "Learning Economic Parameters from Revealed Preferences". In: *Web and Internet Economics - 10th International Conference, WINE 2014, Beijing, China, December 14-17, 2014. Proceedings*. Vol. 8877. Springer, pp. 338–353. DOI: 10.1007/978-3-319-13129-0_28 (Cited on page 5).

Bechavod, Yahav et al. (2022). "Information Discrepancy in Strategic Learning". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162. PMLR, pp. 1691–1715. URL: https://proceedings.mlr.press/v162/bechavod22a.html (Cited on pages 4, 6).

Beigman, Eyal and Rakesh Vohra (2006). "Learning from Revealed Preference". In: *Proceedings of the 7th ACM Conference on Electronic Commerce*. New York, NY, USA: Association for Computing Machinery, pp. 36–42. DOI: 10.1145/1134707.1134712 (Cited on page 5).

Canal, Gregory et al. (2022). "One for All: Simultaneous Metric and Preference Learning over Multiple Users". In: *Advances in Neural Information Processing Systems* 35, pp. 4943–4956. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/1fd4367793bcd3ad38a0b820fcc1b815-Abstract-Conference.html (Cited on pages 5, 15).

Chen, Yiling, Yang Liu, and Chara Podimata (2020). "Learning Strategy-Aware Linear Classifiers". In: *Advances in Neural Information Processing Systems 33*. URL: https://proceedings.neurips.cc/paper/2020/hash/ae87a54e183c075c494c4d397d126a66-Abstract.html (Cited on page 4).

Dong, Jinshuo et al. (2018). "Strategic Classification from Revealed Preferences". In: *Proceedings of the 2018 ACM Conference on Economics and Computation*. Ithaca, NY, USA, pp. 55–70. DOI: 10.1145/3219166.3219193 (Cited on pages 4, 5).

Eberhardt, Frederick and Richard Scheines (2007). "Interventions and Causal Inference". In: *Philosophy of Science* 74.5, pp. 981–995. DOI: 10.1086/525638 (Cited on page 9).

Eilat, Itay et al. (2023). "Strategic Classification with Graph Neural Networks". In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. URL: https://openreview.net/pdf?id=TuHkVOjSAR (Cited on page 5).

Ghalme, Ganesh et al. (2021). "Strategic Classification in the Dark". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 3672–3681. URL: http://proceedings.mlr.press/v139/ghalme21a.html (Cited on page 4).

Hardt, Moritz et al. (2016). "Strategic Classification". In: *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science* (Cambridge, Massachusetts, USA). New York, NY, USA: Association for Computing Machinery, pp. 111–122. DOI: 10.1145/2840728.2840730 (Cited on page 4).

Jagadeesan, Meena, Celestine Mendler-Dünner, and Moritz Hardt (2021). "Alternative Microfoundations for Strategic Classification". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 4687–4697. URL: http://proceedings.mlr.press/v139/jagadeesan21a.html (Cited on page 4).

Karimi, Amir-Hossein, Gilles Barthe, et al. (2022). "A Survey of Algorithmic Recourse: Contrastive Explanations and Consequential Recommendations". In: *ACM Comput. Surv.* 55.5. DOI: 10.1145/3527848 (Cited on page 4).

Karimi, Amir-Hossein, Bernhard Schölkopf, and Isabel Valera (2021). "Algorithmic Recourse: From Counterfactual Explanations to Interventions". In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada). New York, NY, USA, pp. 353–362. DOI: 10.1145/3442188.3445899 (Cited on pages 4, 6, 8).

Levanon, Sagi and Nir Rosenfeld (2021). "Strategic Classification Made Practical". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 6243–6253. URL: http://proceedings.mlr.press/v139/levanon21a.html (Cited on page 4).

Levanon, Sagi and Nir Rosenfeld (2022). "Generalized Strategic Classification and the Case of Aligned Incentives". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162. PMLR, pp. 12593–12618. URL: https://proceedings.mlr.press/v162/levanon22a.html (Cited on page 5).

Nair, Vineet et al. (2022). "Strategic Representation". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162.

PMLR, pp. 16331–16352. URL: https://proceedings.mlr.press/v162/nair22a.html (Cited on page 5).

Pearl, J., M. Glymour, and N.P. Jewell (2016). *Causal Inference in Statistics: A Primer*. Wiley (Cited on pages 6, 7).

Prillo, Sebastian and Julian Martin Eisenschlos (2020). "SoftSort: A Continuous Relaxation for the Argsort Operator". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. PMLR, pp. 7793–7802. URL: http://proceedings.mlr.press/v119/prillo20a.html (Cited on page 10).

Roth, Aaron, Jonathan R. Ullman, and Zhiwei Steven Wu (2016). "Watch and Learn: Optimizing from Revealed Preferences Feedback". In: *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. ACM, pp. 949–962. DOI: 10.1145/2897518.2897579 (Cited on page 5).

Ustun, Berk, Alexander Spangher, and Yang Liu (2019). "Actionable Recourse in Linear Classification". In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. New York, NY, USA: Association for Computing Machinery, pp. 10–19. DOI: 10.1145/3287560.3287566 (Cited on page 8).

Voigt, Paul and Axel von dem Bussche (2017). *The EU General Data Protection Regulation (GDPR): A Practical Guide*. 1st ed. Springer Publishing Company, Incorporated. DOI: 10.1007/978-3-319-57959-7 (Cited on page 4).

Zadimoghaddam, Morteza and Aaron Roth (2012). "Efficiently Learning from Revealed Preference". In: *Internet and Network Economics - 8th International Workshop, WINE 2012, Liverpool, UK, December 10-12, 2012. Proceedings*. Vol. 7695. Springer, pp. 114–127. DOI: 10.1007/978-3-642-35311-6_9 (Cited on page 5).

Zrnic, Tijana et al. (2021). "Who Leads and Who Follows in Strategic Classification?" In: *Advances in Neural Information Processing Systems 34*, pp. 15257–15269. URL: https://proceedings.neurips.cc/paper/2021/hash/812214fb8e7066bfa6e32c626c2c688b-Abstract.html (Cited on page 5).