Actionable Recourse in Linear Classification

Alexander Spangher [*1] **Berk Ustun** [*2]

Abstract

In machine learning, recourse refers to the ability to achieve a desired outcome under a fixed prediction model. In this paper, we present a new approach to audit the recourse of linear classification models. Given a linear classifier, we formulate an optimization problem to find an actionable set of changes that an individual can make to achieve a desired outcome. We then solve our problem to: (i) evaluate the cost and feasibility of recourse of the classifier over a target population; and (ii) generate a list of informative changes for an individual to flip their assigned prediction. We discuss the need to audit recourse through experiments on a credit scoring problem, where we show how common modeling practices can significantly alter the cost and feasibility of recourse of a classifier without affecting its performance.

1. Introduction

In machine learning, *recourse* refers to the ability to achieve a desired outcome under a fixed prediction model. Consider, for example, a classifier built to automate lending decisions. If this model does not provide recourse to a person who is denied a loan, then this person cannot change any of the input variables of the model to be approved for a loan, and will be denied credit so long as the model is deployed.

A prediction model should provide all individuals with actionable recourse to all individuals when they are used to allocate goods that should be universally accessible, such as credit (Siddiqi, 2012), employment (Ajunwa et al., 2016) and social services (Chouldechova et al., 2018). The potential lack of recourse in such applications often motivates calls for transparency in algorithmic decision-making (see e.g., Citron & Pasquale, 2014; Wachter et al., 2017; Doshi-Velez et al., 2017). However, transparency does not guarantee recourse. In practice, even simple transparent models such as linear classifiers can fail to provide an individual with recourse due to common modeling decisions that are difficult to regulate, including:

- *Feature Selection*: A model could use features that are immutable (e.g. *female*), conditionally immutable (e.g. *has_phd*, which can only change from `FALSE` $\rightarrow$ `TRUE`), or should not be considered actionable (e.g. *married*).
- *Choice of Operating Point*: A probabilistic classifier that provides recourse at standard threshold (e.g., $\hat{y}_i = 1$ if predicted risk $\geq 50\%$) could fail to do so at a more conservative threshold (e.g., $\hat{y}_i = 1$ if predicted risk $\geq 80\%$).
- *Out-of-Sample Deployment*: A feature needed for recourse could be missing for individuals in the target population.

Without a formal procedure to audit recourse, we can easily deploy a model that precludes individuals from achieving a desired outcome.

In this paper, we present a new approach to audit recourse for linear classification models (e.g., logistic regression models, linear SVMs, and linearizable boolean models such as rule sets and decision lists). We formulate an optimization problem to find an actionable set of changes that an individual can make to flip the prediction of a given linear classifier. Our problem is specifically designed to find changes that are are *actionable*, so they do not affect immutable features or alter mutable features in an infeasible way (e.g., *n_credit_cards* from $5 \rightarrow 0.5$ or -1, or *has_phd* from `TRUE` $\rightarrow$ `FALSE`). Since such constraints are often discrete, we express our problem as an *integer program* (IP), which can quickly recover a globally optimal set of actions to attain a desired outcome or a certificate to state that the model does not provide actionable recourse.

We solve our IP to design two auditing tools:

1. A procedure to evaluate the feasibility and cost of recourse of the classifier for all individuals in a target population (for model development, procurement, or 3rd party audits such as algorithmic impact assessments, Dillon Reisman, 2018). When our optimization problem is infeasible, this certifies that there is no change that a person to attain the desired outcome (i.e., the classifier does not provide actionable recourse for this person). Accordingly, we can certify that a model provides recourse

[*]Equal contribution [1]New York Times, NY, USA [2]Center for Research in Computation and Society, Harvard University, MA, USA. Correspondence to: Berk Ustun <berk@seas.harvard.edu>.

Proceedings of the 5^{th} Workshop on Fairness, Accountability and Transparency in Machine Learning, Stockholm, Sweden, 2018.

to individuals in a target sample by solving our problem for each point in the sample. By comparing the cost of recourse, we can assess the difficulty of changes required for individuals to achieve a desired outcome.

2. A method to generate a list of actionable changes for an individual to flip the prediction of the classifier. We refer to this list as a *flipset* and show an example in Figure 1. In the United States, for example, the Fair Credit Reporting Act (U.S. Congress, 2003) requires sending an *adverse action notice* to individuals who are denied credit from a prediction model to explain the principal reason for the denial. By including a flipset in an adverse action notice, an individual would know exact changes to guarantee approval the future (see also Taylor, 1980; Selbst & Barocas, 2018, for a discussion of how adverse action notices fail to provide actionable information).

FEATURES TO CHANGE	CURRENT VALUES		REQUIRED VALUES
n_credit_cards	5	$\longrightarrow$	3
current_debt	\$3,250	$\longrightarrow$	\$1,000
has_savings_account	FALSE	$\longrightarrow$	TRUE
has_retirement_account	FALSE	$\longrightarrow$	TRUE

Figure 1. Illustrative flipset for an individual who is denied credit by a classifier. Each *item* (i.e. row) shows an actionable set of changes to a subset of features to "flip" the prediction from $\hat{y} = -1$ to $\hat{y} = +1$. These changes guarantee that the individual will be approved for credit so long as other features do not change.

RELATED WORK

Our work is a new application for *inverse classification* (Aggarwal et al., 2010), which aims to determine how the inputs to a prediction model can be manipulated to obtain a desired outcome (see e.g., Chang et al., 2012; Yang et al., 2012, for other applications).

Our work is broadly related to tools to explain the predictions of machine learning models (see e.g., Ribeiro et al., 2016). While such tools can provide valuable explanations of how a model outputs a specific prediction, these explanations do not correspond to actionable changes that can be used to reliably attain a desired outcome. Moreover, the tools do not provide a formal guarantee for an auditor to certify that an actionable set of changes does not exist.

Our ideas are also related to seminal work on counterfactual explanations by Wachter et al. (2017)[1]. In particular, our tools solve an optimization problem to recover counterfactual explanations that are actionable and globally optimal with respect to a user-specified cost function. Our problem is fundamentally different from the one proposed by Wachter et al. (2017). Their approach can extract counterfactual explanations from black-box models, but does not provide the feasibility or optimality guarantees to audit recourse because: (i) it cannot constrain changes to be actionable; (ii) it restricts feasible changes to differences between points in the training data (i.e., $\boldsymbol{a} \in \{\boldsymbol{x} - \boldsymbol{x}' \text{ for } \boldsymbol{x}, \boldsymbol{x}' \text{ in the training data}\}$) [2].

Other concepts related to recourse include: *anchors*, which are subsets of features that *fix* predicted the outcome (Ribeiro et al., 2018); and *strategic classification*, which considers the converse problem of training classifiers that are robust to manipulation (Hardt et al., 2016).

2. Problem Statement

We consider a standard classification setting where each individual is characterized by features $\boldsymbol{x} = [1, x_1 \dots x_d] \subseteq \mathcal{X}_0 \cup \dots \cup \mathcal{X}_d \subseteq \mathbb{R}^{d+1}$ and a label $y \in \{-1, +1\}$.

We will audit a linear classifier $f(\boldsymbol{x}) = \text{sign}\,(\langle \boldsymbol{w}, \boldsymbol{x} \rangle)$ where $\boldsymbol{w} = [w_0, w_1 \dots w_d] \subseteq \mathbb{R}^{d+1}$ is a coefficient vector and w_0 is the intercept. We denote the desired outcome as $\hat{y} = 1$ and assume $\text{sign}\,(0) = 1$ so that $\hat{y} = \mathbb{1}\,[\langle \boldsymbol{w}, \boldsymbol{x} \rangle \geq 0]$.

Given an individual such that $f(\boldsymbol{x}) = -1$, we determine if there exists an *action* $\boldsymbol{a}$ such that $f(\boldsymbol{x} + \boldsymbol{a}) = 1$ by solving an optimization problem of the form,

$$\begin{aligned} \min \quad & cost(\boldsymbol{a}; \boldsymbol{x}) \\ \text{s.t.} \quad & f(\boldsymbol{x} + \boldsymbol{a}) = 1 \\ & \boldsymbol{a} \in A(\boldsymbol{x}). \end{aligned} \tag{1}$$

Here:

- $A(\boldsymbol{x})$ is a set of feasible actions $\boldsymbol{a} = [0, a_1 \dots a_d]$ from $\boldsymbol{x}$. We constrain each element of $\boldsymbol{a}$ to produce a feasible feature $a_j \in A_j(x_j) \subseteq \{a_j \in \mathbb{R} \mid a_j + x_j \in \mathcal{X}_j\}$. We let $A_j(x_j) = \{0\}$ if feature j is immutable.
- $cost(\cdot\,; \boldsymbol{x}) : A(\boldsymbol{x}) \to \mathbb{R}_+$ is a user-specified cost function that satisfies the following properties: (i) $cost(\boldsymbol{x}; \boldsymbol{x}) = 0$ (no action $\Leftrightarrow$ no cost); (ii) $cost(\boldsymbol{a}; \boldsymbol{x}) \leq cost(\boldsymbol{a} + \epsilon \mathbf{1}_j; \boldsymbol{x})$ (larger actions $\Leftrightarrow$ higher cost).

If (1) is *infeasible*, then no action can achieve a desired outcome from $\boldsymbol{x}$, and thus we have certified that the model does not provide actionable recourse for this person. If (1) is *feasible*, then its optimal solution is the minimal-cost action to flip the prediction of $\boldsymbol{x}$. In this case, we use the solution to create the first item in a flipset and enumerate additional items as described in Section 3.2.

[1] Given a model and an example, a *counterfactual explanation* is the smallest set of changes to features to obtain a desired outcome.

[2] To illustrate some practical consequences of (i) and (ii): the proposed approach could output an explanation that states that a person can flip their prediction by changing an immutable feature, due to (i). If so, an auditor could not conclude that the model did not provide recourse, as there could exist a way to flip the prediction that was not reflected in the training data, due to (ii).

3. Integer Programming Approach

We consider a discrete version of the optimization problem in (1), which we express as an *integer program* (IP) and solve with an IP solver (see Mittleman, 2018, for a list). Our approach has several key benefits: (i) it can directly constrain actions for discrete-valued features (e.g., binary, categorical, ordinal); (ii) it can minimize non-linear and non-convex cost functions (as we can precompute costs and pass them to our IP via the c_{jk} parameters in (2a)); (iii) it allows users to customize the set of feasible actions; (iv) it can quickly recover a globally optimal solution or certify that actionable recourse does not exist. The main shortcoming of this approach is that it requires discretizing changes to real-valued features. To ensure discretization does not affect the cost or feasibility of recourse, we must therefore discretize the actions for such features over a suitably fine grid.

3.1. IP Formulation

Our IP has the form:

$$
\begin{aligned}
\min \quad & \mathit{cost} \\
\text{s.t.} \quad & \mathit{cost} = \sum_{j=1}^{d} \sum_{k=1}^{m_j} c_{jk} v_{jk} && && \text{(2a)} \\
& \sum_{j=1}^{d} w_j a_j \geq \sum_{j=0}^{d} w_j x_j && && \text{(2b)} \\
& a_j = \sum_{k=1}^{m_j} a_{jk} v_{jk} && j = 1 \ldots d && \text{(2c)} \\
& 1 = u_j + \sum_{k=1}^{m_j} v_{jk} && j = 1 \ldots d && \text{(2d)} \\
& a_j \in \mathbb{R} && j = 1 \ldots d \\
& u_j, v_{jk} \in \{0, 1\} && j = 1 \ldots d \quad k = 1 \ldots m_j
\end{aligned}
$$

Here, constraint (2a) sets the cost of a feasible action via the precomputed cost parameters $c_{jk} := \mathit{cost}(x_j + a_{jk}; x_j)$. Constraint (2b) ensures that any feasible action will flip the prediction of a linear classifier with coefficients $\boldsymbol{w}$. Constraints (2c) and (2d) restrict a_j to a grid of $m_j + 1$ feasible values $a_j \in \{0, a_{j1} \ldots a_{jm_j}\}$ via the indicator variables $u_j = 1[a_j = 0]$ and $v_{jk} = 1[a_j = a_{jk}]$.

Customization: We can customize the feasible action set by adding logical constraints to (2). Many such constraints can be expressed with the u_j indicators. To limit actions to change $\leq R$ features, we can add the constraint $\sum_{j=1}^{d}(1 - u_j) \leq R$. To ensure actions only change feature p or q not both, we can add the constraint $(1 - u_p) + (1 - u_q) \leq 1$.

Speedups: Although modern IP solvers can quickly solve instances of (2) ($\leq$ 1s with CPLEX 12.8), we can further reduce the solution time (i.e. for auditing procedures) by: (i) dropping constraints (2c) and (2d) for non-actionable features; (ii) dropping v_{jk} indicators for actions a_{jk} that do not agree in sign with w_j; (iii) declaring $\{v_{j1} \ldots v_{jm_j}\}$ as a special ordered set, which allows the solver to use a more efficient branch-and-bound algorithm for these variables.

3.2. Building Flipsets

The optimal solution to (2) can be used to create the first item in a flipset (i.e., by listing the values of x_j and $x_j + a_j^*$ for all j such that $a_j^* \neq 0$). In order to effectively provide an individual with recourse, however, a flipset should contain multiple items. This is because each item may be infeasible in a way that is only known to the individual.

To build a flipset with multiple items, we use an *enumeration procedure* that repeatedly solves (2). Our proposed procedure recovers $T \geq 2$ actions that use distinct subsets of features by repeating the following steps T times: (i) solve (2); (ii) use the optimal action $\boldsymbol{a}^*$ to add a new item to flipset; (iii) add a constraint to eliminate the active set of changes $S = \{j : a_j^* \neq 0\}$ from the feasible set $\sum_{j \in S}(1 - u_j) + \sum_{j \notin S} u_j \leq d - 1$.

3.3. Choosing a Cost Function

While users can design their own cost functions, we propose two generic functions for auditing and building flipsets. Both functions measure costs using the *percentiles* of x_j and $x_j + a_j$ in the target population; that is, $Q_j(x_j + a_j)$ and $Q_j(x_j)$ where $Q_j(\cdot)$ is the CDF of x_j . This standardizes the cost of changes across features and ensures that costs reflect the distribution of features in the target population.

For auditing applications, we propose optimizing the *maximum percentile shift*

$$
\mathit{cost}(\boldsymbol{x} + \boldsymbol{a}; \boldsymbol{x}) = \max_{j=1 \ldots d} \left| Q_j(x_j + a_j) - Q_j(x_j) \right|. \quad (3)
$$

Our choice is motivated by the interpretation of the optimal cost under (3). If the optimal cost is 0.25, for example, then this means that all feasible action must change a feature by at least 25 percentiles (i.e., no feasible action can flip the prediction without changing a feature by < 25 percentiles). To use (3), one must replace constraint (2a) with the constraints $\mathit{cost} \geq \sum_{k=1}^{m_j} c_{jk} v_{jk}$ for $j = 1 \ldots d$.

For building flipsets, we propose optimizing the *total log-percentile shift*:

$$
\mathit{cost}(\boldsymbol{x} + \boldsymbol{a}; \boldsymbol{x}) = \sum_{j : a_j \neq 0} \log \left(\frac{1 - Q_j(x_j + a_j)}{1 - Q_j(x_j)} \right). \quad (4)
$$

In this case, our choice aims to select items that may reflect "easy" changes with respect to the target population. In particular, (4) ensures that the cost that changing feature j by a_j increases exponentially as $Q_j(x_j) \to 1$. This captures the notion that changes become harder at higher percentiles (e.g., changing *income* from percentiles $50 \to 55$ is easier than $90 \to 95$).

4. Demonstration

We now demonstrate how our tools could be used to audit the recourse of linear classifiers in a hypothetical credit scoring problem. We provide a software implementation of our tools and scripts to reproduce the analysis in this section at http://github.com/ustunb/actionable-recourse.

Setup

Data: We consider a processed version of `credit` dataset from the UCI Repository (Yeh & Lien, 2009). Here, $y_i = -1$ if person i will default on an upcoming credit card payment. Our dataset contains $n = 30\,000$ individuals and $d = 16$ features related to spending and payment patterns, education, credit history, age, and marital status. We assume spending and payment patterns and education are actionable, and consider all other variables to be immutable.

Model Training and Auditing: We train ℓ_1-penalized logistic regression (LR) models for values of the ℓ_1-penalty in the set $\{1, 2, 5, 10, 20, 50, 100, 500, 1000\}$ and estimate their test error via stratified 10-fold CV. We audit the recourse of each classifier using the training data as our target sample by solving (2) for each individual i such that $\hat{y}_i = -1$. Our IP uses the cost function in (3) and include the following constraints to ensure changes are actionable: (i) changes for discrete features must be discrete (e.g. *MonthsWithLowSpendingInPast6Months* $\in \{0, 1 \dots 6\}$); (ii) *EducationLevel* can only increase; (iii) immutable features cannot change.

Results

We summarize our audit in Figure 3 and present a flipset for an individual who was denied credit in 2.

As shown, tuning the ℓ_1-penalty has a minor effect on test error, but significantly affects the cost and feasibility of recourse. Here, classifiers with small ℓ_1-penalties provide all individuals with recourse. As the ℓ_1-penalty increases, however, the % of individuals with recourse falls as the coefficients for actionable features are more heavily penalized in comparison to those for immutable features. Among the individuals who retain recourse, we observe that increasing the ℓ_1-penalty almost doubles the median cost of recourse from 0.20 to 0.39. Since we have used the cost function in (3), a cost of q implies an individual must change a feature by at least q percentiles to attain a desired outcome.

Our aim is not to suggest a relationship between recourse and ℓ_1-regularization, but to show how seemingly innocuous practices such as parameter tuning can impact the cost and feasibility of recourse. Here, a practitioner who is primarily interested in performance could deploy a classifier that precludes individuals from achieving a desired outcome (e.g., the one that minimizes mean 10-CV test error), even as there exists a classifier that attains similar performance but provides all individuals with recourse. Other practices that affect recourse include preprocessing, using a more conservative decision point, or evaluating recourse on a hold-out set. In practice, it is unlikely that such practices can be effectively regulated.

Feature Subset	Current Values		Required Values
MostRecentPaymentAmount	\$0	$\longrightarrow$	\$790
MostRecentPaymentAmount	\$0	$\longrightarrow$	\$515
MonthsWithZeroBalanceOverLast6Months	1	$\longrightarrow$	2
MonthsWithZeroBalanceOverLast6Months	1	$\longrightarrow$	4
MostRecentPaymentAmount	\$0	$\longrightarrow$	\$775
MonthsWithLowSpendingOverLast6Months	6	$\longrightarrow$	5
MostRecentPaymentAmount	\$0	$\longrightarrow$	\$500
MonthsWithLowSpendingOverLast6Months	6	$\longrightarrow$	5
MonthsWithZeroBalanceOverLast6Months	1	$\longrightarrow$	2

Figure 2. Flipset for a person who is denied credit by the most accurate classifier. Each item describes a set of actionable minimal-cost changes for the individual to attain the desired outcome. We enumerated all 5 items in ≤ 1 second using the cost function in 4 and the enumeration scheme in Section 3.2.

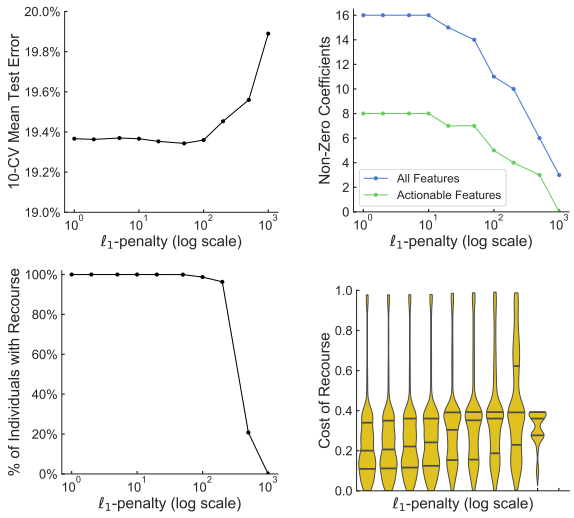

Figure 3. Model performance and recourse over the training sample for ℓ_1-penalized LR classifiers. We show the mean 10-CV test error (top left), # of non-zero coefficients (top right), % of individuals with recourse (bottom left), and the distribution of the cost of recourse (bottom left) for all classifiers.

5. Discussion

We have presented a new approach to study recourse in machine learning. Our approach allows regulators to certify that a linear classifier provides actionable recourse within a target population, and produce informative lists to help individuals achieve a desired outcome. In future work, we aim to extend our framework to audit non-linear classifiers and derive out-of-sample guarantees for a classifier will provide actionable recourse.

References

Aggarwal, Charu C, Chen, Chen, and Han, Jiawei. The inverse classification problem. *Journal of Computer Science and Technology*, 25(3):458–468, 2010.

Ajunwa, Ifeoma, Friedler, Sorelle, Scheidegger, Carlos E, and Venkatasubramanian, Suresh. Hiring by algorithm: predicting and preventing disparate impact. *Available at SSRN*, 2016.

Chang, Allison, Rudin, Cynthia, Cavaretta, Michael, Thomas, Robert, and Chou, Gloria. How to reverse-engineer quality rankings. *Machine learning*, 88(3):369–398, 2012.

Chouldechova, Alexandra, Benavides-Prado, Diana, Fialko, Oleksandr, and Vaithianathan, Rhema. A case study of algorithm-assisted decision making in child maltreatment hotline screening decisions. In *Conference on Fairness, Accountability and Transparency*, pp. 134–148, 2018.

Citron, Danielle Keats and Pasquale, Frank. The scored society: due process for automated predictions. *Wash. L. Rev.*, 89:1, 2014.

Dillon Reisman, Jason Schultz, Kate Crawford Meredith Whittaker. Algorithmic impact assessments: A practical framework for public agency accountability. AI Now Technical Report, April 2018.

Doshi-Velez, Finale, Kortz, Mason, Budish, Ryan, Bavitz, Chris, Gershman, Sam, O'Brien, David, Schieber, Stuart, Waldo, James, Weinberger, David, and Wood, Alexandra. Accountability of ai under the law: The role of explanation. *arXiv preprint arXiv:1711.01134*, 2017.

Hardt, Moritz, Megiddo, Nimrod, Papadimitriou, Christos, and Wootters, Mary. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pp. 111–122. ACM, 2016.

Mittleman, Hans. Mixed integer linear programming benchmark (miplib2010). http://plato.asu.edu/ftp/milpc.html, 2018.

Ribeiro, Marco Tulio, Singh, Sameer, and Guestrin, Carlos. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016.

Ribeiro, Marco Tulio, Singh, Sameer, and Guestrin, Carlos. Anchors: High-precision model-agnostic explanations. In *AAAI Conference on Artificial Intelligence*, 2018.

Selbst, Andrew D and Barocas, Solon. The intuitive appeal of explainable machines. 2018.

Siddiqi, Naeem. *Credit risk scorecards: developing and implementing intelligent credit scoring*, volume 3. John Wiley & Sons, 2012.

Taylor, Winnie F. Meeting the equal credit opportunity act's specificity requirement: Judgmental and statistical scoring systems. *Buff. L. Rev.*, 29:73, 1980.

U.S. Congress. The fair and accurate credit transactions act, 2003.

Wachter, Sandra, Mittelstadt, Brent, and Russell, Chris. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. 2017.

Yang, Chen, Street, Nick W, and Robinson, Jennifer G. 10-year cvd risk prediction and minimization via inverse-classification. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pp. 603–610. ACM, 2012.

Yeh, I-Cheng and Lien, Che-hui. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480, 2009.