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: **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 $\boldsymbol{x}'$ 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 $\boldsymbol{x}$ 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 $x_{\text{RACE}} =$ "Black", then $x'_{\text{RACE}} =$ "White" is not feasible).

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

[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 $\boldsymbol{x}$ to $\boldsymbol{x}'$. 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 two 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 obatining 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.

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

[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.

of experiments on synthetic data are in chapter 5 and concluding remarks are made in chapter 6.

2 Literature Review

2.1 Algorithmic Recourse

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.

2.1.1 Problem Formulation

The algorithmic recourse problem can be defined formally as shown in equation 2.1.1, where an individuals' original features are $\boldsymbol{x}$, $h : \mathbb{R}^D \rightarrow \{0, 1\}$ is the classifier and $\mathcal{F}$ is the set of feasible features values. The set of feasible feature values constrains $\boldsymbol{x}'$ by only allowing positive/integer/similar constraints values of $\boldsymbol{x}'_i$ 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 $\boldsymbol{x}_i = \boldsymbol{x}'_i$.

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

2.1.2 Cost Functions

Typically, the cost function is either of the form $L_p(\boldsymbol{x}' - \boldsymbol{x}')$, 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 $\boldsymbol{x} = \boldsymbol{x}'$. The further away from the original features $\boldsymbol{x}$ that the counterfactual values $\boldsymbol{x}'$ 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 2.1.2, where D is the number of features and Q_i represents the CDF of $\boldsymbol{x}_i$. This cost function also considers the cost of each feature changed independently. It punishes increasing from the percentile $Q_i(\boldsymbol{x}'_i)$ is larger than the original percentile $Q_i(\boldsymbol{x}_i)$. A key advantage of this cost function is that changes

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

become harder when starting from a higher percentile, e.g, moving from the 50$^{\text{th}}$ to 55$^{\text{th}}$ 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 $\boldsymbol{x}_i = \boldsymbol{x}'_i$, it becomes negative when $Q_i(\boldsymbol{x}'_i) < Q_i(\boldsymbol{x}_i)$. 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

$$\texttt{cost}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{D} \log\left(\frac{1 - Q_i(\boldsymbol{x}_i)}{1 - Q_i(\boldsymbol{x}'_i)}\right) \tag{2.1.2}$$

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 $\boldsymbol{x}$ to $\boldsymbol{x}'$. 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 2.1.3, where $\boldsymbol{M}$ is a positive semi-definite matrix.

$$\texttt{cost}(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}') \tag{2.1.3}$$

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 $\boldsymbol{x} = [2, 3, 4]^T$ and $\boldsymbol{x}' = [1, 1, 1]^T$. First, note that $(\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^2$ if $\boldsymbol{M}$ is the identity matrix. If the values along the diagonal of $\boldsymbol{M}$ were different, this would encode different costs for changing each feature.

$$[1, 2, 3]^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1, 2, 3]^T = 1^2 + 2^2 + 3^2 = 14 \tag{2.1.4}$$

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

$$[1, 2, 3]^T \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [1, 2, 3]^T = 1^2 + 2.5(2) + 3^2 = 15 \tag{2.1.5}$$

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

explain how this is correlative, not causal

[TO CONSIDER WHICH OF THE BELOW ACTUALLY NEEDS TO GO IN THE LIT REVIEW]

2.1.3 Recourse methods

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

2.2 Learning from 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 (Balcan et al., 2014; Zadimoghaddam and Roth, 2012).
- 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).

3 Causal Algorithmic Recourse

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

Add citations

this paragraph needs updating in lieu of re-written literature review

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

1. **Changing one feature has a causal effect on 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.

   to add a worked example and reference specific variables in a dataset

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. This is not typically taken into account in the literature.

This thesis will primarily deal with the first issue, that changing features can have downstream, causal effects on other features. The second issue is out of the scope of this thesis. However, the methodology for estimating cost functions proposed in the thesis could be extended to a setting where individuals are divided in to groups (or clusters), and for each cluster a different cost function is estimated (a similar approach is taken by Bechavod et al. (2022)). Whilst this is not quite individualised, it also leads to a cost function being estimated that could be used for other downstream tasks, such as strategic classification.

3.1 Structural Causal Models

To address the first issue, that changing one feature has a causal effect on the cost of changing another feature, we must model the effect of changing the feature $\boldsymbol{x}_1$ to $\mathbf{x}'_1$ has on changing the feature $\boldsymbol{x}_2$ 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).

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 3.1.1.

$$\begin{aligned}
x_1 &= u_1 & u_1 &\sim N(30, 5) \\
x_2 &= u_2 + 0.5x_1 & u_2 &\sim N(20, 10) \\
x_3 &= u_3 - 0.25x_2 & u_3 &\sim \text{Uniform}(0, 20) \\
x_4 &= \frac{x_3}{x_1}
\end{aligned} \tag{3.1.1}$$

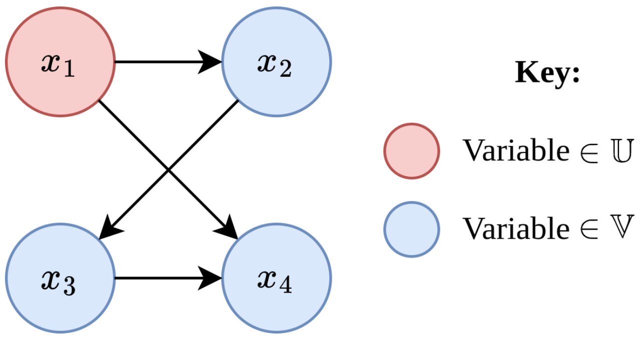

Figure 3.1: Causal graph $\mathcal{G}$ of SCM $\mathcal{M}$.

The SCM can also be presented with a causal graph $\mathcal{G}$, which is shown in Figure 3.1. Let x_1 represent salary (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 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.

[DISCUSSION ON PEARL-STYLE DAGS/INBENS CRITIQUE]

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).

$$\begin{aligned}
u_1 &= x_1 && = 30,000 \\
u_2 &= x_2 - 0.5x_1 = 25,000 - 0.5(30,00) && = 10,000 \\
u_3 &= x_3 + 0.25x_2 = 10,000 + 0.25(25,000) && = 16,250
\end{aligned} \tag{3.1.2}$$

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 $\mathcal{F}_1$.

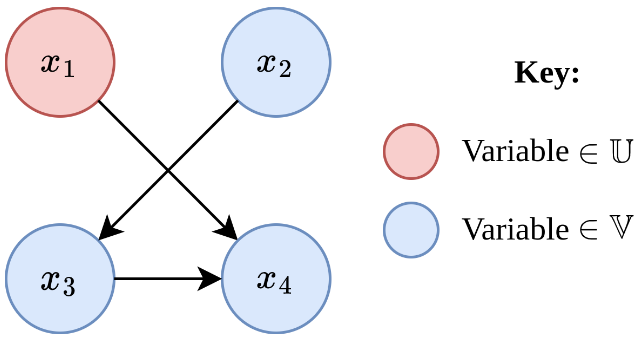

Figure 3.2: Causal graph $\mathcal{G}$ of SCM $\mathcal{M}_1$.

$$\begin{aligned} x_1 &= u_1 \\ x_2 &= 30,000 \\ x_3 &= u_3 - 0.25x_2 \\ x_4 &= \frac{x_3}{x_1} \end{aligned} \tag{3.1.3}$$

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

$$\begin{aligned} x_1^{\text{SCF}} &= u_1 && = 35,000 \\ x_2^{\text{SCF}} &= 30,000 && = 30,000 \\ x_3^{\text{SCF}} &= u_3 - 0.25x_2^{\text{SCF}} = 16,250 - 0.25(30,000) && = 8,750 \\ x_4^{\text{SCF}} &= \frac{x_3^{\text{SCF}}}{x_1^{\text{SCF}}} = \frac{8,750}{35,000} && = 0.25 \end{aligned} \tag{3.1.4}$$

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, I is the set of variables that intervened upon (e.g,) and $\mathbb{I}$ is the indicator function.

Introduce δ_i notation

$$x_i^{\text{SCF}} = \mathbb{I}_{i \in I}(x_i + \delta_i) + \mathbb{I}_{i \notin I}\Big(x_i + f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)\Big) \tag{3.1.5}$$

To calculate recourse $\boldsymbol{x}^*$ with an SCM over all the features within $\boldsymbol{x}$, 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 $\boldsymbol{x}$ to $\boldsymbol{x}'$ with minimising the cost of *interventions* $A = \text{do}\{\boldsymbol{x}_i := \boldsymbol{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D}$ 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 $x_i + \delta_i$

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

We define the cost of interventions A on original features $\boldsymbol{x}$ 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).

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

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 = $[£32{,}000,\ £27{,}500,\ £10{,}000,\ 1/3]^T$, it is assumed that salary as increased at the same time as savings, as opposed to taking place sequentially.

3.1.1 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 $f_i(\text{pa}_i^{\text{SCF}}) - f_i(\text{pa}_i)$.

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

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.

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

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 $c(do(x_i := x_i + \delta_i))$ 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 $A = \{(\mathbf{S}, \text{do}\{\boldsymbol{x}_i := \boldsymbol{x}_i + \boldsymbol{\delta}_i\}_{i=1,\ldots,D})\}$ 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 $O = [2,5,4]^T$.

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

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

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

3.3 Generating Recourse

To generate recourse, we need to solve the below constrained optimisation problem from equation 3.1.9, where A is the ordered set $A = \{(\mathbf{S}, \text{do}\{\boldsymbol{x}_i := \boldsymbol{x}_i + \boldsymbol{\delta}_i\}_{i=1,\dots,D})\}$.

$$\begin{aligned} A^* = &\operatorname*{argmin}_A \texttt{cost}(\boldsymbol{x}, A) \qquad (3.3.1)\\ \text{s.t. } & h(\boldsymbol{x}^{\text{SCF}}) = 1, \\ & \boldsymbol{x}^{\text{SCF}} = \mathbb{I}_{i \in I} \boldsymbol{\delta}_i + \Big(\boldsymbol{x}_i + f_i(\mathbf{pa}_i^{\text{SCF}}) - f_i(\mathbf{pa}_i)\Big), \\ & A \in \mathcal{F} \end{aligned}$$

Due to the non-convexities introduced from `SoftSort`, we cannot solve this problem using convex optimisation (commonly used for constrained optimisation). Instead 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 $\boldsymbol{x}^{\text{SCF}}$ with a score of at least 0.5 (this can be changed for a given classifier score).

and any others such as the causal graph

$$\min_{A \in \mathcal{F}} \max_{\boldsymbol{\lambda}} \sum_{i=1}^{D} \texttt{cost}(\boldsymbol{x}, A) - \boldsymbol{\lambda}\Big(h(\boldsymbol{x}^{\text{SCF}}) - 0.5\Big) \tag{3.3.2}$$

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 $h(\boldsymbol{x}^{\text{SCF}})$ 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 $\boldsymbol{x}^{\text{SCF}}$. It is these feature values that are presented to the negatively classified individuals, as opposed to the actions themselves.

We only present $\boldsymbol{x}^{\text{SCF}}$ because, whilst we know that $h(\boldsymbol{x}^{\text{SCF}}) = 1$, we cannot guarantee that $h(\boldsymbol{x}^{\text{SCF}}) = 1$ unless there is perfect knowledge of the structural causal model. To demonstrate this, we will re-use the SCM from equation 3.1.1, which we denote as $\mathcal{M}^{\text{TRUE}}$.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(30, 5) \\ x_2 &= u_2 + 0.5x_1 & u_2 &\sim N(20, 10) \\ x_3 &= u_3 - 0.25x_2 & u_3 &\sim \text{Uniform}(0, 20) \\ x_4 &= \frac{x_3}{x_1} \end{aligned} \tag{3.3.3}$$

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.

$$x_2 = u_2 + x_1 \tag{3.3.4}$$

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 $\text{do}(\boldsymbol{x}_1 = 35)$, which results in $\boldsymbol{x}^{\text{SCF}} = [35, 30, 8.75, 0.25]^T$ 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 $\text{do}(\boldsymbol{x}_1 = 35)$ is $\boldsymbol{x}^{\text{SCF}} = [35, 27.5, 9.375, 0.27]^T$, 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.

4 Cost Learning

At the beginning of chapter 3, we noted that the cost function used in the literature is typically of the form $c(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^2$. Chapter 3 dealt with one of the limitations of this cost function - that it does not take into account the downstream effects of changing one feature on other features. In this chapter, we deal with another limitation of cost functions of this form - they fail to take into account user preferences over how difficult to manipulate different features are. 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[1] 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.

There do exist standard cost functions which take into account user preference, for example a quadratic form cost function $c(\boldsymbol{x}, \boldsymbol{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{M}(\mathbf{x} - \mathbf{x}')$ where $\mathbf{M}$ is a fixed, known, positive semi-definite square matrix (Bechavod et al., 2022). However, $\mathbf{M}$ must be selected in advance of generating recourse. In this chapter, we propose a methodology for *learning* user preferences.

Explain how $\mathbf{M}$ encodes user preference

[TO RE-WRITE THE ABOVE, TAKING INTO ACCOUNT THE INTRODUCTION]

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 revealed 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.

Rewrite this and expand, potentially state was WARP and SARP are.

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 $\boldsymbol{x}^*$, 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 $\boldsymbol{x}'$. 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 $\boldsymbol{x}$ to $\boldsymbol{x}'$. 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

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

feature is.

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 Set 1	**Action Set 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 sets of actions are easier to complete?
☐ Action Set 1
☐ Action Set 2

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

We assume that, *on average*, users have perfect knowledge of the SCM when evaluating which of the sets of actions they find least costly. Knowledge of the underlying structural causal model is crucial when evaluating actions, as actions may have causal effects of other variables and will affect the cost of other actions. For the same user preferences, the cost of the same actions could have different costs if there is a different underlying structural causal model. Consider a two variable SCM with income causing savings. In the first case, income has a very large positive effect on savings, and the in second case, income has a small positive effect on savings. In both cases the actions are to first increase income by £10,000 and then increase savings by £10,000. The cost of increasing income is the same in both cases, but the cost of increase savings is lower in the first case, as there is a large downstream effect of the increase in income on savings, meaning the additional amount to save is lower, and therefore, less costly in the first case.

justify this - feels a bit like an extension of rational choice theory except that it relates to the real world as opposed to their own preferences

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 $\boldsymbol{x}_n$ 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 individuals have perfect knowledge of the structural causal model. In this case, the individuals have a ground truth cost function $\texttt{cost}^G(\boldsymbol{x}, A|\beta, W)$ which is parameterised by β, how mutable each feature is and W, the parameters of the structural equations in the SCM. As the model deployer has perfect knowledge of the SCM, they do not need to learn W. As they are also aware of the actions being proposed to the individuals in each comparison, they also have A available to them. Therefore, in this case, the only variable they need to learn is the relative mutability of each feature β.

The model deployer's task is therefore to learn a β that which would explain 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 tanh function is used to force the predicted values $\hat{y}_{kn}$ into $[-1, 1]$.

$$\hat{y}_{kn} = \tanh\left(\lambda\Big(\texttt{cost}(\boldsymbol{x}_n, A^1_{kn}|\beta, W) - \texttt{cost}(\boldsymbol{x}_n, A^1_{kn}|\beta, W)\Big)\right) \tag{4.2.2}$$

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.

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

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 Imperfect Knowledge 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 as well as the parameters of the structural equations W. The formula for the predicted responses $\hat{y}_{kn}$ remains unchanged from equation 4.2.2 and we add W in to the objective function as follows.

$$\underset{\beta, W}{\operatorname{argmin}} = \frac{1}{KN}\sum_{k=1}^{K}\sum_{n=1}^{N}\ell(y_{kn}, \hat{y}_{kn}) + \underbrace{\lambda_1||\beta||_2 + \lambda_2||W||_2}_{\text{L2 regularisation}} \tag{4.2.4}$$

4.2.3 Noisy Responses

In the two previous set-ups the model deployer learns β (and W) from the responses of the negatively classified individuals, who have perfect knowledge of β and W. 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 by assuming that individuals knowledge of the SCM up to some noise and second by adding some noise to the response that they report.

To add noise to individuals' knowledge of the SCM, we assume that, instead of W, each individual's cost function is parameterised by $\tilde{W}$, where $\tilde{W} = W + N(0, \sigma^2_W)$. To add noise to the responses, we add some Gaussian noise with mean 0 and standard deviation σ_R to the ratio of the cost of actions A^1 to the cost of actions A^2.

explain the relevance of having noise with expectation=0

$$r = \frac{\texttt{cost}(\boldsymbol{x}_n, A^1_{kn}|\beta, \tilde{W})}{\texttt{cost}(\boldsymbol{x}_n, A^2_{kn}|\beta, \tilde{W})} \tag{4.2.5}$$

$$y_{kn} = \begin{cases} -1 & \text{if } r + N(0, \sigma_R^2) \leq 1 \\ +1 & \text{if } r + N(0, \sigma_R^2) > 1 \end{cases}$$

With these updated noisy responses, the optimisation problem still remains the same from the point of the model deployer. However, given the noisy responses, the role of the regularisation becomes more important.

4.3 Cost Function

In each of the following subsections, outlining the different cost functions used. Currently have implemented weighted squared costs (weights are feature mutability β).

4.3.1 Weighted Squared Costs

Let the individual cost function take the below form, where $\beta \in \mathbb{R}^D$ is a vector which expresses the mutability of each feature i. The order of the interventions in A does not matter in this case, as the cost function only depends on δ, not $\boldsymbol{x}$ (see Proposition 1).

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

5 Experiments

The end-to-end workflow is as follows:

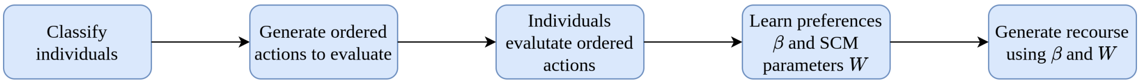

Figure 5.1: Workflow diagram

5.1 Simple synthetic data setting

In a simple synthetic data setting, data is generated from the below SCM.

$$\begin{aligned} x_1 &= u_1 & u_1 &\sim N(0,1) \\ x_2 &= 0.5x_1 + u_2 & u_2 &\sim N(0,1) \\ x_3 &= 0.2x_1 + 0.3x_2 + u_3 & u_3 &\sim N(0,0.5) \\ y &\sim \text{Bernoulli}(\sigma(0.1x_1 + 0.2x_2 + 0.3x_3)) \end{aligned} \tag{5.1.1}$$

Ground truth β: [0.5, 0.333, 0.1667]

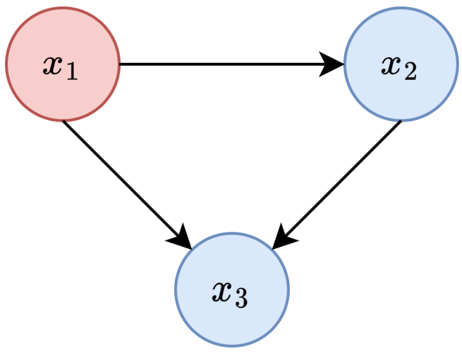

Figure 5.2: Simple SCM

5.1.1 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]

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

5.1.3 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

Amin, Kareem, Rachel Cummings, Lili Dworkin, Michael J. Kearns, and Aaron Roth (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 (visited on 07/17/2023) (Cited on page 8).

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).

Balcan, Maria-Florina, Amit Daniely, Ruta Mehta, Ruth Urner, and Vijay V. Vazirani (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. (Visited on 07/17/2023) (Cited on page 8).

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, 9, 17).

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. (Visited on 07/17/2023) (Cited on page 8).

Dong, Jinshuo, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu (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 page 8).

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 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, 7).

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 9, 11).

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 16).

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 page 10).

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 14).

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).

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 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 17).

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 17).

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, 12).

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).

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. (Visited on 07/17/2023) (Cited on page 8).