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: **August 16, 2023**

Abstract

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

Contents

1 Introduction

Introduction chapter.

2 Literature Review

2.1 Algorithmic Recourse

2.1.1 Motivation

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

2.1.2 Problem Set-up

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

$$\begin{aligned} \mathbf{x}^f = &\underset{\mathbf{x}' \in \mathcal{X}}{\operatorname{argmin}}\, c(\mathbf{x}, \mathbf{x}') \\ \text{s.t. } &h(\mathbf{x}') = 1, \\ &c(\mathbf{x}, \mathbf{x}') \leq B \end{aligned} \tag{2.1.1}$$

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

2.1.3 Recourse methods

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

2.2 Strategic Classification

2.2.1 Standard Strategic Classification

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

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

2.2.2 Causal Strategic Classification

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

2.3 Revealed Preferences

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

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

2.4 Pairwise Metric Learning

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

2.5 Canonical Datasets

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

- `Adult` - dataset to predict whether someone earns over $50,000 or more.
- `German Credit` - dataset identifies people as either good or bad credit risks.

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

3 Cost Learning

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

Add citations

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

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

To address through clustering

To address the first issue, that changing one feature can change the cost of changing another feature, we must model the effect of changing the feature $\mathbf{x}_1$ to $\mathbf{x'}_1$ has on changing the feature $\mathbf{x}_2$ to $\mathbf{x'}_2$. A structural causal model, such as the simple example in Figure 3.1, can be used to encode the downstream effects of changing one feature on other features. In the example in Figure 3.1, let x_1 represent salary, x_2 represent savings, u represent an unobserved variable causing savings (such as inherited savings) and y represent whether or not an individual was approved for a mortgage. If the individual receives an increase in salary, this will lead to an increase in savings, and both the increase in savings and income will increase the probability of mortgage approval. If the individual's initial features were $[£30,000, £50,000]^T$ and their salary increases to £35,000, then the cost of increasing savings to £65,000 are now lower than when their salary was £30,000.

When incorporating a causal graph into the cost function, it is more appropriate to express the cost function as the cost of *interventions* $\mathbf{a} = [a_1, a_2]^T$ with

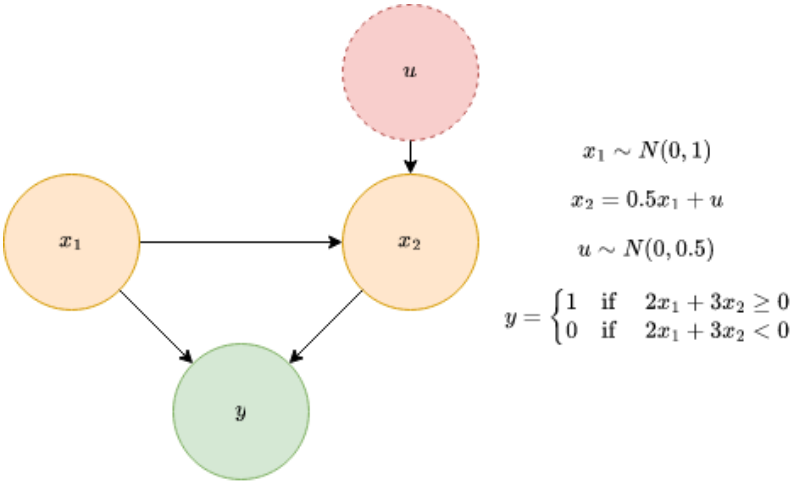

Figure 3.1: A simple structural causal model

Add in worked example showing ordering matters in cost function.

ordering $\mathbf{o} = [o_1, o_2]$, as opposed to cost of changing features $\mathbf{x}$ to $\mathbf{x}'$. The ordering dictates the order in which in the features are intervened upon, and the interventions dictate the size of the intervention. The causal graph can be expressed as a weighted adjacency matrix W, where W_{ij} is the marginal effect of intervening on $\mathbf{x}_i$ on $\mathbf{x}_j$. The cost of the interventions $\mathbf{a}$, given ordering $\mathbf{o}$ and weighted adjacency matrix W, is therefore the sum of a cost function over each individual intervention $\mathbf{a}_i$ (for example $c(\mathbf{a}_i) = \mathbf{a}_i^2$). The cost of the intervention can be expressed as follows, where D is the number of features.

$$c(\mathbf{a}) = \sum_{i=1}^{D} c(\mathbf{a}_i) \tag{3.0.1}$$

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

Algorithm 1 Intervention Evaluation Function

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

3.1 Generating recourse

Using the new formulation of the cost function as the cost of interventions, we can also re-write the recourse generation problem, given a classifier h, original features x and weighted adjacency matrix W. We denote the intervention evaluation as described in Algorithm 1 as `eval`. For negatively classified

individuals, the task of recourse involves solving equation 3.1.1, where we minimise the cost of intervention subject to the new features $\mathbf{x}'$ leading to a positive classification (which corresponds to a classification score of greater than 0.5).

$$\mathbf{a}', \mathbf{o}' = \underset{\mathbf{a},\mathbf{o}}{\operatorname{argmin}} \sum_{i=1}^{D} c(\mathbf{a}_i) \quad \text{s.t. } h(\texttt{eval}(\mathbf{a}, \mathbf{x}, \mathbf{o}, W)) \geq 0.5 \tag{3.1.1}$$

As `eval` is a non-convex function, we cannot solve this constrained optimisation problem using convex optimisation. Instead, is converted into a unconstrained problem using Lagrange multipliers as shown in equation 3.1.2. This is solved using gradient descent, where at each iteration, a step is first taken to maximise λ and then a step is then taken to minimise $\mathbf{a}$ and $\mathbf{o}$.

$$\min_{\mathbf{a},\mathbf{o}} \max_{\lambda} \sum_{i=1}^{D} c(\mathbf{a}_i) - \lambda(h(\texttt{eval}(\mathbf{a}, \mathbf{x}, \mathbf{o}, W)) - 0.5) \tag{3.1.2}$$

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

3.1.1 Differentiable sorting

The key point to make in this section is that technically, as long as a function is differentiable and maps from $f : \mathbb{R}^D \rightarrow \texttt{ordering}^D$, we should be able to take derivatives of f and use gradient descent to find the value of the input to f which minimises the objective defined in equation 3.1.2. The smoother the function is in its mapping from a vector of real numbers to the ordering (i.e., vectors close to each other map to similar orderings), the fewer local minima it should have, making optimisation easier.

3.2 Learning from revealed preferences

To re-write below this point.

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

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

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

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

Once a cost function is learned, we need to solve the optimisation problem mentioned in equation 3.1.2 to generate the recourse $(\mathbf{a}', \mathbf{o}')$.

3.2.1 Weighted Squared Costs

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

Seems much more efficient to assume a fixed ordering

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

$$y_n = \begin{cases} -1 & \text{if } \sum_{i=1}^{D} c(\mathbf{a}_{ni}^1) \leq \sum_{i=1}^{D} c(\mathbf{a}_{ni}^2) \\ +1 & \text{if } \sum_{i=1}^{D} c(\mathbf{a}_{ni}^1) > \sum_{i=1}^{D} c(\mathbf{a}_{ni}^2) \end{cases} \tag{3.2.1}$$

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

$$\underset{\beta}{\text{argmin}} = \frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \ell(y_{kn}, \hat{y}_{kn}(\beta)) + \underbrace{\lambda ||\beta||_2}_{\text{L2 regularisation}} \tag{3.2.2}$$

This is an unconstrained optimised problem that can be optimised via gradient descent. However, operation in equation 3.2.1 is non, differentiable, so instead it is approximated with the below expression, which fits $\hat{y}_n$ into [-1,1] where λ is a hyperparameter regularising for 'confidence' of the predictions.

$$\hat{y}_n = \tanh\left(\lambda\Big(\sum_{i=1}^{D} c(\mathbf{a}_{ni}^1) - \sum_{i=1}^{D} c(\mathbf{a}_{ni}^2)\Big)\right) \tag{3.2.3}$$

3.3 Mahalanobis distance

The Mahalanobis distance between the vector $\mathbf{x}$ and the vector $\mathbf{y}$ is defined in equation 3.3.1, where M is a positive semi-definite matrix.

$$||\mathbf{x} - \mathbf{y}||_{\mathbf{M}} = \sqrt{(\mathbf{x} - \mathbf{y})^T \mathbf{M}^{-1} (\mathbf{x} - \mathbf{y})} \tag{3.3.1}$$

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

3.3.1 Learning the Mahalanobis distance

In order to use the Mahalanobis distance as a cost function, we must learn the matrix $\mathbf{M}$. In this set-up, each individual k with original features $\mathbf{x}_k$ is presented with N recourse options $(\mathbf{x}_{kn}^a, \mathbf{x}_{kn}^b)$. The response y_{kn} is defined in equation 3.3.2, where c_k^G represents the ground truth cost function of individual k.

$$y_{kn} = \begin{cases} -1 & \text{if } c_k^G(\mathbf{x}_{kn}, \mathbf{x}_{kn}^a) \leq c_k^G(\mathbf{x}_{kn}, \mathbf{x}_{kn}^b) \\ +1 & \text{if } c_k^G(\mathbf{x}_{kn}, \mathbf{x}_{kn}^a) > c_k^G(\mathbf{x}_{kn}, \mathbf{x}_{kn}^b) \end{cases} \tag{3.3.2}$$

To optimise for $\mathbf{M}$, we compare the squared Mahalanobis distances between $\mathbf{x}_k$ and $\mathbf{x}_{kn}^a$ and between $\mathbf{x}_k$ and $\mathbf{x}_{kn}^b$. The optimisation problem to learn $\mathbf{M}$ is shown in equation 3.3.3, where ℓ represents either the hinge or logistic loss function. The optimisation is an adaptation of the optimisation problem presented in Canal et al. (2022).

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

$$\begin{aligned} &\min_{\mathbf{M}} \frac{1}{KN} \sum_{k=1}^{K} \sum_{n=1}^{N} \ell\Big(y_{kn}(||\mathbf{x}_k - \mathbf{x}_{kn}^a||_{\mathbf{M}}^2 - ||\mathbf{x}_k - \mathbf{x}_{kn}^b||_{\mathbf{M}}^2)\Big) \\ &\text{s.t. } \mathbf{M} \succeq 0, \end{aligned} \tag{3.3.3}$$

3.4 Convex layers

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

4 Experiments

4.1 Synthetic data

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

The experiments that follow use synthetic data generated from a structural causal model, which is shown in Figure 4.1 .

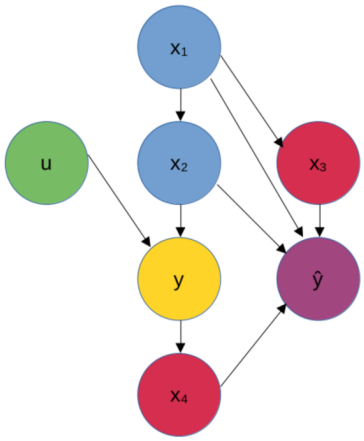

Figure 4.1: The Structural Causal Model used for synthetic data generation

The structural causal model contains 2 unobserved variables

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

And 4 observed variables

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

4.2 Simulation process

[**TO CONVERT TO AN ALGORITHM/PSEUDOCODE**]

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

1. Split the data into test and train sets.

2. Fit a classifier using just the train set and measure accuracy against the *true* labels.

3. Predict labels for *all* data points (both train and test)

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

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

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

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

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

4.3 Mahalanobis distance

References

Agrawal, Akshay et al. (2019). "Differentiable Convex Optimization Layers". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 9558–9570. URL: https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html (Cited on page 13).

Ahmadi, Saba et al. (2022). "On Classification of Strategic Agents Who Can Both Game and Improve". In: *3rd Symposium on Foundations of Responsible Computing (FORC 2022)*. Vol. 218. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 3:1–3:22. DOI: 10.4230/LIPIcs.FORC.2022.3 (Cited on page 6).

Amin, Kareem et al. (2015). "Online Learning and Profit Maximization from Revealed Preferences". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, pp. 770–776. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9984 (Cited on page 7).

Balcan, Maria-Florina et al. (2014). "Learning Economic Parameters from Revealed Preferences". In: *Web and Internet Economics - 10th International Conference, WINE 2014, Beijing, China, December 14-17, 2014. Proceedings*. Vol. 8877. Springer, pp. 338–353. DOI: 10.1007/978-3-319-13129-0_28 (Cited on page 7).

Bechavod, Yahav et al. (2022). "Information Discrepancy in Strategic Learning". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162. PMLR, pp. 1691–1715. URL: https://proceedings.mlr.press/v162/bechavod22a.html (Cited on pages 6, 9).

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

Canal, Gregory et al. (2022). "One for All: Simultaneous Metric and Preference Learning over Multiple Users". In: *Advances in Neural Information Processing Systems* 35, pp. 4943–4956. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/1fd4367793bcd3ad38a0b820fcc1b815-Abstract-Conference.html (Cited on pages 7, 13).

Chen, Yiling, Yang Liu, and Chara Podimata (2020). "Learning Strategy-Aware Linear Classifiers". In: *Advances in Neural Information Processing Systems*

33. URL: https://proceedings.neurips.cc/paper/2020/hash/ae87a54e183c075c494c4d397d126a66-Abstract.html (Cited on page 6).

Dong, Jinshuo et al. (2018). "Strategic Classification from Revealed Preferences". In: *Proceedings of the 2018 ACM Conference on Economics and Computation*. Ithaca, NY, USA, pp. 55–70. DOI: 10.1145/3219166.3219193 (Cited on pages 6, 7).

Eilat, Itay et al. (2023). "Strategic Classification with Graph Neural Networks". In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. URL: https://openreview.net/pdf?id=TuHkVOjSAR (Cited on page 7).

Ghalme, Ganesh et al. (2021). "Strategic Classification in the Dark". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 3672–3681. URL: http://proceedings.mlr.press/v139/ghalme21a.html (Cited on page 6).

Hardt, Moritz et al. (2016). "Strategic Classification". In: *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science* (Cambridge, Massachusetts, USA). New York, NY, USA: Association for Computing Machinery, pp. 111–122. DOI: 10.1145/2840728.2840730 (Cited on page 6).

Jagadeesan, Meena, Celestine Mendler-Dünner, and Moritz Hardt (2021). "Alternative Microfoundations for Strategic Classification". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 4687–4697. URL: http://proceedings.mlr.press/v139/jagadeesan21a.html (Cited on page 6).

Karimi, Amir-Hossein, Gilles Barthe, et al. (2022). "A Survey of Algorithmic Recourse: Contrastive Explanations and Consequential Recommendations". In: *ACM Comput. Surv.* 55.5. DOI: 10.1145/3527848 (Cited on page 6).

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 page 6).

Levanon, Sagi and Nir Rosenfeld (2021). "Strategic Classification Made Practical". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. PMLR, pp. 6243–6253. URL: http://proceedings.mlr.press/v139/levanon21a.html (Cited on page 6).

Levanon, Sagi and Nir Rosenfeld (2022). "Generalized Strategic Classification and the Case of Aligned Incentives". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162.

PMLR, pp. 12593–12618. URL: https://proceedings.mlr.press/v162/levanon22a.html (Cited on page 7).

Nair, Vineet et al. (2022). "Strategic Representation". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Vol. 162. PMLR, pp. 16331–16352. URL: https://proceedings.mlr.press/v162/nair22a.html (Cited on page 7).

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 7).

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 6).

Zadimoghaddam, Morteza and Aaron Roth (2012). "Efficiently Learning from Revealed Preference". In: *Internet and Network Economics - 8th International Workshop, WINE 2012, Liverpool, UK, December 10-12, 2012. Proceedings*. Vol. 7695. Springer, pp. 114–127. DOI: 10.1007/978-3-642-35311-6_9 (Cited on page 7).

Zrnic, Tijana et al. (2021). "Who Leads and Who Follows in Strategic Classification?" In: *Advances in Neural Information Processing Systems 34*, pp. 15257–15269. URL: https://proceedings.neurips.cc/paper/2021/hash/812214fb8e7066bfa6e32c626c2c688b-Abstract.html (Cited on page 7).