Causal Algorithmic Recourse from Revealed Preferences

Rory Creedon
University College London
Department of Computer Science

Submitted to University College London 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: **11th September 2023**

Abstract

- Most works in algorithmic recourse 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

Picture a scenario where, after renting for the last 10 years, you are in the process of buying your first home. You require a mortgage of £300,000. Your salary is £50,000, you have savings of £40,000 (excluding that saved for the deposit) and a credit score of 800[1]. You decide to apply for a mortgage with a new digital bank, which promises to instantly provide a decision, through its 'AI-powered system'. After entering all your details online, you press 'Submit'. A new screen appears, informing you that your mortgage application has been unsuccessful. However, it also informs you that if you can increase your income to £70,000 and increase your savings to £50,000 , you will be approved for the mortgage. These actions you can take in order for your mortgage to be approved are known as *algorithmic recourse*.

Algorithmic or automated decision making systems are widely used in the real world, often in high stakes environments, where they have a significant impact on the lives of individuals. Some examples include credit scoring, where classifiers are widely used to approve loans and mortgages (O'Dwyer, 2018), in criminal justice, to assess the risk of re-offending (Angwin et al., 2016) and hiring, where candidate screening and video interviews are often automated (Kramer, 2022). Algorithmic recourse refers to a set of actions an individual can take to remedy a negative decision made by an algorithmic or automated decision making system. For example, in the mortgage approval case, the recourse provided was to increase income and credit score. In a hiring setting, it may involve increasing educational achievements or work experience.

cite this

Whilst algorithmic decision making is highly prevalent, algorithmic recourse is not as common. This is concerning, given that recourse has benefits such as increasing trust in algorithmic systems and aiding individuals' ability to make plans and decisions over time (Venkatasubramanian and Alfano, 2020). Moreover, algorithmic recourse has been argued to have a legal basis in the EU's General Data Protection Regulation (GDPR), where individuals have a *right to recourse* (Voigt and Bussche, 2017).

cite this

check this, not sure if this is true - written in survey paper

Generating algorithmic recourse involves finding a feasible alternative set of actions $\mathbf{x}'$ that results in a minimal cost to the individual, subject to a positive outcome from the classifier h. Mathematically, we can formulate the algorithmic recourse problem as follows, where $\mathbf{x}$ are the individual's original features, the classifier is assumed to be a binary classifier $h : \mathbb{R}^D \to \{0, 1\}$ and $\mathcal{F}$ represents the set of feasible features values (e.g, if x_{RACE} = "Black", then x'_{RACE} = "White" is not feasible).

$$\begin{aligned} \mathbf{x}^* = &\underset{\mathbf{x}'}{\operatorname{argmin}}\, \texttt{cost}(\mathbf{x}, \mathbf{x}') \\ \text{s.t. } &h(\mathbf{x}') = 1, \\ &\mathbf{x}' \in \mathcal{F} \end{aligned} \tag{1.0.1}$$

[1]Assume that this is an Experian credit score, where a score 800 corresponds to a 'fair' score, see more details here.

These alternative sets of features are often referred to as *counterfactual explanations* - counterfactual features that would have resulted in a positive or more favourable outcome[2]. There have been various different methods proposed to generate counterfactual explanations, which can accommodate different types of classifiers (e.g., linear models, tree-based models, neural networks), which have different feasibility, plausibility and other constraints, which can be used on different types of data (e.g., tabular data, image data) and can be computed using different methods (e.g., gradient descent, integer linear programming, brute force) (see Table 1 in Karimi et al. (2022)).

could more citations here, but survey paper should do fine

One of the key challenges in providing algorithmic recourse is the cost function itself. Given that humans are not able to express their individual costs/preferences mathematically, it a highly non-trivial task to estimate the cost of changing features $\mathbf{x}$ to $\mathbf{x}'$. Estimating the cost function is crucial to providing algorithmic recourse, as without a good understanding of the cost of changing features, the recourse provided could be very costly and difficult to achieve. Again consider the scenario where you have unsuccessfully applied for a mortgage on your first home. Imagine that a highly inaccurate cost function has been used to generate recourse and you have been told to increase your income from £50,000 to £80,000 and to increase your savings from £40,000 to £42,500. This may be highly costly, as it is difficult to obtain a 60% increase in salary, whilst increasing your savings by £2,500 may be comparatively much easier. Another set of features on the classification boundary, is to increase your income to £65,000 and to increase your savings to £55,000. You would consider the second set of features much less costly to achieve than the first. However, due to the poor estimation of your true cost function, you have been provided with recourse that is difficult to achieve.

cite and explain why this is

In this thesis, we highlight and propose solutions for three key issues with existing cost functions in the algorithmic recourse literature.

1. A set of actions (e.g, increase income to £70,000, increase savings to £50,000) are typically enacted sequentially, as opposed to simultaneously. Costs should consider the order of these actions, as actions (or *interventions*) have downstream (causal) effects on other variables. For example, after obtaining a raise and increasing your income to £70,000, it now becomes much easier to increase your savings to £50,000. This is addressed in chapter 3.

2. The cost function should take into account user preferences over feature mutability. For example, picture a scenario where an individual is applying for a PhD and is provided with recourse. They are asked to increase their GRE[3] quantitative reasoning score and produce more academic work (i.e., published papers). It is likely that increasing your GRE quantitative score is more easily mutable than producing additional published papers, which take considerable time and effort. This is addressed in chapter 4, where a novel human-in-the-loop approach is proposed to learn user preferences.

3. Causal algorithmic recourse requires specification of the underlying Structural Causal Model (SCM). When we intervene on income (by increasing it to £70,000), we use the SCM to evaluate the downstream (causal) effects, such as leading to an increase

[2]This thesis will focus on binary classification problems where there is a positive outcome and a negative outcome. However, the problem naturally extends to multi-class classification, where there are different counterfactual explanations for each class.

[3]The Graduate Record Examination (GRE) is a standardised test that is an admissions requirement for some Masters and PhD programmes.

in savings. The size and form of this effect are dictated by the SCM. It could be a simple linear relationship (such as 40% of income is savings, so an increase in income of £20,000 leads to an increase in income of £8,000) or a more complex relationship that includes many non-linear and non-convex functions. An incorrectly specified SCM can lead to an incorrect cost function. For the actions (a) increase income to £70,000 and then (b) increase savings to £50,000, if we believe that in the SCM, there is a linear relationship between income and savings where an increase in income of £1 leads to a £0.40 increase in savings, then we assume that after the increase in income, savings are £8,000 higher (original income is £50,000). However, the true relationship is linear, where an increase in income of £1 leads to a £0.20 increase in savings. This means after the increase in come, savings are only £4,000 higher. The true cost of two actions are higher than the estimated cost in the our estimation of the SCM. This problem is also addressed in chapter 4, where human-in-the-loop approach learns an *linear approximation* of the SCM as well as user preferences.

The structure of the thesis is as follows. Relevant literature is reviewed in chapter 2, causal algorithmic recourse and sequential interventions are discussed in chapter 3, the methodology proposed to learn costs is discussed in chapter 4, results of and discussion of experiments on synthetic data are in chapter 5 and concluding remarks are made in chapter 6.

The accompanying code to replicate the experiments in the thesis can be found here: `https://github.com/rorycreedon/msc_thesis`.

2 Literature Review

Algorithmic recourse was first defined in the machine learning literature as "the ability of a person to obtain a desired outcome from a fixed model" (Ustun et al., 2019). In our example from the introduction, where you are declined a mortgage, the digital bank provides recourse in the form of an alternative set of features (also referred to in the literature as 'flipsets'). Should you change your features to the alternative set of features (where your income is £70,000 and savings are £40,000), then the mortgage will be approved (a positive outcome). In this example, the digital bank's mortgage approval classifier is fixed - the alternative set of features will not result in the mortgage being declined again when you re-apply.

2.1 Problem Formulation

The algorithmic recourse problem can be defined formally as shown in equation 2.1.1, where an individuals' original features are $\mathbf{x}$, $h : \mathbb{R}^D \to \{0, 1\}$ is the classifier and $\mathcal{F}$ is the set of feasible features values. The set of feasible feature values constrains $\mathbf{x}'$ by only allowing positive/integer/similar constraints values of $\mathbf{x}'_i$ where relevant (e.g, number of credit cards must be either 0 or a positive integer, credit score must be between 0 and 999[1]). For *immutable* variables (e.g., race, birthplace, etc.) it must be that the new feature value is the same as the original, that is $\mathbf{x}_i = \mathbf{x}'_i$.

$$\begin{aligned} \mathbf{x}^* = &\underset{\mathbf{x}'}{\operatorname{argmin}}\, \texttt{cost}(\mathbf{x}, \mathbf{x}') \\ \text{s.t. } &h(\mathbf{x}') = 1, \\ &\mathbf{x}' \in \mathcal{F} \end{aligned} \tag{2.1.1}$$

2.2 Cost Functions

Typically, the cost function is either of the form $L_p(\mathbf{x}' - \mathbf{x}')$, with the L_1 and L_2 norm being the most common (Karimi et al., 2022; Ramakrishnan et al., 2020). For the L_1 and L_2 norms, this cost function is always greater than or equal to 0, and is minimised when $\mathbf{x} = \mathbf{x}'$. The further away from the original features $\mathbf{x}$ that the counterfactual values $\mathbf{x}'$ are, the higher the cost. Intuitively, this means that leaving the features unchanged will result in a cost of 0, whilst significant changes (either positive or negative) will occur a cost that increases with the size of the changes.

Another function prevalent in the algorithmic recourse literature is the total log percentile shift (Ustun et al., 2019), shown in equation 2.2.1, where D is the number of features and Q_i represents the CDF of $\mathbf{x}_i$. This cost function also considers the cost of each feature changed independently. It punishes increasing from the percentile $Q_i(\mathbf{x}'_i)$ is larger than the original percentile $Q_i(\mathbf{x}_i)$. A key advantage of this cost function is that changes become harder when starting from a higher percentile, e.g, moving from the 50^{th} to 55^{th}

[1] Experian credit scores run from 0-999, see link here.

percentile carries a cost of 0.105, where as moving from the 90$^{\text{th}}$ to 95$^{\text{th}}$ percentile carries a cost of 0.693. This is likely to reflect reality more than the same cost for increasing percentile by 5 percentage points. Whilst the cost is 0 when $\mathbf{x}_i = \mathbf{x}'_i$, it becomes negative when $Q_i(\mathbf{x}'_i) < Q_i(\mathbf{x}_i)$. Therefore, for this cost function to applied correctly, it requires a monotonic constraint (e.g. increasing income is positively associated with credit score). improve this

$$\texttt{cost}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{D} \log\left(\frac{1 - Q_i(\mathbf{x}_i)}{1 - Q_i(\mathbf{x}'_i)}\right) \tag{2.2.1}$$

A related branch of literature, *strategic classification*, studies the effect of the behaviour of strategic agents on classifiers. Individuals strategically manipulate their features in order to gain a favourable outcome, as opposed to increasing the underlying variable being classified, for example 'credit-worthiness' in a credit scoring setting or practical skills relevant to a specific job in the hiring setting. Designing strategy-robust classifiers or classifiers that incentivise improvement requires a cost function of changing features from $\mathbf{x}$ to $\mathbf{x}'$. Whilst the L_1 and L_2 norms are prevalent in the strategic classification literature, Bechavod et al. (2022) also consider the Mahalanobis distance. A Mahalanobis distance (or quadratic form) cost function is shown in equation 2.2.2, where $\mathbf{M}$ is a positive semi-definite matrix.

$$\texttt{cost}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}') \tag{2.2.2}$$

As well as allowing for different relative costs of changing features independently (along the diagonal), a Mahalanobis-based cost function allows changing the value of one feature to change the cost of changing another feature. A worked example is shown below. Let $\mathbf{x} = [2, 3, 4]^T$ and $\mathbf{x}' = [1, 1, 1]^T$. First, note that $(\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^2$ if $\mathbf{M}$ is the identity matrix. If the values along the diagonal of $\mathbf{M}$ were different, this would encode different costs for changing each feature.

$$[1, 2, 3]^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1, 2, 3]^T = 1^2 + 2^2 + 3^2 = 14 \tag{2.2.3}$$

When the diagonals are non-zero, this results changing one feature affects the cost of changing another feature. See the below example, where changing $\mathbf{x}_1$ leads to an decreased cost in changing $\mathbf{x}_2$.

$$[1, 2, 3]^T \begin{bmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1, 2, 3]^T = 1^2 + 1.5(2) + 3^2 = 13 \tag{2.2.4}$$

The matrix $\mathbf{M}$ must be manually specified, meaning that the principal (the provide of recourse) must estimate $\mathbf{M}$ before providing recourse.

2.2.1 Learned Cost Functions

The main contribution of this thesis is a methodology to *learn* cost functions. There are two main works that used learned cost functions, as opposed to the manually specified cost functions in the previous sections.

The first is Rawal and Lakkaraju (2020), who propose to learn and incorporate how mutable each feature by presenting pairwise feature comparisons to experts on the subject matter. The experts are presented with two features, feature i and feature j and are asked respond with which feature is more easily mutable. A Bradley-Terry model is used to calculate how mutable each feature is. How mutable features i and j are denoted as β_i and β_j. The probability that an expert will respond with feature i is more mutable than feature j can be calculated as shown below.

$$p_{ij} = \frac{\exp(\beta_i)}{\exp(\beta_i) + \exp(\beta_j)} \tag{2.2.5}$$

From the responses of the experts to the pairwise comparisons, p_{ij} is directly computed as follows.

$$p_{ij} = \frac{\text{number of } i > j \text{ responses}}{\text{total number of } i, j \text{ responses}} \tag{2.2.6}$$

As p_{ij} can be computed, the feature mutability β_i and β_j are then learned by MAP estimation, using a minorisation-maximisation algorithm (Caron and Doucet, 2012).

Whilst this approach does learn a measure of how mutable each feature is, it does not lead to an *individualised* measure of how mutable each feature is. Picture two individuals who are applying for a loan, one of which has high discretionary spending, low irreducible costs (e.g., childcare, rent, debt repayments, etc.) and another with low discretionary spending and high irreducible costs. It likely that is (relatively) much easier for the first individual to increase their savings (for example by decreasing discretionary spending) than the second individual. To provide recourse to individuals that is as low cost as possible, it is important to also take into account *individual* preferences over feature mutability.

This problem is addressed by De Toni et al. (2023), who proposed PEAR, a methodology to provide individualised recourse. Similar to the methodology in this thesis, they also propose to present users with a series of interventions and learn user preferences from the responses to these questions. The overview of the methodology is shown below in Figure 2.1.

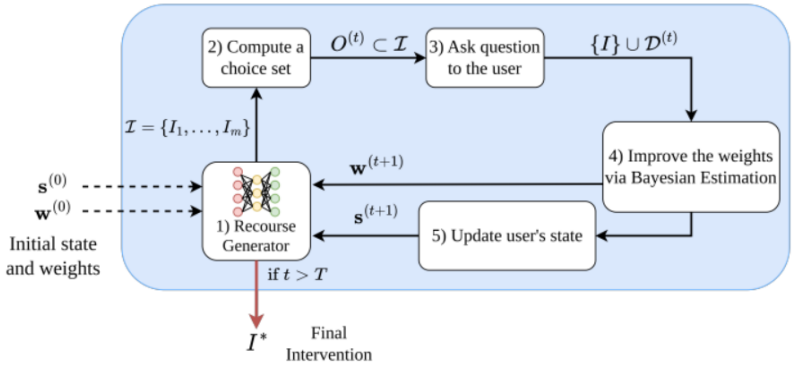
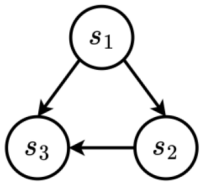


Figure 2.1: Overview of PEAR - taken from Figure 1 in De Toni et al. (2023)

They propose to compute a 'choice set', a list of interventions for the user to responds with which intervention they would prefer. Bayesian estimation is then performed to update

the weights, which correspond (a) user preferences over feature mutability and (b) the correlative effect of the value of x_i on the cost of *changing* x_j.

The methodology proposed in this thesis differs from De Toni et al. (2023) in two key ways. Firstly, we adopt a fully causal setting (motivated and discussed in detail in chapter 3). The cost function they use is shown below in Figure 2.2, where a_i corresponds to the action (or change) on variable i from s_i to s_i'. The weights w_i correspond to the user preferences over feature mutability and the the weights w_{ij} correspond to the correlative effect of the value of x_i on the cost of *changing* x_j. However, as seen in Figure 2.2, the cost of an action of *any* size on x_i is affected by the current value of its parents multiplied by a weight. Regardless of the size of the change in variables i, its parents have a fixed effect. For example, in a scenario where income has a causal effect on savings, then having an income of £50,000 increases/reduces the cost of increasing savings by the same amount if we increase savings from £1 to £2 as it does if we increase savings from £1m to £2m. We do not believe that this is a meaningful way to capture the effects of causal relations on the cost of changing feature values.



$$\mathbf{s} = [s_1, s_2, s_3]$$
$$\mathcal{A} = \{a_1, a_2, a_3\}$$

$$C(a_1, \mathbf{s}|\mathbf{w}) = w_1|s_1' - s_1|$$
$$C(a_2, \mathbf{s}|\mathbf{w}) = w_2|s_2' - s_2| + w_{12}s_1$$
$$C(a_3, \mathbf{s}|\mathbf{w}) = w_3|s_3' - s_3| + w_{13}s_1 + w_{23}s_2$$

Figure 2.2: Cost function in PEAR - taken from Figure 2 in De Toni et al. (2023)

There are two major (and several minor) differences in the methodology presented in this thesis and De Toni et al. (2023). Firstly, as detailed in chapter 3, we incorporate formal causal modelling (Pearl et al., 2016) in our cost function. Secondly, we learn an approximation of the structural causal model (SCM) that governs the world (and is the same for all individuals) as well as user preferences over feature mutability.

2.3 Recourse Generation Methods

There exist many different methods to generate recourse, which are reviewed in depth by Karimi et al. (2022). The methods used to generate recourse can broadly be grouped into three techniques.

- *Gradient-based optimisation.* When the cost function and constraint are both differentiable, gradient based approaches such as gradient descent (L-)BFGS and augmented Lagrangian methods (used in this thesis) are used to find alternative features $\mathbf{x}^*$.
- *(Mixed) Integer Linear Programming.* In non-differentiable settings or where the features are discrete, tools such as (Mixed) Integer Linear Programming are used to solve for $\mathbf{x}^*$.
- *Brute Force.* For cases with limited amounts of data and features and discrete data, brute force search through all possible values of $\mathbf{x}^*$ can be used (von Kügelgen et al., 2022).

This thesis aims to propose a methodology for learning cost functions, so to simplify the recourse problem we will only consider the case where continuous, actionable features are used. Examples of continuous, actionable features include income, a discrete, actionable feature would include level of education and a non-actionable feature would include age. As our data is continuous and actionable, we use gradient based optimisation (see section 3.4) to generate recourse.

3 Causal Algorithmic Recourse

In the introduction, three problems with cost functions used in the literature were outlined. The first of these was that actions are typically enacted sequentially (e.g., you first increase your income, and then increase your savings). This is important within the *causal* algorithmic setting, where intervening on a variable (e.g, increasing income) has downstream (causal) effects on other variables such as savings. In this chapter, we first motivate *causal* algorithmic recourse, then explore causal algorithmic recourse with soft, as opposed to hard interventions and finally propose a solution to optimise for ordering of sequential interventions when generating recourse.

3.1 Motivation

As discussed in section 2.2, the cost of changing features is typically modelled as an $\texttt{cost}(\mathbf{x}, \mathbf{x}') = L_p(\mathbf{x} - \mathbf{x}')$ or the total log percentile shift (Ustun et al., 2019). These cost functions fail to take into account the downstream (causal) effects of changing one feature on other features.

In order to correctly measure the cost of changing from features $\mathbf{x}$ to $\mathbf{x}'$, we need to be able to evaluate the causal effects of changing one variable on others. For example, 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.

The Mahalanobis distance cost function $\texttt{cost}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')$, where $\mathbf{M}$ is a fixed, known, positive semi-definite square matrix (Bechavod et al., 2022), does allow for changing one feature to affect the cost of changing other features through the off-diagonal elements in $\mathbf{M}$ (see equation 2.2.4 for a worked example). However, the causal effects rely on the changes being made simultaneously. If we first apply a change to $\mathbf{x}_1$ and then to $\mathbf{x}_2$, as shown below. In equations 3.1.1 and 3.1.2, the changes are applied sequentially and there are no causal effects, leading to a total cost of 5. In equation 3.1.3, the changes are applied simultaneously and the effect of changing $\mathbf{x}_1$ leads a lower cost of changing $\mathbf{x}_2$, with a total cost of 4. The importance of sequential changes as opposed to simultaneous changes are discussed in section 3.2.1.

$$[1,0,0]^T \begin{bmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1,0,0]^T = 1^2 + 0^2 + 0^2 = 1 \tag{3.1.1}$$

$$[0,2,0]^T \begin{bmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [0,2,0]^T = 0^2 + 2^2 + 0^2 = 4 \tag{3.1.2}$$

$$[1,2,0]^T \begin{bmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1,2,0]^T = 1^2 + 1.5(2) + 0^2 = 4 \tag{3.1.3}$$

3.2 Structural Causal Models

To incorporate changing one feature having a causal effect on 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 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 $\mathcal{M} = \langle \mathbb{U}, \mathbb{V}, \mathbb{F} \rangle$ 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)[1].

We can illustrate a simple SCM $\mathcal{M}$ with $\mathbb{U} = \{x_1\}$, $\mathbb{V} = \{x_2, x_3, x_4\}$ and structural equations $\mathbb{F}$ are defined in equation 3.2.1.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(30,5) \\ x_2 &= u_2 + 0.5x_1 & u_2 &\sim N(20,10) \\ x_3 &= u_3 - 0.25x_2 & u_3 &\sim \text{Uniform}(0,20) \\ x_4 &= \frac{x_3}{x_1} \end{aligned} \tag{3.2.1}$$

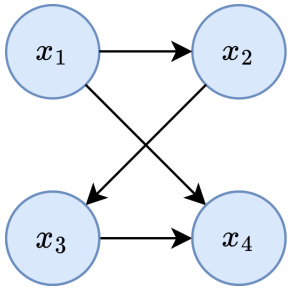


Figure 3.1: Causal graph $\mathcal{G}$ of SCM $\mathcal{M}$.

The SCM can also be presented with a causal graph $\mathcal{G}$, which is shown in Figure 3.1. Let x_1 represent salary (in thousands of pounds), x_2 represent savings (in thousands of pounds), x_3 represent the debt (in thousands of pounds) and x_4 represent debt-to-income ratio. If an individual for whom the SCM $\mathcal{M}$ holds receives an increase in salary, this will lead to an increase in savings and a reduction the debt-to-income ratio. If the individual's initial

[1]Whilst Pearl-style Directed Acyclic Graphs are the dominant methodology for modelling causal relations in the computer science and machine learning literature, it should ne noted that other approaches are common in other branches of literature, for example the potential outcomes framework in econometrics (Imbens, 2020)

features were [£30,000, £25,000, £10,000, 1/3]^T and their salary increases to £35,000, then it should then be easier to increase their savings and decrease their debt-to-income ratio 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, £10,000, 1/3]^T

go over this example again, should be such that the hard intervention actually leads to the severing of an edge to illustrate the point on hard interventions

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

$$\begin{aligned} u_1 &= x_1 && = 30,000 \\ u_2 &= x_2 - 0.5x_1 = 25,000 - 0.5(30,00) && = 10,000 \\ u_3 &= x_3 + 0.25x_2 = 10,000 + 0.25(25,000) && = 16,250 \end{aligned} \tag{3.2.2}$$

2. Action. Modify the model $\mathcal{M}$ by implementing the intervention on x_1 (i.e; $do(x_2 = 30,000)$). This leads to a new SCM $\mathcal{M}_1$ where all incoming edges to the intervened upon variable x_2 are severed and the value of the intervened upon variable x_2 is set to the intervention value £30,000. The resulting SCM $\mathcal{M}_1$ is shown in Figure 3.2, with structural equations $\mathcal{F}_1$.

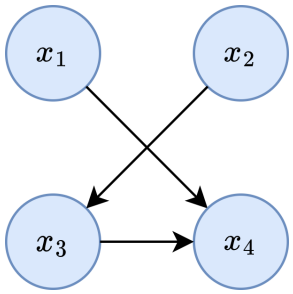


Figure 3.2: Causal graph $\mathcal{G}$ of SCM $\mathcal{M}_1$.

$$\begin{aligned} x_1 &= u_1 \\ x_2 &= 30,000 \\ x_3 &= u_3 - 0.25x_2 \\ x_4 &= \frac{x_3}{x_1} \end{aligned} \tag{3.2.3}$$

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

$$\begin{aligned} x_1^{\text{SCF}} &= u_1 && = 30,000 \\ x_2^{\text{SCF}} &= 30,000 && = 30,000 \\ x_3^{\text{SCF}} &= u_3 - 0.25x_2^{\text{SCF}} = 16,250 - 0.25(30,000) && = 8,750 \\ x_4^{\text{SCF}} &= \frac{x_3^{\text{SCF}}}{x_1^{\text{SCF}}} = \frac{8,750}{30,000} && = 0.292 \end{aligned} \tag{3.2.4}$$

Mathematically, we can denote the Action-Abduction-Prediction steps as shown in equation 3.2.5, 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, I is the set of variables that intervened upon (e.g,) and $\mathbb{I}$ is the indicator function.

$$x_i^{\text{SCF}} = \mathbb{I}_{i \in I}(x_i + \delta_i) + \mathbb{I}_{i \notin I}\Big(x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)\Big) \tag{3.2.5}$$

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 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 3.2.6.

$$\begin{aligned} A^* = &\underset{A}{\text{argmin}}\ \texttt{cost}(\mathbf{x}, A) \\ \text{s.t. } &h(\mathbf{x}^{\text{SCF}}) = 1, \\ &\mathbf{x}_i^{\text{SCF}} = \mathbb{I}_{i \in I}(\mathbf{x}_i + \boldsymbol{\delta}_i) + \mathbb{I}_{i \notin I}\Big(x_i + f_i(\mathbf{pa}_i^{\text{SCF}}) - f_i(\mathbf{pa}_i)\Big), \\ &A \in \mathcal{F} \end{aligned} \tag{3.2.6}$$

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).

$$\texttt{cost}(\mathbf{x}, A) = \sum_{i=1}^{D} c\Big(\text{do}(\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i)\Big) \tag{3.2.7}$$

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, £10,000, 1/3]^T, it is assumed that salary as increased at the same time as savings, as opposed to taking place sequentially.

3.2.1 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. Structural (hard) interventions are typically used in randomised control trials. For example,

in a medical trial, patients are assigned to either to receive either a treatment or a placebo. Any other variables that may have caused the patient to receive a treatment or not now have no causal effect, as in carrying out the trial we simply assign the patient to receive either the treatment or a placebo.

In the cases where assumption 1 is violated, we can then represent the interventions as soft (or parametric) interventions, which do not result in severing of incoming edges (Eberhardt and Scheines, 2007). After the soft (parametric) intervention, we denote the resulting value of $\mathbf{x}$ after the intervention as $\mathbf{x}^{\text{PCF}}$, the *parametric* counterfactual, which is defined below.

$$x_i^{\text{PCF}} = \mathbb{I}_{i \in I}\delta_i + \Big(x_i + f_i(\text{pa}_i^{\text{PCF}}) - f_i(\text{pa}_i)\Big) \tag{3.2.8}$$

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 3.2.9.

$$\begin{aligned} A^* = &\underset{A}{\text{argmin}}\ \texttt{cost}(\mathbf{x}, A) \\ \text{s.t. } &h(\mathbf{x}^{\text{PCF}}) = 1, \\ &\mathbf{x}^{\text{PCF}} = \mathbb{I}_{i \in I}\boldsymbol{\delta}_i + \Big(\mathbf{x}_i + f_i(\mathbf{pa}_i^{\text{PCF}}) - f_i(\mathbf{pa}_i)\Big), \\ &A \in \mathcal{F} \end{aligned} \tag{3.2.9}$$

3.2.2 Sequential Interventions

Assumption 2 of the causal recourse problem formulation in 3.2.5 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 3.2.8, the order of the intervention does not affect the counterfactual values x_i^{PCF}, but can affect the cost of actions A.

Potentially worth showing how $f_i(\text{pa}_i^{\text{PCF}})$ 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 $c(do(x_i := x_i + \delta_i))$ 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.

$$x_i^{\text{PCF}_i} = x_i + \delta_i \tag{3.2.10}$$

$$x_i^{\text{PCF}_j} = x_i + f_i(\text{pa}_i^{\text{PCF}}) - f_i(\text{pa}_i) \tag{3.2.11}$$

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 $x_i + f_i(\text{pa}_i^{\text{PCF}}) - f_i(\text{pa}_i)$ (the value after intervening on x_j, seen in equation 3.2.11). 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 $x_j^{\text{PCF}_i}$ 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 $A = \{(\mathbf{S}, \text{do}\{\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D})\}$ where $\mathbf{S}$ is a permutation of the set $\{1, \dots, D\}^N$ and represents the ordering of the intervention. Given this updated definition of A, the causal recourse formulation stays the same as shown in 3.2.9.

To replace o_i with a permutation set?

3.3 Differentiable Sorting

In order to solve equation 3.2.9 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 $\{1, \dots, D\}$) and equation 3.2.9 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 $S = \texttt{argsort}(O)$, where $\texttt{argsort}(O)$ returns the indexes of O that sorts O in ascending order. By optimising for O, we indirectly optimise the ordering S as S is defined as $S = \texttt{argsort}(O)$.

However, the operation `argsort` (a piecewise-constant function) 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).

For a given permutation (i.e., ordering) $\pi \in \{1, \dots, D\}^D$, 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 3.3.1. For example, the permutation matrix of the permutation $\pi = [3, 1, 2]^T$ is shown in equation 3.3.2.

$$P_\pi[i, j] = \begin{cases} 1 & \text{if } j = \pi_i \\ 0 & \text{otherwise} \end{cases} \tag{3.3.1}$$

$$\pi = [3, 1, 2]^T \implies P_\pi = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.3.2}$$

`SoftSort` defines a continuous relaxation for $P_{\texttt{argsort}(O)}$, defined in equation 3.3.3, 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.

$$\texttt{SoftSort}^d_\tau(O) = \texttt{softmax}\left(\frac{-d(\texttt{sort}(O)\mathbb{1}^T, \mathbb{1}O^T)}{\tau}\right) \tag{3.3.3}$$

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 $O = [2,5,4]^T$.

$$O = \begin{bmatrix}2\\5\\4\end{bmatrix} \implies P_{\texttt{argsort}(O)} = \begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix} \tag{3.3.4}$$

$$\texttt{SoftSort}^{|\cdot|}_1(O) = \texttt{softmax}\left(-\begin{bmatrix}|5-2| & |5-5| & |5-4|\\ |4-2| & |4-5| & |4-4|\\ |2-2| & |2-5| & |2-4|\end{bmatrix}\right) = \begin{bmatrix}0.04 & \mathbf{0.70} & 0.26\\ 0.09 & 0.24 & \mathbf{0.67}\\ \mathbf{0.85} & 0.04 & 0.11\end{bmatrix} \tag{3.3.5}$$

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 3.3.5, 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 3.2.9, we take the maximum 'probability' in each row as the ordering S, as opposed to weighting the costs of differing orderings by their 'probabilities'.

3.4 Generating Recourse

To generate recourse, we need to solve the below constrained optimisation problem from equation 3.2.9, where A is the ordered set $A = \{(\mathbf{S}, \text{do}\{\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D})\}$.

$$\begin{aligned} A^* = &\operatorname*{argmin}_A \texttt{cost}(\mathbf{x}, A) \\ \text{s.t. } &h(\mathbf{x}^{\text{PCF}}) = 1, \\ &\mathbf{x}^{\text{PCF}} = \mathbb{I}_{i\in I}\boldsymbol{\delta}_i + \Big(\mathbf{x}_i + f_i(\mathbf{pa}_i^{\text{PCF}}) - f_i(\mathbf{pa}_i)\Big), \\ &A \in \mathcal{F} \end{aligned} \tag{3.4.1}$$

To solve this problem, we convert the problem from a constrained optimisation problem to a unconstrained optimisation problem and solve using gradient descent. We make use

of Lagrange multipliers to reach the problem formulation as a min-max objective with constraint that the classifier scores the updated feature values $\mathbf{x}^{\text{PCF}}$ with a score of at least 0.5 (this can be changed for a given classifier score).

$$\min_{A \in \mathcal{F}} \max_{\boldsymbol{\lambda}} \sum_{i=1}^{D} \texttt{cost}(\mathbf{x}, A) - \boldsymbol{\lambda}\Big(h(\mathbf{x}^{\text{PCF}}) - 0.5\Big) \tag{3.4.2}$$

This expression is then optimised using gradient descent where in each epoch, a gradient descent step is taken for $\boldsymbol{\lambda}$ and then another gradient step is taken for A. The expression is optimised until $h(\mathbf{x}^{\text{SCF}})$ converges to 0.5.

Once the optimisation has converged, we are left with optimal actions A^* for a given classifier h, cost function `cost` and structural causal model $\mathcal{M}$. From the optimal actions, we can obtain the structural counterfactual feature values $\mathbf{x}^{\text{PCF}}$. It is these feature values that are presented to the negatively classified individuals, as opposed to the actions themselves.

We only present $\mathbf{x}^{\text{PCF}}$ because, whilst we know that $h(\mathbf{x}^{\text{PCF}}) = 1$, we cannot guarantee that $h(\mathbf{x}^{\text{PCF}}) = 1$ unless there is perfect knowledge of the structural causal model. To demonstrate this, we will re-use the SCM from equation 3.2.1, which we denote as $\mathcal{M}^{\text{TRUE}}$.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(30, 5) \\ x_2 &= u_2 + 0.5x_1 & u_2 &\sim N(20, 10) \\ x_3 &= u_3 - 0.25x_2 & u_3 &\sim \text{Uniform}(0, 20) \\ x_4 &= \frac{x_3}{x_1} \end{aligned} \tag{3.4.3}$$

Suppose the true SCM was misspecified and we instead used $\mathcal{M}^{\text{FALSE}}$, which is the same as $\mathcal{M}^{\text{TRUE}}$, with the exception of the relationship between x_1 and x_2 (income and savings). In $\mathcal{M}^{\text{FALSE}}$, the relationship is shown below.

$$x_2 = u_2 + x_1 \tag{3.4.4}$$

For an individual with original features $[30,\ 25,\ 10,\ 1/3]^T$, where h is a linear classifier with bias $b = 0$ and weights $w = [0.2, 0.2, -1.25, -3]$, they are initially negatively classified (8% of approval). Using $\mathcal{M}^{\text{FALSE}}$, they are provided actions $\text{do}(\mathbf{x}_1 = 35)$, which results in $\mathbf{x}^{\text{SCF}} = [35, 30, 8.75, 0.25]^T$ and would result in a positive classification (79% chance of approval). However, as $\mathcal{M}^{\text{FALSE}}$ is a misspecified SCM, the true structural counterfactual from $\text{do}(\mathbf{x}_1 = 35)$ is $\mathbf{x}^{\text{SCF}} = [35, 27.5, 9.375, 0.27]^T$, which results in a negative classification (49% chance of approval). Karimi et al. (2020) provide a more formal proof that, that if the true SCM is not known, then recourse cannot be guaranteed when only providing interventions.

expand this explanation, is a bit too dense

4 Cost Learning

In the introduction, three key issues were highlighted with existing cost function in the algorithmic recourse. The first issue, the use of sequential interventions, were covered in chapter 3. The second and third issues, taking into account preferences over feature mutability and specification of the underlying SCM, are covered in this chapter. We outline a novel methodology to simultaneously learn user preferences over feature mutability and specification of the SCM from the users' *revealed preferences*.

User preferences over feature mutability, or how easy it is for individuals to change different features, is important to take into account in our cost function. Without taking into account individual preferences over feature mutability, an individual with very low discretionary spending and highly irreducible costs (e.g., childcare) and limited opportunities for a salary increase may be asked to significantly increase their savings in order to be approved for a loan, instead of providing recourse in more more mutable features, such as improving credit score and number of loans successfully paid back.

The SCM used to calculate causal recourse is also an important to take into account. As outlined in section 3.1, in order to correctly measure the cost of changing features from $\mathbf{x}$ to $\mathbf{x}'$, we need to take into account the causal effects of changing one variable on the others. However, if we misspecify the SCM, then we inaccurately measure the causal of effect of changing one variables on other variables, which will lead to sub-optimal recourse provided to negatively classified individuals.

4.1 Learning from Revealed Preferences

Revealed preference theory (Samuelson, 1938, 1948) is an economic theory that states that if a consumer if offered a bundle of goods x and a bundle of goods y both within a budget set B and the consumer chooses x over y, then x is preferred to y. From the consumer's decision to purchase bundle x over bundle y, the consumer *reveals* their (normative) *preferences* over x and y.

We can use this idea to learn individuals' revealed preferences over feature mutability. Whilst humans are not able to describe their own preferences mathematically, they are able to express their preferences when given limited options to choose from. We propose to ask negatively classified individual a series of questions. In each question, individuals are asked to compare two sets of ordered actions A^1 and A^2 and responds with whichever one is least costly. If A^1 is preferred to A^2. We can use this information to parameterise a cost function which takes into account user preference.

Whilst when recourse is finally provided after costs have been learned, it is provided in the form of an alternative vector of features $\mathbf{x}^*$, note that we propose to ask the individuals to compare two sets of ordered actions. This is because are many combinations of different actions and orderings that would result in the same alternative vector of features $\mathbf{x}^{'}$. The deployer of the model (i.e., the digital bank in the credit scoring setting) does not observe the which actions the individuals would take in order to change their feature values from $\mathbf{x}$

to $\mathbf{x}'$. As the deployer of the model does not observe how much each feature would have been intervened upon, it becomes very difficult to learn how relatively mutable each feature is.

this paragraph needs work

A demonstration of one of the pairwise comparisons that users are asked to evaluate are shown in Figure 4.1. We envision that users are asked to evaluate these comparisons in an online system, where they must answer the pairwise comparisons in order for recourse to be generated and presented to the users.

Action Sequence 1	**Action Sequence 2**
1. Salary → £70,000	1. Savings → £45,000
then	*then*
2. Savings → £52,000	2. Debt → £5,000
then	*then*
3. Debt → £7,000	3. Salary → £65,000

Question for user:
Which of the above sequences of actions are easier to complete?
☐ Action Sequence 1
☐ Action Sequence 2

Figure 4.1: Hypothetical pairwise comparison a user is asked to respond to.

In order to effectively learn from the responses of the negatively classified individuals, we make an assumption - that individuals can evaluate their actions using the true SCM. Whilst on the surface, this may appear to be a very strong assumption, it is not unreasonable to think that, in certain settings such as credit scoring, an individual could evaluate the effect of a change in one variable, for example income from £50,000 to £70,000 on other variables such as savings and debt. This is because an individual would be able to estimate how much of the additional income they would be able to save, how much they could use to pay down any existing debt, and how much would likely be allocated to discretionary spending. We do not assume complete knowledge of the SCM - we do not assume that individuals are able to evaluate causal effects for any individual other than themselves. We assume that individuals have 'local' knowledge of the SCM. Later in this chapter, we relax this assumption slightly to assume that individuals have noisy, local knowledge of the SCM.

4.2 Cost Learning Formulation

We now formulate the cost learning problem, where there are N individuals who are each presented with K pairwise comparisons. In each comparison, each individual compares two sets of ordered actions (A^1, A^2). We record the response of the n^{th} individual to their k^{th} comparison as y_{kn}, which is defined below, where $\mathbf{x}_n$ represents the n^{th} individuals' original features.

$$y_{kn} = \begin{cases} -1 & \text{if } A^1_{kn} \text{ preferred to } A^2_{kn} \\ +1 & \text{if } A^2_{kn} \text{ preferred to } A^1_{kn} \end{cases} \tag{4.2.1}$$

4.2.1 Perfect Knowledge of the Structural Causal Model

Let us first consider the case where both the deployer of the model and the users have perfect knowledge of the underlying SCM $\mathcal{M}$. In this case, the individual k has a cost function $\texttt{cost}(\mathbf{x}, A|\beta_k, \mathcal{F})$, parameterised by $\mathcal{F}$, the corresponding structural equations of the SCM $\mathcal{M}$ and β_k, a vector which represents how easily mutable each feature is. As the model deployer has perfect knowledge of the SCM, they do not need to learn (or approximate) $\mathcal{F}$. The model deployer's task become to learn a β_i for each individual which represents each users' preferences (of how mutable each feature is) that explains as many of the individuals' responses as possible. We denote the model deployers' predicted response as $\hat{y}_{kn}$, which is defined below, where λ is a hyperparameter regularising for 'confidence' of the predictions. As true values $y_{kn} \in \{-1, 1\}$, the hyperbolic tangent function is used to squash the predicted values $\hat{y}_{kn}$ into $[-1, 1]$.

$$\hat{y}_{kn} = \tanh\left(\lambda\Big(\texttt{cost}(\mathbf{x}_n, A^1_{kn}|\beta_k, \mathcal{F}) - \texttt{cost}(\mathbf{x}_n, A^2_{kn}|\beta_k, \mathcal{F})\Big)\right) \tag{4.2.2}$$

The model deployer's task can be achieved through the below objective, where there are K individuals and N pairwise comparisons and $\ell(y, \hat{y}) = \max[0, 1 - y\hat{y}]$ represents the hinge loss.

$$\beta^* = \underset{\beta}{\operatorname{argmin}} \frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \ell(y_{kn}, \hat{y}_{kn}) + \underbrace{\lambda||\beta_k||_2}_{\text{L2 regularisation}} \tag{4.2.3}$$

This is an unconstrained optimised problem that can be optimised using gradient descent. In order to avoid overfitting to responses of the sample of the negatively classified individuals who answer the pairwise comparisons, L2 regularisation is also added to the objective function.

4.2.2 Linear Approximation of the Structural Causal Model

Now we consider the (more realistic) case where the model deployer does not have perfect knowledge of the structural causal model. In this case, the model deployer learns both the user preferences β_k as well as an approximation of the structural equations $\mathcal{F}$.

In this case, model deployer knows neither the form of the structural equations $\mathcal{F}$ nor the parameters of the equations. For example, the true structural equations could be linear, as shown below for a three variable SCM.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(\mu_1, \sigma_1^2) \\ x_2 &= ax_1 + u_2 & u_2 &\sim N(\mu_2, \sigma_2^2) \\ x_3 &= bx_1 + cx_2 + u_3 & u_3 &\sim N(\mu_3, \sigma_3^2) \end{aligned} \tag{4.2.4}$$

The structural equations could also be a more complex non-linear set of equations. An example is shown below.

$$
\begin{aligned}
x_1 &= u_1 & u_1 &\sim N(\mu_1, \sigma_1^2) \\
x_2 &= ax_1 + bx_1^2 + u_2 & u_2 &\sim N(\mu_2, \sigma_2^2) \\
x_3 &= \frac{cx_1}{1 + \exp(dx_2)} + u_3 & u_3 &\sim N(\mu_3, \sigma_3^2)
\end{aligned} \tag{4.2.5}
$$

The model deployer knows neither the form of the equations nor the parameters (e.g, a, b, c, d in the above example). As a solution to this problem, we learn a *linear approximation* of the SCM[1].

In the linear approximation of the SCM, each variable is modelled as a linear combination of the other variable. We can represent the parameters of the linear approximation of the SCM as a square matrix W, where each element of the matrix represents $\frac{\partial x_j}{\partial x_i}$. As each variable x_j is a linear combination of the other variables $x_{\neg j}$, $\frac{\partial x_j}{\partial x_i}$ will be a scalar (and will be 0 if x_j has no causal effect on x_i). For the linear SCM described in equation 4.2.4, the parameters W can be described as shown below. It can be thought of as a weighted adjacency matrix, where the weights are the marginal effects of x_j on x_i.

$$
W = \begin{bmatrix} 1 & a & ab + c \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2.6}
$$

With a linear approximation of the SCM, the model deployer's task is now to learn the parameters W and user preferences β_k. We add W in to the objective function as follows.

$$
\beta^*, W^* = \underset{\beta, W}{\operatorname{argmin}} \frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \ell(y_{kn}, \hat{y}_{kn}) + \underbrace{\lambda_1 ||\beta_k||_2 + \lambda_2 ||W||_2}_{\text{L2 regularisation}} \tag{4.2.7}
$$

We additionally restrict W such that the diagonal is fixed at 1 - meaning that an intervention of size δ_i on variable i has a causal effect of δ_i. For example, if we increased salary from £30,000 to £35,000 ($\delta_i = £5,000$), then income after the intervention is £35,000. It does not lead to downstream (causal) effects that result in salary changing to any value other than £35,000 ($x_i + \delta_i$).

4.2.3 Noisy Responses

In the two previous set-ups the model deployer learns β_k (and W) from the responses of the negatively classified individuals, who have perfect knowledge of their own preferences β_k and the true SCM $\mathcal{M}$. However, in reality, often responses to such questions can be noisy and it is highly unlikely that individuals actually have perfect knowledge of the SCM. We now relax these two assumptions. First, we assume that users only know the SCM up to some noise. This means that, when evaluating which of the sets of actions A^1 and A^2 they prefer, they assume that the downstream (causal) effect of each action within the set of actions is a noisy version of the true downstream effect. If we assume that this noise is

[1] A linear approximation of the SCM is used in this thesis largely for its simplicity. However a more flexible specification of the approximation of the SCM, such as kernel ridge regression, could be used in future works.

drawn from a probability distribution with mean 0 then, *on average*, users have perfect knowledge of the true SCM. Secondly, we assume that individuals additionally evaluate their preferences with some noise (they are not able to perfectly evaluate which of the sets of actions they would prefer perfectly given their noisy belief of the SCM as some noise is added to the decision making process).

To add noise to the users' knowledge of the SCM, after an intervention on x_i, we add noise to the other variables $x_{\neg i}$ after the prediction step in Abduction-Action-Prediction steps (Pearl et al., 2016) (see section 3.2 for an explanation of the Abduction-Action-Prediction steps). We assume that the noise is drawn from a Gaussian distribution with mean 0 and standard deviation σ_F. We denote the SCM that the users evaluate the effects of interventions with as $\tilde{\mathcal{M}}$ with associated structural equations $\tilde{\mathcal{F}}$.

To add noise the the users' evaluation of their own preferences, we calculate the ratio of the costs of the sets of actions A^1 and A^2 and then add some noise. Again, we assume that this noise is Gaussian with mean 0 and with standard deviation σ_B. Now, the response of the users is defined as follows.

$$r = \frac{\texttt{cost}(\mathbf{x}_n, A^1_{kn} | \beta_k, \tilde{\mathcal{F}})}{\texttt{cost}(\mathbf{x}_n, A^2_{kn} | \beta_k, \tilde{\mathcal{F}})} + N(0, \sigma_B^2) \tag{4.2.8}$$

$$y_{kn} = \begin{cases} -1 & \text{if } r \leq 1 \\ +1 & \text{if } r > 1 \end{cases}$$

With these updated noisy responses, the optimisation problem still remains the same from the point of the model deployer. However, given the added noise, the responses of users may not always reflect the true SCM and true preferences of the users. Therefore, the role of regularisation becomes more important.

4.3 Cost Function

The cost function of actions A and user preferences over feature mutability β_k for the individual k takes the below form, where $\beta \in \mathbb{R}^D$ and D is the number of features[2]. Squaring the intervention δ_i on variable i has several desirable properties. Notably that (a), the cost is always greater than or equal to 0, (b) an intervention of $\delta_i = 0$ results is a cost of 0 (i.e., doing nothing has no associated cost) and (c) larger interventions results in (polynomially) increasing costs, meaning that an intervention of $2a$ on variable x_i has more than twice the cost of a.

$$\texttt{cost}(A, \beta) = \sum_{i=1}^{D} \beta_i \delta_i^2 \tag{4.3.1}$$

[2]The order of the interventions in A does not matter in this case, as the cost function only depends on δ, not $\mathbf{x}$ (see Proposition 1)

5 Experiments

In this chapter, we evaluate how effective our learned cost function is, which is described below, where D is the number of features, A is the set of actions and the resulting value of $\mathbf{x}$ after the interventions is $\mathbf{x}^{\text{PCF}}$.

$$\begin{aligned} \texttt{cost}(A, \beta_k, \mathcal{F}) &= \sum_{i=1}^{D} \delta_i \beta_{ki}^2 \\ A &= \{(\mathbf{S}, \text{do}\{\mathbf{x}_i := \mathbf{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D})\} \\ \mathbf{x}^{\text{PCF}} &= \mathbb{I}_{i \in I} \boldsymbol{\delta}_i + \left(\mathbf{x}_i + f_i(\mathbf{pa}_i^{\text{PCF}}) - f_i(\mathbf{pa}_i) \right) \end{aligned} \tag{5.0.1}$$

5.1 End-to-End Methodology

The end to end methodology for generating recourse with a learned cost is shown in Figure 5.1, and more details on each of the steps in the methodology are described below.

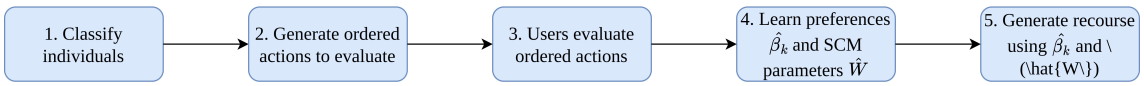


Figure 5.1: End-to-End Methodology

1. **Classify individuals**. Individuals with features $\mathbf{X}$ are assessed by a classifier and are classified either positively (e.g., loan approved, admission acceptance) or negatively (e.g., loan rejected, admission rejected). In this thesis, logistic regression has been used, largely for simplicity. However, as an augmented Lagrangian gradient descent optimisation is used to generate recourse (see section 3.4), any differentiable classifier could be used instead of logistic regression.
2. **Generate ordered actions to evaluate**. We randomly generate actions for each negatively classified individual and associated randomly generated orderings. This is another area for future work, where an online or Bayesian approach could be used to generate recourse actions in order to maximise information gain, such as the approach used in De Toni et al. (2023).
3. **Users evaluate ordered actions**. Using the cost function described in equation 5.0.1 with noisy user preferences $\tilde{\beta}_k$ and noisy SCM $\tilde{\mathcal{F}}$, users evaluate which of the ordered actions they are presented with they prefer.
4. **Learn preferences $\hat{\beta}_k$ and SCM approximation parameters $\hat{W}$**. The model deployer then learns user preferences $\hat{\beta}_k$ and a linear approximation of the true SCM $\hat{W}$ using the optimisation presented in section 4.2.
5. **Generate recourse using $\hat{\beta}_k$ and $\hat{W}$**. The model deployer then generates recourse $\mathbf{X}^*$ for the negatively classified users.

5.2 Synthetic Data

We create synthetic SCMs, from which we sample our data $\mathbf{X}$. We simulate two different SCMs, the first being a linear SCM and the second a non-linear SCM. In both cases, we assign individuals to have the same randomly generated feature mutability β plus some noise. This means that individuals tend to have similar preferences, but do not have the same preferences.

5.2.1 Linear SCM

Shown below are the structural equations of the linear SCM $\mathcal{M}_{\text{LIN}}$ as well as the associated causal graph.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(0,1) \\ x_2 &= u_2 + 0.5x_1 & u_2 &\sim N(0,1) \\ x_3 &= u_3 + 0.2x_1 + 0.3x_2 & u_3 &\sim N(0,0.5) \\ y &\sim \text{Bernoulli}(\sigma(0.1x_1 + 0.2x_2 + 0.3x_3)) \end{aligned} \tag{5.2.1}$$

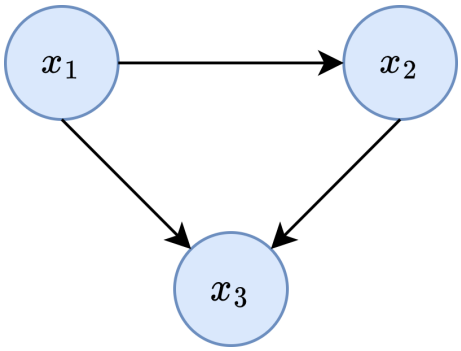


Figure 5.2: Linear SCM $\mathcal{M}_{\text{LIN}}$

5.2.2 Non-Linear SCM

Shown below are the structural equations of the non-linear SCM $\mathcal{M}_{\text{NL}}$ as well as the associated causal graph.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(0,1) \\ x_2 &= u_2 - \frac{2}{1+x_1^2} & u_2 &\sim N(0,1) \\ x_3 &= u_3 + 0.2x_1 + 0.1x_1x_2 & u_3 &\sim N(0,0.5) \\ x_4 &= u_4 + 0.4x_1 + 0.7x_2x_3 & u_4 &\sim N(0,0,5) \\ x_5 &= u_5 + 0.8x_1 & u_5 &\sim N(0,0.5) \\ y &\sim \text{Bernoulli}(\sigma(0.1x_1 + 0.2x_2 + 0.3x_3 + 0.4x_4 + 0.5x_5)) \end{aligned} \tag{5.2.2}$$

5.3 Evaluation Criteria

can probably ditch this section, consider have basically got rid of it

To how useful the learned user preferences $\hat{\beta}_k$ and SCM approximation parameters $\hat{W}$ are, we need to be able to evaluate $\texttt{cost}(\mathbf{x}, \mathbf{x}^*)$, where $\mathbf{x}^*$ represents the recourse generated.

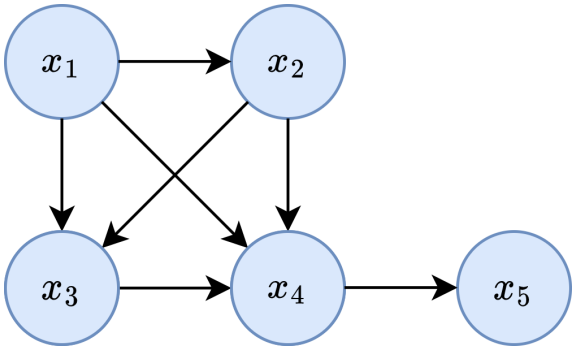


Figure 5.3: Non-Linear SCM $\mathcal{M}_{\text{NL}}$

To do this, we propose simply minimise the squared distance between the recourse $\mathbf{x}^*$ and $\mathbf{x}'$, which represents the parametric counterfactual from ordered actions A under the true SCM. The objective function that is optimised is shown below.

$$A^* = \underset{A}{\text{argmin}}(\mathbf{x}^* - \mathbf{x}')^2 \tag{5.3.1}$$

where

$$\begin{aligned} A &= \{(\mathbf{S}, \text{do}\{\mathbf{x}'_i := \mathbf{x}'_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D})\} \\ \mathbf{x}'^{\text{PCF}} &= \mathbb{I}_{i \in I}\boldsymbol{\delta}_i + \left(\mathbf{x}'_i + f_i(\mathbf{pa}_i^{\text{PCF}}) - f_i(\mathbf{pa}_i)\right) \end{aligned} \tag{5.3.2}$$

This can be optimised using gradient descent. Once we have obtained ordered actions A^* that lead to our recourse $\mathbf{x}^*$ under the true SCM, we can calculate $\texttt{cost}(\mathbf{x}, \mathbf{x}')$ as $\texttt{cost}(A^*, \beta_k, \mathcal{F})$ as defined in equation 5.0.1, where β_k represents the true user preferences and $\mathcal{F}$ are the true structural equations.

5.4 Results

In this section, we present the results of the learned cost function on the two SCMs described in the previous section. In all the experiments conducted in this section, we simulate 2,000 individuals from the SCMs, which are then classified either positively or negatively.

5.4.1 Linear SCM

In this section, we detail the performance of our methodology on the linear SCM outlined in section 5.2.1. We compare the cost of the recourse we generated to the cost of the recourse generated using the ground truth user preferences β_k and the ground truth SCM. Assuming that there is no noise in either the users' evaluation of their own preferences and their local evaluation of the SCM, we have calculated the percentage increase in cost of the recourse generated compared to the ground truth recourse.

5.4.2 Non-Linear SCM

==========

[TO CLEAN UP]

Model	Cost	Increase in cost (%)
Ground truth	0.214	0%
Identity W	0.363	69.5%
5 comparisons	0.332	55.1%
10 comparisons	0.290	35.4%
20 comparisons	0.250	16.7%
50 comparisons	0.224	4.6%

Table 5.1: Ground truth cost of recourse

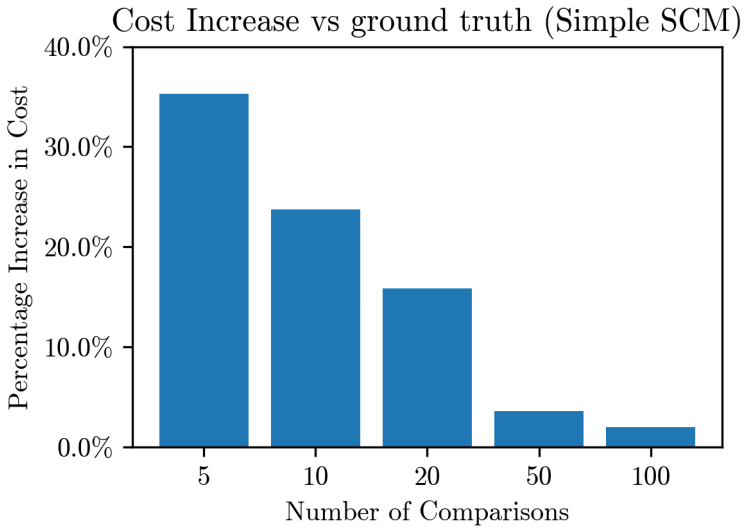


Figure 5.4: Comparison of the average cost of recourse for learned costs functions compared to the ground truth cost function for the Simple SCM.

5.4.3 Noiseless responses with perfect knowledge of the causal graph

Ground truth β: [0.5, 0.333, 0.1667]
Learned β : [0.4965, 0.3355, 0.1681]

5.4.4 Noiseless responses with imperfect knowledge of the causal graph

Ground truth β: [0.5, 0.333, 0.1667]
Learned β : [0.4975, 0.3354, 0.1671]

Ground truth W: $\begin{bmatrix} 0 & 0.5 & 0.2 \\ 0 & 0 & 0.3 \\ 0 & 0 & 0 \end{bmatrix}$

Learned W: $\begin{bmatrix} 0 & 0.4864 & 0.2136 \\ 0.0033 & 0 & 0.3221 \\ 0.0008 & 0.0086 & 0 \end{bmatrix}$

5.4.5 Noisy responses

So far, only noisy responses have been added (not noise knowledge of the causal graph)

With noise distribution $N(0, 0.1)$:

Ground truth β: [0.5, 0.333, 0.1667]
Learned β : [0.4917, 0.3418, 0.1665]

Ground truth W: $\begin{bmatrix} 0 & 0.5 & 0.2 \\ 0 & 0 & 0.3 \\ 0 & 0 & 0 \end{bmatrix}$

Learned W: $\begin{bmatrix} 0 & 0.5424 & 0.1683 \\ -0.0070 & 0 & 0.3950 \\ 0.0166 & -0.0007 & 0 \end{bmatrix}$

With noise distribution $N(0, 0.5)$:

Ground truth β: [0.5, 0.333, 0.1667]
Learned β : [0.4224, 0.3686, 0.2090]

Ground truth W: $\begin{bmatrix} 0 & 0.5 & 0.2 \\ 0 & 0 & 0.3 \\ 0 & 0 & 0 \end{bmatrix}$

Learned W: $\begin{bmatrix} 0 & 0.5772 & 0.3522 \\ -0.0062 & 0 & 0.1657 \\ 0.0974 & 0.0459 & 0 \end{bmatrix}$

With noise distribution $N(0, 2)$:

Ground truth β: [0.5, 0.333, 0.1667]
Learned β : [0.4958, 0.2685, 0.2357]

Ground truth W: $\begin{bmatrix} 0 & 0.5 & 0.2 \\ 0 & 0 & 0.3 \\ 0 & 0 & 0 \end{bmatrix}$

Learned W: $\begin{bmatrix} 0 & 0.5353 & 0.08712 \\ -0.1487 & 0 & 0.7206 \\ -0.2152 & 0.2409 & 0 \end{bmatrix}$

6 Conclusion

Conclusion chapter.

References

Angwin, Julia, Jeff Larson, Lauren Kirchner, and Surya Mattu (2016). *Machine Bias*. ProPublica. URL: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing (visited on 08/28/2023) (Cited on page 3).

Bechavod, Yahav, Chara Podimata, Zhiwei Steven Wu, and Juba Ziani (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 7, 11).

Caron, François and Arnaud Doucet (2012). "Efficient Bayesian Inference for Generalized Bradley—Terry Models". In: *Journal of Computational and Graphical Statistics* 21.1, pp. 174–196. URL: https://www.jstor.org/stable/23248829 (visited on 09/08/2023) (Cited on page 8).

De Toni, Giovanni, Paolo Viappiani, Stefano Teso, Bruno Lepri, and Andrea Passerini (2023). *Personalized Algorithmic Recourse with Preference Elicitation*. DOI: 10.48550/arXiv.2205.13743. (Visited on 09/06/2023). preprint (Cited on pages 8, 9, 24).

Eberhardt, Frederick and Richard Scheines (2007). "Interventions and Causal Inference". In: *Philosophy of Science* 74.5, pp. 981–995. DOI: 10.1086/525638. (Visited on 08/18/2023) (Cited on page 15).

Imbens, Guido W. (2020). "Potential Outcome and Directed Acyclic Graph Approaches to Causality: Relevance for Empirical Practice in Economics". In: *Journal of Economic Literature* 58.4, pp. 1129–1179. DOI: 10.1257/jel.20191597. (Visited on 09/07/2023) (Cited on page 12).

Karimi, Amir-Hossein, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera (2022). "A Survey of Algorithmic Recourse: Contrastive Explanations and Consequential Recommendations". In: *ACM Comput. Surv.* 55.5. DOI: 10.1145/3527848 (Cited on pages 4, 6, 9).

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 12, 14).

Karimi, Amir-Hossein, Julius von Kügelgen, Bernhard Schölkopf, and Isabel Valera (2020). "Algorithmic Recourse under Imperfect Causal Knowledge: A Probabilistic Approach". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 265–277. URL: https://proceedings.neurips.cc/paper/2020/hash/02a3c7fb3f489288ae6942498498db20-Abstract.html (visited on 08/30/2023) (Cited on page 18).

Kramer, Anna (2022). *The Problems with AI Video Interviews*. Protocol. URL: https://www.protocol.com/workplace/automated-video-interviews-hirevue-modernhire (visited on 08/29/2023) (Cited on page 3).

O'Dwyer, Rachel (2018). *Are You Creditworthy? The Algorithm Will Decide*. Undark Magazine. URL: https://undark.org/2018/05/07/algorithmic-credit-scoring-machine-learning/ (visited on 08/28/2023) (Cited on page 3).

Pearl, J., M. Glymour, and N.P. Jewell (2016). *Causal Inference in Statistics: A Primer*. Wiley (Cited on pages 9, 12, 13, 23).

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 (visited on 08/20/2023) (Cited on page 16).

Ramakrishnan, Goutham, Yun Chan Lee, and Aws Albarghouthi (2020). "Synthesizing Action Sequences for Modifying Model Decisions". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.04 (04), pp. 5462–5469. DOI: 10.1609/aaai.v34i04.5996. (Visited on 08/29/2023) (Cited on page 6).

Rawal, Kaivalya and Himabindu Lakkaraju (2020). "Beyond Individualized Recourse: Interpretable and Interactive Summaries of Actionable Recourses". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 12187–12198. URL: https://proceedings.neurips.cc/paper/2020/hash/8ee7730e97c67473a424ccfeff49ab20-Abstract.html (visited on 09/08/2023) (Cited on page 8).

Samuelson, Paul (1938). "A Note on the Pure Theory of Consumer's Behaviour". In: *Economica* 5.17, pp. 61–71. DOI: 10.2307/2548836. (Visited on 08/23/2023) (Cited on page 19).

Samuelson, Paul (1948). "Consumption Theory in Terms of Revealed Preference". In: *Economica* 15.60, pp. 243–253. DOI: 10.2307/2549561. (Visited on 08/23/2023) (Cited on page 19).

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. (Visited on 08/19/2023) (Cited on pages 6, 11, 14).

Venkatasubramanian, Suresh and Mark Alfano (2020). "The Philosophical Basis of Algorithmic Recourse". In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. New York, NY, USA: Association for Computing Machinery, pp. 284–293. DOI: 10.1145/3351095.3372876. (Visited on 08/29/2023) (Cited on page 3).

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 3).

Von Kügelgen, Julius, Amir-Hossein Karimi, Umang Bhatt, Isabel Valera, Adrian Weller, and Bernhard Schölkopf (2022). "On the Fairness of Causal Algorithmic Recourse". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.9 (9), pp. 9584–9594. DOI: 10.1609/aaai.v36i9.21192. (Visited on 09/08/2023) (Cited on page 9).