CRUDS: Counterfactual Recourse Using Disentangled Subspaces

Michael Downs [1 2] **Jonathan L. Chu** [1] **Yaniv Yacoby** [1] **Finale Doshi-Velez** [1] **Weiwei Pan** [1]

Abstract

Algorithmic recourse is the task of generating a set of actions that will allow individuals to achieve a more favorable outcome under a given algorithmic decision system. Using the Conditional Subspace Variational Autoencoder (CSVAE), we propose a novel algorithmic recourse generation method, CRUDS, that generates multiple recourse satisfying underlying structure of the data as well as end-user specified constraints. We evaluate our method qualitatively and quantitatively on several synthetic and real datasets, demonstrating that CRUDS proposes recourse that are more realistic and actionable than baselines.

1. Introduction

Machine learning decision systems are increasingly being applied to domains like finance and criminal justice (Dressel & Farid, 2018), where outcomes have a significant social and human impact. Often in these domains, the focus lies not only in the outcomes themselves, but in the set of options available to individuals wishing to improve unfavorable outcomes. For instance, if a loan applicant is issued a denial, the main point of interest is how the applicant might adjust their applicant profile to increase future chances for approval. This set of changes is known in the literature as *recourse*. When the decision system is a complex machine learning model, we want to find ways of algorithmically generating recourse.

[1]John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA [2]The MITRE Corporation, Bedford, MA, USA. The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the author. Approved for Public Release; Distribution Unlimited. Public Release Case Number 20-1762. Correspondence to: Michael Downs <mdowns@g.harvard.edu>, Jonathan L. Chu <jonathanchu@college.harvard.edu>, Weiwei Pan <weiweipan@g.harvard.edu>, Yaniv Yacoby <yanivyacoby@g.harvard.edu>.

2020 ICML Workshop on Human Interpretability in Machine Learning (WHI 2020).

In order for the recourse to be useful, it should be *actionable*, in that the recommended changes are realistically achievable. The suggested recourse should also be *valid*, in that the recourse truly brings an individual closer to the desired outcome. The main challenge of algorithmically generating recourse is capturing the human heuristic definition of "actionable" in formal terms. In this work, we define actionable recourse as those that obey the underlying dependencies between the data covariates in the spirit of Joshi et al. (2019) (e.g. age positively correlates with years of education), in addition to satisfying both causal constraints as in Karimi et al. (2020) (e.g. more years of education necessitates an increase in age) and specific constraint sets for each individual seeking recourse (e.g. getting more education is impossible for one individual but not another).

There is a broad class of algorithms that generate actionable recourse by first modeling the underlying structure of the data. However, these algorithms typically assume significant amounts of prior knowledge of the inner-workings of the decision making system or the underlying structure of the data. Oftentimes in practice, the only information available for recourse generation is a labeled dataset – we do not have access to the classifiers themselves nor knowledge of the underlying causal model for the data (Karimi et al., 2020). In this work, we are precisely interested in generating actionable recourse in this fully general setting.

Our contribution is two-fold: (1) we provide an analysis of failure modes for a broad class of algorithms that generate recourse that lie on the learned data manifold, and (2) based on this analysis, we provide a novel generative approach to the recourse suggestion problem that requires no more than a labeled dataset. Using a Conditional Subspace Variational Autoencoder (CSVAE) model that is capable of extracting latent features that are relevant for prediction (Klys et al., 2018), we propose a computationally efficient method, Counterfactual Recourse Using Disentangled Subspaces (CRUDS), for generating recourse that are *actionable* and *valid*. That is, we generate recourse that obey the dependencies between data covariates, as well as causal and individualized constraint sets when such knowledge is available; our recourse also reverses the original decision under the classifier in question. On synthetic and real datasets, we compare CRUDS's performance against baselines in terms of actionability and validity. We show that CRUDS reliably

produces high quality recourse without assuming access to a classifier or the complete causal model for the data.

Related Works Following the literature on counterfactual recourse, we define recourse as the difference between the observed data and an observed or synthetic data point that is likely to have the opposite label; the latter is called a *counterfactual*. The body of works addressing counterfactual generation falls largely into two categories. The first category minimizes the magnitude of suggested changes in the recourse, measured using a user-specified (often Euclidean) distance between the counterfactual and the original data point, subject to constraints that formalize user-specified notions of actionability (Ustun et al., 2019; Wachter et al., 2017; Kommiya Mothilal et al., 2019; Russell, 2019). In the second category of work, one first learns the underlying structure of the data, either implicitly as a manifold supported distribution (Joshi et al., 2019; Liu et al., 2019; Poyiadzi et al., 2020) or explicitly as a structural causal model (Karimi et al., 2020; Louizos et al., 2017), and then one generates counterfactuals that obey the learned structure. Recently, some works have combined unsupervised learning of structures in the data with partial knowledge of causal model for the data (Mahajan et al., 2019). In these works, the notion of minimal change is defined with respect to the learned structure rather than with respect to Euclidean distance in input space.

The draw-backs of the existing approaches are the following: (1) methods minimizing the magnitude of the suggested changes in recourse assume that counterfactuals close in Euclidean distance to the observed data represent actionable recourse. However, changes suggested by the minimal Euclidean distance counterfactual may not obey dependencies between the covariates in the data nor causal constraints (Karimi et al., 2020); (2) most methods rely on prior knowledge of causal structure in the data, similarity measures on points in the input space, a priori examples of real-counterfactual, and/or access to the gradients of the classifier, none of which may be available in practice; (3) furthermore, models that generate counterfactuals using the gradient of the classifier over the data manifold are sensitive to both the classification boundary and the geometry of the manifold (Section 3); (4) lastly, most methods suggest one "optimal" recourse satisfying causal or individualized constraints. However, this prevents down-stream users from ad-hoc modifying their constraints or exploring alternatives to their a priori preference.

In this work, we propose a method to generate counterfactuals that obey dependencies between data covariates while assuming nothing more than a labeled dataset. When causal and individualized constraints are available, we satisfy them without substantial additional computational costs. Our method is not sensitive to the classification boundary nor the geometry of the data manifold. Lastly, our method is able to suggest multiple counterfactuals, allowing for easy adaptation to individualized constraints or changes in user preferences.

2. Background

We assume N observations $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^{N}$, where the input x is in $\mathbb{R}^D$ and $y^{(n)}$ is a binary label.

A **Variational Autoencoder (VAE)** (Kingma & Welling, 2013) is deep generative latent variable model used for estimating the density of x in the input space. It is comprised of two models. The first is a generative model:

$$x|z \sim \mathcal{N}(f_\theta(z), \sigma_\epsilon^2 \cdot I), \quad z \sim \mathcal{N}(0, \sigma_z^2 \cdot I) \tag{1}$$

where f_θ is a neural network parameterized by θ, which transforms a simple latent $p(z)$ distribution into a data distribution $p_\theta(x)$. The second model is the inference model, $q_\phi(z|x)$, trained to approximate the posterior $p_\theta(z|x)$. The two models are trained jointly by maximize a lower bound of the evidence log-likelihood (known as the ELBO):

$$p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] = \text{ELBO}(x;\theta,\phi) \tag{2}$$

We suppose that the approximate posterior $q_\phi(z|x)$ is a mean-field Gaussian with mean and variance $\mu_\phi(x), \sigma_\phi^2(x)$ parameterized by neural networks with parameters ϕ.

The **REVISE Algorithm** (Joshi et al., 2019) is an example of a class of algorithms that first learns the underlying structure in the data by capturing $p(x)$, say, by using a VAE. In REVISE, given an observation $x^{(n)}$ and a corresponding most likely latent representation $z^{(n)}$ (i.e. $z^{(n)}$ is the mean of the variational posterior $q_\phi(z|x)$), a counterfactual is generated by performing gradient descent in the latent space, starting at $z^{(n)}$ and maximizing $p(y|x = f(z))$ as well as a user-specified similarity measure over the input space, $c(f(z), x)$. The stopping condition is when the probability of the desired flipped outcome $p(y|x = f(z))$ is larger than 0.5. The intuition behind REVISE is that by optimizing over the latent space, the counterfactual generated will lie on the data manifold. While intuitive, in Section 3, we show that algorithms like REVISE may fail to generate counterfactuals that satisfy causal constraints even if the underlying model (e.g. a VAE) is able to capture $p(x)$.

The **Conditional Subspace VAE (CSVAE)** (Klys et al., 2018) is a VAE-variant in which the latent space is partitioned into two parts – one to learn representations that are predictive of the labels, and one to learn the remaining latent representations necessary for generating the data:

$$\begin{aligned} w|y &\sim \mathcal{N}(\mu_y, \sigma_y^2 \cdot I), & y &\sim \text{Bern}(p), \\ x|w,z &\sim \mathcal{N}(f_\theta(w,z), \sigma_\epsilon^2 \cdot I), & z &\sim \mathcal{N}(0, \sigma_z^2 \cdot I), \end{aligned} \tag{3}$$

where (x, y) are labeled points, w is a latent variable that is predictive of the label y, and z is a latent variable unrelated to the label. Like in VAE inference, we introduce an inference model, $q_\phi(w, z|x, y) = q_\phi(w|x, y)q_\phi(z|x)$, and maximize a lower bound of the evidence log-likelihood:

$$p_\theta(x, y) \geq \mathbb{E}_{q_\phi(w,z|x,y)}\left[\frac{p_\theta(x, y, w, z)}{q_\phi(w, z|x, y)}\right] \tag{4}$$

The right hand side of the inequality is the ELBO, denoted $\text{ELBO}(x, y; \theta, \phi)$. Since under the generative model, w and z are independent, they should also be independent in the learned model. Klys et al. (2018) explicitly enforce this property by minimizing the mutual information between Y and Z (Appendix A).

3. Issues with Counterfactual Recourse via Optimization over Data Manifolds

In this section we describe three types of failures one may encounter when using a broad class of existing, intuitive algorithms for counterfactual generation – algorithms optimizing an objective function that includes the classifier's loss function as well as the Euclidean distance from the starting point, in either observation or latent space. We use REVISE as a specific example in our discussion. The failures we identify are surprising because, in all cases, the model (a VAE) approximates the data distribution correctly.

Counterfactuals may be out-of-distribution. While algorithms such as REVISE propose counterfactuals that lie on the data manifold and thus will obey the dependencies between the data covariates, not all counterfactuals generated in this way will lie in a high data density region. As such, REVISE may generate counterfactuals that are extremely unlikely under $p(x)$ and therefore look exceptionally different from the observed data. Consider the "Mixture-Example" in Figure 1a (details in Appendix B): REVISE stops right after the outcome is flipped, and the resultant counterfactual is unlikely under $p(x)$ due to the distance between the modes in the data distribution. We can expect this pathology to occur anytime $x|y = 0$ and $x|y = 1$ are easily separable, i.e. there is a region between the classes with low data density.

Counterfactuals are sensitive to the decision boundary. Algorithms that use the gradient of the classifier's loss function are sensitive to the decision boundary. Consider the "S-Example" in Figure 1b (details in Appendix B): REVISE prefers to cross the classification boundary nearby, resulting in an unlikely counterfactual instead of moving towards the region in which $p(x|y = 1)$ is high.

It may be impossible to flip the outcome. Algorithms whose optimization trajectories are constrained to lie on the data manifold can become "stuck" due to unfriendly geometry and topology of these manifolds Consider the "C-Example" in Figure 1c (details in Appendix B). In this example, taking gradient steps towards the boundary leads to a region where the manifold "ends" before the outcome can be flipped. There do not exist points in latent space in the direction of the optimization which correspond to points in observation space that lie on a path to the decision boundary. As such, REVISE gets stuck on the same side of the boundary. This failure may occur on many non-linear manifolds in which initial steps in the optimization must be taken away from the boundary.

In the next section, we propose a method for generating counterfactuals that obey the dependencies between the data covariates and avoids the three failure modes outlined above.

4. CRUDS: Counterfactual Recourse Using Disentangled Subspaces

The partitioned latent subspace structure of the CSVAE suggests a natural method for generating counterfactual data that target desirable outcomes. Our approach is simple.

Step 1: Disentangling latent features relevant for classification from those that are not. Given a labeled dataset $\mathcal{D}$, we first train a CSVAE to capture $p(x)$ as well as learn a latent subspace w that is predictive of the label y.

Step 2: Generating counterfactuals by changing only relevant latent features. For a data point $(x^n, y = 0)$ with corresponding approximate posteriors $q_\phi(w|x^n, y = 0)$ and $q_\phi(z|x^n)$, we take S samples from the posterior $q_\phi(z|x^n)$ over z and S samples from the prior $p(w|y = 1)$ over w:

$$z^{(s)} \sim q_\phi(z|x^n),\ w^{(s)} \sim p(w|y = 1).$$

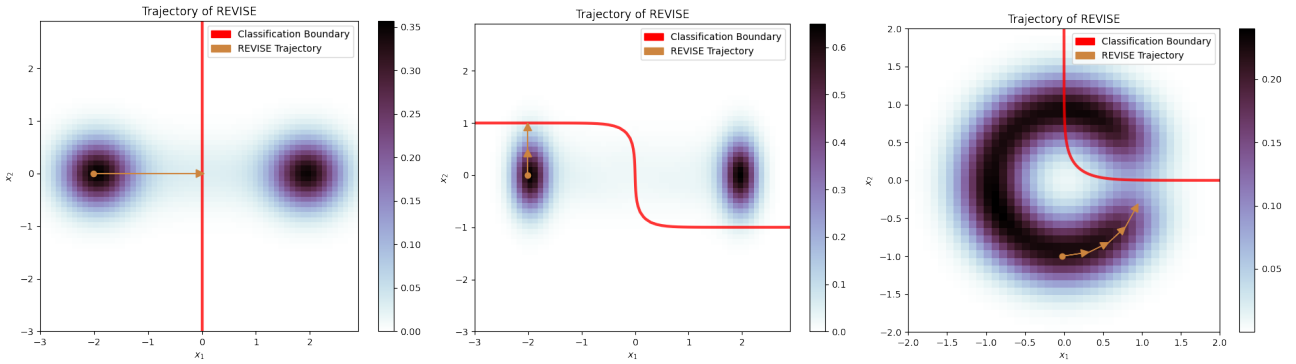
The intuition is that, by sampling z from the posterior, we preserve latent attributes of x^n that do not affect the outcome y; whereas by sampling from the prior over w, we generate a diverse set of latent attributes that encode for a positive outcome. Our set of counterfactuals is then given by

$$x_{\text{CF}}^{(s)} \sim \mathcal{N}(f_\theta(w^{(s)}, z^{(s)}), \sigma_\epsilon^2 \cdot I), s \in \{1, \ldots, S\}.$$

Note that we are approximating the distribution over counterfacturals $p(x_{\text{CF}}|x^n, y = 1)$ with samples.

Step 3: Filtering counterfactuals given constraints. When knowledge of causal constraints (e.g. more years of education necessitates an increase in age), or individual end-user preferences (e.g. getting more education is impossible for one individual but not another) are available, we filter our counterfactuals for samples that satisfy those constraints.

Step 4: Summarizing counterfactuals for interpretability. Lastly, we approximate the distribution over counterfactuals $p(x_{\text{CF}}|x^n, y = 1)$ using a decision tree, and we



(a) **Mixture-Example** (b) **S-Example** (c) **C-Example**

Figure 1. Failure cases for algorithms that perform gradient descent on the data manifold. (a) Algorithm produced counterfactual unlikely under the data distribution. (b) Algorithm is sensitive to the classification boundary. (c) Algorithm could not flip outcome.

summarize paths in the decision tree model as decision rules. We condense the clauses in the decision rule sets and sort the rules by the likelihood of the counterfactuals generated by each rule. This way, we obtain an interpretable discretization of the input space that is ranked by the likelihood of each region under $p(x_{\text{CF}}|x^n, y = 1)$. See Appendix D for a complete description.

By design, CRUDS offers four important improvements to existing approaches:

A. CRUDS does not require significant prior knowledge in order to preserve data covariate relationships in the generated counterfactuals. To generate counterfactuals that obey the relationship between the covariates in the data, CRUDS only requires a labeled dataset. Unlike (Joshi et al., 2019), we do not assume access to ground truth classifiers or user-defined similarity measures; unlike (Mahajan et al., 2019), we do not assume a priori examples of counterfactuals, or external oracles.

B. CRUDS is not sensitive to decision boundaries or geometry of data-manifolds. Instead of optimizing a synthetic data-point to have the desired outcome, CRUDS allows us to directly sample synthetic data-points with the desired outcomes. A such, it is insensitive to the irregularities in the classifier's decision boundary and difficult geometries of the data manifold that typically trouble optimization-based algorithms. Figures 6, 7 and 8 in Appendix C demonstrate that a CSVAE is able to model $p(x_{\text{CF}}|x, y = 1)$ on the three pathological examples of Section 3, while algorithms such as REVISE propose unrealistic or invalid counterfactuals.

C. CRUDS can satisfy additional causal or individualized constraints on counterfactuals with ease. When additional knowledge of causal constraints or user preferences are available, CRUDS can suggest counterfactuals that satisfy these constraints without significant additional computational cost, by filtering for satisfactory counterfactuals in the generated set. This flexibility allows for end-users to perform ad-hoc exploration.

D. CRUDS produces in-distribution counterfactuals. As we noted in Section 3, algorithms that prioritize minimizing distance (in observed or latent space) to the factual data may propose counterfactuals that are located in low data density regions. By sampling and filtering counterfactuals, CRUDS only generates counterfactuals that are within the observed data distribution.

5. Experiments

Data Sets We evaluate CRUDS on seven synthetically generated (described in Appendix B) and three real datasets, the Credit Default dataset, the Wine Quality dataset, and the South German Credit dataset (Dua & Graff, 2017). On the "Mixture", "C", "S", and "Overlap" Examples, we verify that CRUDS does not suffer the failure modes we observed in Section 3. On data generated from a simple causal model, we check that CRUDS counterfactuals preserve the true dependencies between the covariates in the data. On "Hidden Confounding", we show that when the data generating process does not match that assumed by the CSVAE (a hidden confounder is present), CRUDS counterfactuals can fail to preserve the relationship between the covariates. On "Nonlinear", we show that CRUDS is agnostic to the shape of the decision boundary.

Baselines We compare the performance of CRUDS against REVISE as well as against two gradient-based approaches using a CSVAE (rather than a VAE). We refer to the latter as Path-CRUDS (p-CRUDS) (details in Appendix E).

Evaluation We evaluate the *validity* by computing the fraction of counterfactuals that actually flip the classifier's decision, and by computing the average classification probability

Figure 2. Recourse options produced by CRUDS on the South German Credit Example. Each row represents an option, consisting of several clauses that must be satisfied to get the desirable outcome. Green/red means that the user does / does not satisfy the clause, respectively.

on the generated counterfactuals. When the true structure of the data is known and in cases where we have knowledge of causal constraints, we evaluate how well counterfactuals obey these relationships, by computing the constraint feasibility score from Mahajan et al. (2019). We also inspect some of the counterfactuals as a qualitative check. Finally, we measure the "difficulty" of the recommended recourse through several distance-based and percentile-shift-based cost metrics (details in Appendix F.1).

6. Results and Discussion

CRUDS generates valid counterfactuals. CRUDS counterfactuals are 100% valid on all real and synthetic datasets, except for "Hidden Confounding". Importantly, CRUDS is able to produce valid counterfactuals for any decision boundary, while REVISE becomes inefficient (w.r.t number of gradient steps) with a nonlinear decision boundary (as in the "Nonlinear" Example, Figure 10, and as in the "Overlap" Example, Figure 5), and fails to generate valid counterfactuals in "C" Example (Figure 8).

Recourse proposed by CRUDS is more actionable than that of baselines. While our CRUDS-based methods tend to perform slightly worse overall on the distance based metrics, they perform better on the causal constraint feasibility score. For example, on the Credit Default data-set, CRUDS satisfies all causal constraints (listed in Appendix F.1) 100% of the time, while REVISE satisfies the constraints only 69% of the time. We note that this percentage is computed as an average across a set of six causal constraints – the number of counterfactuals that satisfy *all* constrains simultaneously is substantially lower (refer to Table 3a for details). This again reflects that distance based metrics alone do not capture actionability. The CRUDS-based methods also propose recourse which are qualitatively reasonable.

Recourse proposed by CRUDS is more interpretable than that of baselines. Figure 2 details the recourse options for the South German Credit dataset (Dua & Graff, 2017), produced by the interpretable summary of $p(x_{\text{CF}}|x, y = 1)$ in step 4 of CRUDS (Section 4): each row represents an option, consisting of several clauses that must be satisfied for to get the desirable outcome. Overall, CRUDS recommends increasing available funds within a checking account, but gives several options for how to do this – one of which may work for the user. For example, option 1 and option 8 represent a choice between increasing savings or obtaining a better credit history while option 3 suggests that it is sufficient to acquire a better credit history and increase savings while ignoring the checking account funds. We therefore see that CRUDS's recommendations are aligned with human judgement, whereas other methods often give the end-user only a single (and potentially non-actionable) recommendation. For example, for an applicant deemed as high risk due to insufficient checking and savings account funds, problems with credit history, a long loan duration, and large amount of money requested, REVISE recommend decreasing age (not possible), as well as increasing the installment payments and the loan amount, the latter of which increases loan risk. More details in Appendix G.

Distance/shift minimization is a poor heuristic for actionability. As we discuss above, although REVISE tends to score better on distance and percentile shift metrics, CRUDS recourse better satisfy dependencies between the covariates in the data and causal constraints. Recourse that propose smaller amounts of change are not necessarily the most actionable or valid.

CRUDS cannot recover covariate relationships under model mismatch. "Hidden Confounding" illustrates the challenge of learning data dependencies under hidden confounding variables for all approaches (Table 1b); for CRUDS in particular, target class validity is relatively low, the distance-based metrics are higher, and the constraint feasibility is low. Qualitatively, CRUDS proposes recourse that are nonsensical (Table 2b).

7. Conclusion

We show failure cases for a broad class of counterfactual generation algorithms that optimize a classifier-based objective over the data manifold. We propose a novel method for generating recourse, CRUDS, and demonstrate through experiments that CRUDS consistently produces valid recourse that respects dependencies in the data, as well as causal and individualized constraints when available. In future work, we aim to extend CRUDS to the semi-supervised case in which labels are not available for all data points as well as scenarios where the feature-set is of mixed type. We also aim to formally evaluate the decision-rule summaries of counterfactuals for interpretability with a user-study.

Acknowledgments

MD and WP are supported by the Harvard Institute of Applied Computational Sciences. YY acknowledges support from NIH 5T32LM012411-04 and from the IBM Faculty Research Award.

References

Dressel, J. and Farid, H. The accuracy, fairness, and limits of predicting recidivism. Science advances, 4(1):eaao5580, 2018.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Joshi, S., Koyejo, O., Vijitbenjaronk, W., Kim, B., and Ghosh, J. Towards realistic individual recourse and actionable explanations in black-box decision making systems. CoRR, abs/1907.09615, 2019. URL http://arxiv.org/abs/1907.09615.

Karimi, A.-H., Schölkopf, B., and Valera, I. Algorithmic recourse: from counterfactual explanations to interventions, 2020.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. arXiv e-prints, art. arXiv:1412.6980, December 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes, 2013.

Klys, J., Snell, J., and Zemel, R. S. Learning latent subspaces in variational autoencoders. CoRR, abs/1812.06190, 2018. URL http://arxiv.org/abs/1812.06190.

Kommiya Mothilal, R., Sharma, A., and Tan, C. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. arXiv e-prints, art. arXiv:1905.07697, May 2019.

Liu, S., Kailkhura, B., Loveland, D., and Han, Y. Generative Counterfactual Introspection for Explainable Deep Learning. arXiv e-prints, art. arXiv:1907.03077, July 2019.

Louizos, C., Shalit, U., Mooij, J. M., Sontag, D., Zemel, R., and Welling, M. Causal effect inference with deep latent-variable models. In Advances in Neural Information Processing Systems, pp. 6446–6456, 2017.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. From local explanations to global understanding with explainable ai for trees. Nature Machine Intelligence, 2(1):2522–5839, 2020.

Mahajan, D., Tan, C., and Sharma, A. Preserving causal constraints in counterfactual explanations for machine learning classifiers. ArXiv, abs/1912.03277, 2019.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., and Flach, P. Face: feasible and actionable counterfactual explanations. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, pp. 344–350, 2020.

Russell, C. Efficient search for diverse coherent explanations. In Proceedings of the Conference on Fairness, Accountability, and Transparency, pp. 20–28, 2019.

Ustun, B., Spangher, A., and Liu, Y. Actionable recourse in linear classification. Proceedings of the Conference on Fairness, Accountability, and Transparency - FAT* '19, 2019. doi: 10.1145/3287560.3287566. URL http://dx.doi.org/10.1145/3287560.3287566.

Wachter, S., Mittelstadt, B., and Russell, C. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. arXiv e-prints, art. arXiv:1711.00399, November 2017.

A. Conditional Subspace VAE Inference

The **Conditional Subspace VAE (CSVAE)** (Klys et al., 2018) is a VAE-variant in which the latent space is partitioned into two parts – one to learn representations that are predictive of the labels, and one to learn the remaining latent representations necessary for generating the data:

$$\begin{aligned} w|y &\sim \mathcal{N}(\mu_y, \sigma_y^2 \cdot I), & y &\sim \text{Bern}(p), \\ x|w,z &\sim \mathcal{N}(f_\theta(w,z), \sigma_\epsilon^2 \cdot I), & z &\sim \mathcal{N}(0, \sigma_z^2 \cdot I), \end{aligned} \tag{5}$$

where (x, y) are labeled points, w is a latent variable that is predictive of the label y, and z is a latent variable unrelated to the label. Like in VAE inference, we introduce an inference model, $q_\phi(w, z|x, y) = q_\phi(w|x, y)q_\phi(z|x)$, and maximize a lower bound of the evidence log-likelihood:

$$p_\theta(x, y) \geq \mathbb{E}_{q_\phi(w,z|x,y)} \left[\frac{p_\theta(x, y, w, z)}{q_\phi(w, z|x, y)} \right] \tag{6}$$

The right hand side of the inequality is the ELBO, denoted $\text{ELBO}(x, y; \theta, \phi)$.

We follow the inference proposed by Klys et al. (2018). We maximize the CSVAE ELBO, averaged over the data:

$$\mathcal{M}_1 = \frac{1}{N} \sum_{n=1}^{N} \text{ELBO}(x_n, y_n, \theta, \phi) \tag{7}$$

Since in practice, maximizing $\mathcal{M}_1$ does not guarantee that Z is independent of Y in the learned model (as specified by the generative model), we maximize $\mathcal{M}_1$ while minimizing the information shared by Y and Z:

$$\mathcal{M}_2 = I(Y; Z) \tag{8}$$

$$= \mathbb{H}(Y) - \mathbb{H}(Y|Z) \tag{9}$$

where

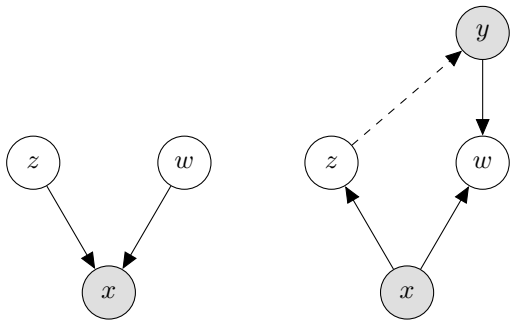
$$\mathbb{H}(Y|Z) = \int_{z,y,x} p_\theta(z|x)p(x)p_\theta(y|z) \log p_\theta(y|z) dx dy dz. \tag{10}$$

Since the integral above is intractable, we substitute in approximate posteriors, $p_\theta(z|x) \approx q_\phi(z|x)$, and $p_\theta(y|z) \approx q_\delta(y|z)$:

$$\mathbb{H}(Y|Z) \approx \mathbb{H}(Y) - \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{q_\phi(z|x_n)q_\delta(y_n|z)} \left[\log q_\delta(y_n|z) \right] \tag{11}$$

wherein $q_\delta(y|z)$ is trained to approximate $p_\theta(y|z)$ by maximizing

$$\mathcal{M}_3 = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{q_\phi(z|x_n)} \left[\log q_\delta(y_n|z) \right] \tag{12}$$



(a) Generative Model (b) Inference Model

Figure 3. CSVAE Model (Klys et al., 2018)

Putting it all together, the training objective is:

$$\min_{\theta,\phi} -\beta_1 \mathcal{M}_1 + \beta_2 \mathcal{M}_2, \quad \max_{\delta} \beta_3 \mathcal{M}_3 \tag{13}$$

where the β_i's represent the relative importance of the terms.

B. Pedagogical Examples

Mixture Example Consider data generated from the following VAE model (visualized in Figure 6a):

$$\begin{aligned} z &\sim \mathcal{N}(0, 1) \\ u(z) &= 20.0 \cdot (\Phi_{p(z)}(z) - 0.5) \\ t(u) &= \tanh(u + 0.5) + \tanh(u - 0.5) + 0.001 \cdot u \\ f(z) &= \begin{bmatrix} t\,(u(z)) \\ 0.0 \end{bmatrix} \\ x|z &= \mathcal{N}(f(z), \sigma_\epsilon^2) \end{aligned} \tag{14}$$

where $\sigma_\epsilon^2 = 0.2$ and $\Phi_{p(z)}(\cdot)$ is the CDF of $p(z)$. Equivalently, consider the same data generated from a CSVAE model (visualized in Figure 6c):

$$\begin{aligned} y &\sim \text{Bern}\left(\frac{1}{2}\right) \\ w|y &\sim \mathcal{N}(-1 + 2 \cdot y, \sigma_w^2) \\ z &\sim \mathcal{N}(0, 1) \\ u(w) &= 20.0 \cdot (\Phi_{p(w)}(w) - 0.5) \\ t(u) &= \tanh(u + 0.5) + \tanh(u - 0.5) + 0.001 \cdot u \\ f(w, z) &= \begin{bmatrix} t\,(u(w)) \\ 0.0 \end{bmatrix} \\ x|w, z &= \mathcal{N}(f(w, z), \sigma_\epsilon^2) \end{aligned} \tag{15}$$

where $\sigma_\epsilon^2 = 0.2, \sigma_w^2 = 0.01$ and $\Phi_{p(w)}(\cdot)$ is the CDF of $p(w)$. For these examples, the true classifier $y|x$ is given by:

$$p(y|x_1, x_2) = \frac{1}{1 + \exp(-5.0 \cdot x_1)} \tag{16}$$

which for a threshold of 0.5 gives a classification boundary at $x_1 = 0.0$. Note that in this example, z, the latent code not correlated with the label y, is ignored for simplicity.

S-Example Consider data generated from the following VAE model (visualized in Figure 7a):

$$\begin{aligned} z_1, z_2 &\sim \mathcal{N}(0, 1) \\ u(z) &= 20.0 \cdot (\Phi_{p(z)}(z) - 0.5) \\ t(u) &= \tanh(u + 0.5) + \tanh(u - 0.5) + 0.001 \cdot u \\ f(z_1, z_2) &= \begin{bmatrix} t\left(u(z_1)\right) \\ 0.4 \cdot z_2 \end{bmatrix} \\ x|z_1, z_2 &= \mathcal{N}(f(z_1, z_2), \sigma_\epsilon^2) \end{aligned} \tag{17}$$

where $\sigma_\epsilon^2 = 0.05$ and $\Phi_{p(z)}(\cdot)$ is the CDF of $p(z)$. Equivalently, consider the same data generated from a CSVAE model (visualized in Figure 7c):

$$\begin{aligned} y &\sim \text{Bern}\left(\frac{1}{2}\right) \\ w|y &\sim \mathcal{N}(-1 + 2 \cdot y, \sigma_w^2) \\ z &\sim \mathcal{N}(0, 1) \\ u(w) &= 20.0 \cdot (\Phi_{p(w)}(w) - 0.5) \\ t(u) &= \tanh(u + 0.5) + \tanh(u - 0.5) + 0.001 \cdot u \\ f(w, z) &= \begin{bmatrix} t\left(u(w)\right) \\ 0.4 \cdot z \end{bmatrix} \\ x|w, z &= \mathcal{N}(f(w, z), \sigma_\epsilon^2) \end{aligned} \tag{18}$$

where $\sigma_\epsilon^2 = 0.05, \sigma_w^2 = 0.01$ and $\Phi_{p(w)}(\cdot)$ is the CDF of $p(w)$. For these examples, the true classifier $y|x$ is given by:

$$\begin{aligned} p(y|x_1, x_2) =& \sigma(-5.0 \cdot x_1) \cdot \sigma(5.0 * (x_2 - 1.0)) \\ &+ \sigma(5.0 \cdot x_1) \cdot \sigma(5.0 * (x_2 + 1.0)) \end{aligned} \tag{19}$$

where σ represents the sigmoid function.

C-Example Consider data generated from the following VAE model (visualized in Figure 8a):

$$\begin{aligned} z &\sim \mathcal{N}(0, 1) \\ u(z) &= \pi \cdot (1.9 - 1.8 \cdot \Phi_{p(z)}(z)) \\ f(z) &= \begin{bmatrix} \cos\left(u(z)\right) \\ \sin\left(u(z)\right) \end{bmatrix} \\ x|z &= \mathcal{N}(f(z), \sigma_\epsilon^2) \end{aligned} \tag{20}$$

where $\sigma_\epsilon^2 = 0.1$ and $\Phi_{p(z)}(\cdot)$ is the CDF of $p(z)$. Equivalently, consider the same data generated from a CSVAE model (visualized in Figure 8c):

$$\begin{aligned} y &\sim \text{Bern}\left(\frac{2}{9}\right) \\ w|y &\sim \mathcal{N}(-1 + 2 \cdot y, \sigma_w^2) \\ z &\sim \mathcal{N}(0, 1) \\ u(w) &= \pi \cdot (1.9 - 1.8 \cdot \Phi_{p(w)}(w)) \\ f(w, z) &= \begin{bmatrix} \cos\left(u(w)\right) \\ \sin\left(u(w)\right) \end{bmatrix} \\ x|w, z &= \mathcal{N}(f(w, z), \sigma_\epsilon^2) \end{aligned} \tag{21}$$

where $\sigma_\epsilon^2 = 0.1, \sigma_w^2 = 0.01$ and $\Phi_{p(w)}(\cdot)$ is the CDF of $p(w)$. For these examples, the true classifier $y|x$ is given by:

$$p(y|x_1, x_2) = \frac{1}{1 + \exp(-5.0 \cdot x_1)} \cdot \frac{1}{1 + \exp(-5.0 \cdot x_2)} \tag{22}$$

which for a threshold of 0.5 gives a classification boundary at,

$$x_2 = -0.2 \cdot \log\left(\frac{2.0}{1.0 + \exp(-5.0 \cdot x_1)} - 1.0\right) \tag{23}$$

Note that in this example, z, the latent code not correlated with the label y, is ignored for simplicity.

Simple Causal Example We adopt a causal data generative process from Karimi et al. (2020) (Figure 4a):

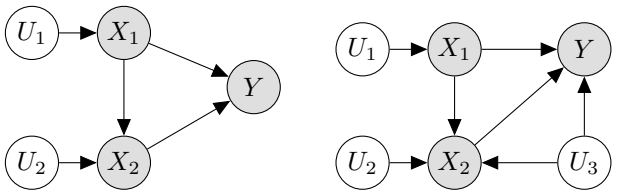
$$\begin{aligned} U_1 &\sim \text{Pois}(100000) \\ U_2 &\sim \mathcal{N}(0, 2500^2) \\ X_1 &= U_1 \\ X_2 &= 0.3 * X_1 + U_2 \\ Y &= \text{Bern}(\sigma(X_1 + 5 * X_2 - 225000)) \end{aligned} \tag{24}$$

where σ is the sigmoid function. We standardize the features after generating them.

We consider the simple real world manifestation of this causal structure as the process of making loan decisions, where X_1 represents income, X_2 represents savings, and Y represents the loan decision ($Y = 1$ being approval).

Hidden Confounding Example We extend the simple causal example (24) with the inclusion of an additional unobserved confounding variable (Figure 4b):

$$\begin{aligned} U_1 &\sim \text{Pois}(100000) \\ U_2 &\sim \mathcal{N}(0, 2500^2) \\ U_3 &\sim \text{Clip}(\text{Pois}(600), 300, 850) \\ X_1 &= U_1 \\ X_2 &= 0.3 * X_1 + U_2 + (-50000) * \text{N}(U_3) \\ Y &= \text{Bern}(\sigma(0.4 * \text{S}(X_1 + 3 * X_2) + 0.6 * \text{S}(U_3))) \end{aligned} \tag{25}$$



(a) Simple causal (b) Hidden confounding

Figure 4. Causal generative graphs. Shaded and unshaded nodes represent observed and unobserved variables, respectively.

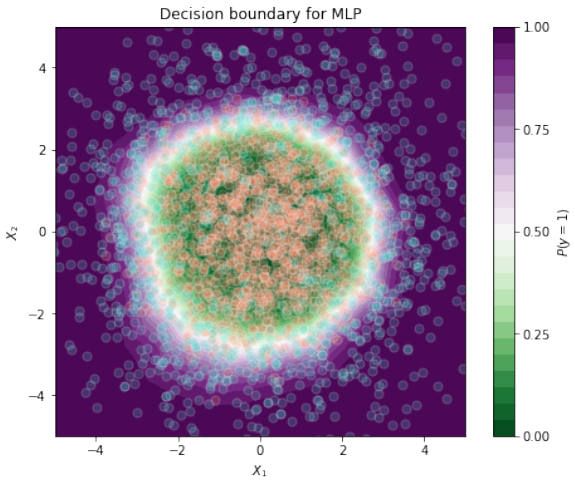


Figure 5. Nonlinear decision boundary example

where $\mathrm{N}(X)$ and $\mathrm{S}(X)$ refer to normalized and standardized X, respectively, and $\mathrm{Clip}(X, min, max)$ clips the random variable X such that values less than min are set to min and likewise with max. We standardize the features after generating them.

Extending the loan decision scenario, the hidden confounder U_3 represents an unobserved credit score. We assume that a higher credit score increases the chance of the loan being approved, and decreases the amount of savings available because the higher credit score represents a more robust credit history due to increased spending. The true effect of savings on the outcome of the loan application is positive, but looks negative due to the credit score confounder.

Nonlinear Example We generate data using a spherical gaussian in $\mathbb{R}^2$ and a circular decision boundary (Figure 5):

$$\begin{aligned} x_1, x_2 &\sim \mathcal{N}(0, 2^2) \\ y &= \mathrm{Bern}(\sigma(.5 * (x_1^2 + x_2^2 - 8))) \end{aligned} \tag{26}$$

Overlap Example Consider data generated from the following generative process, in which $x|y = 0$ and $x|y = 1$ overlap significantly:

$$\begin{aligned} y &\sim \mathrm{Bern}\left(\frac{1}{2}\right) \\ w &\sim \mathcal{N}(-1 + 2 \cdot y, 0.25^2) \\ z &\sim \mathcal{N}(0, 1) \\ x|w, z &\sim \mathcal{N}(f(w, z), 0.1^2) \end{aligned} \tag{27}$$

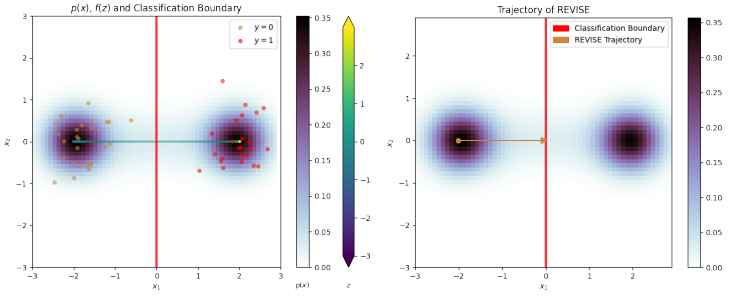
where

$$\begin{aligned} a(w) &= \mathbb{I}(w \geq 0) \cdot \left(0.5 \cdot \Phi(w; 1.0, 0.25^2) + 0.5\right) \\ &\quad + \mathbb{I}(w < 0) \cdot \left(0.5 \cdot \Phi(w; -1.0, 0.25^2)\right) \\ b(w) &= B^{-1}(a(w); 0.2, 0.2) \\ u_0(z) &= \pi \cdot \sqrt{\Phi(z; 0, 1)} \\ u_1(z) &= \pi \cdot (\Phi(z; 0, 1))^2 \\ f(w, z) &= \begin{bmatrix} \cos\left(b(w) * u_0(z) + (1.0 - b(w)) * u_1(z)\right) \\ \sin\left(b(w) * u_0(z) + (1.0 - b(w)) * u_1(z)\right) \end{bmatrix} \end{aligned} \tag{28}$$

in which $\Phi(\cdot; \mu, \sigma^2)$ is the CDF of $\mathcal{N}(\cdot; \mu, \sigma^2)$, and $B^{-1}(\cdot; \alpha, \beta)$ is the inverse-CDF of the Beta distribution with parameters α, β.

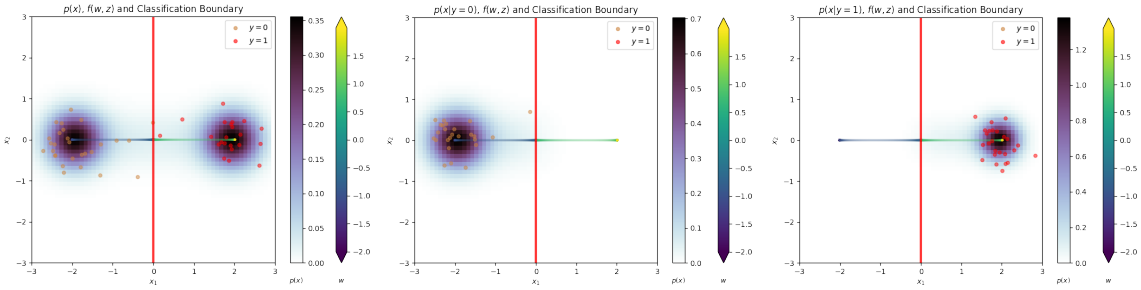
C. A Case for Sampling-Based Recourse with CSVAEs

As Section 3 demonstrates, obtaining recourse by performing gradient-descent in latent space is subject to several problems: (1) the proposed counterfactual may be infeasible (or unlikely), (2) the algorithm may fail to propose a counterfactual, and (3) only one counterfactual is proposed, which may not correspond to successful recourse. As we show here, sampling-based recourse mitigates all of the above issues. In sampling-based counterfactual generation, we always sample from high regions of the space and are therefore not fooled by properties of the classification boundary. Figures 6d, 7d and 8d all show that the two data conditionals $p(x|y)$ of the CSVAE sample points in the high-mass region on the correct side of the boundary. Furthermore, due to sampling, we generate a diverse set of valid counterfactuals, some of which may correspond to actionable recourse.



(a) Classifier boundary, and $p(x)$, $f(z)$ of VAE

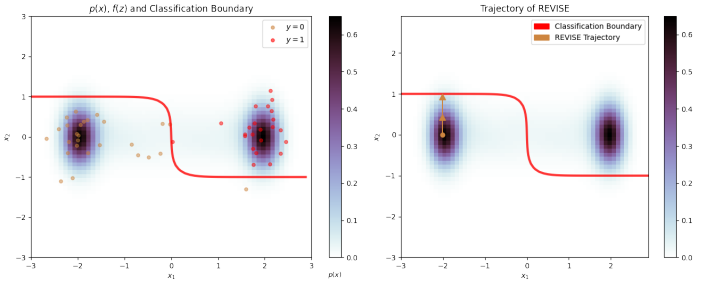
(b) REVISE generates unlikely counterfactual



(c) Classifier boundary, $p(x)$, $f(w, z)$ of CSVAE

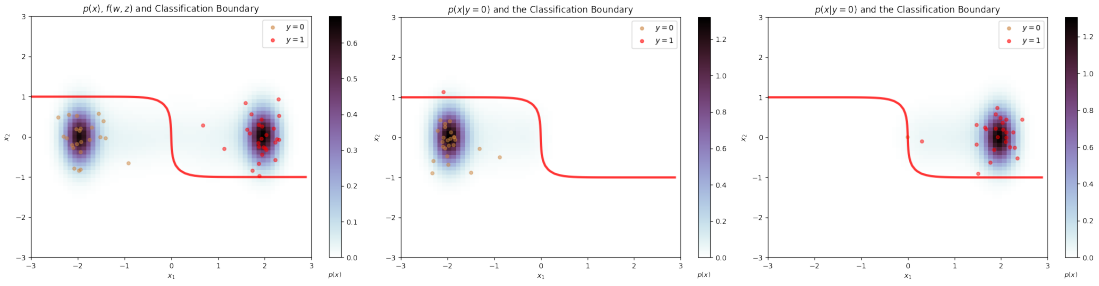
(d) CSVAE is able to partition the space into two regions: $p(x|y = 0)$ on the left and $p(x|y = 1)$ on the right.

Figure 6. **Mixture-Example.** Although both VAE and CSVAE model $p(x)$ perfectly, REVISE+VAE generates a counterfactual that is extremely unlikely under the data distribution. CSVAE+Sampling, on the other hand, simply by conditioning on $y = 1$, guarantees that generated counterfactuals are the correct side of the boundary and are likely.



(a) Classifier boundary and $p(x)$ of VAE

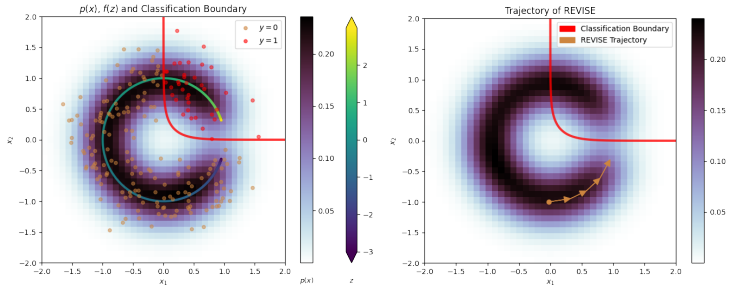
(b) REVISE is sensitive to the classification boundary



(c) Classifier boundary and $p(x)$ of CSVAE

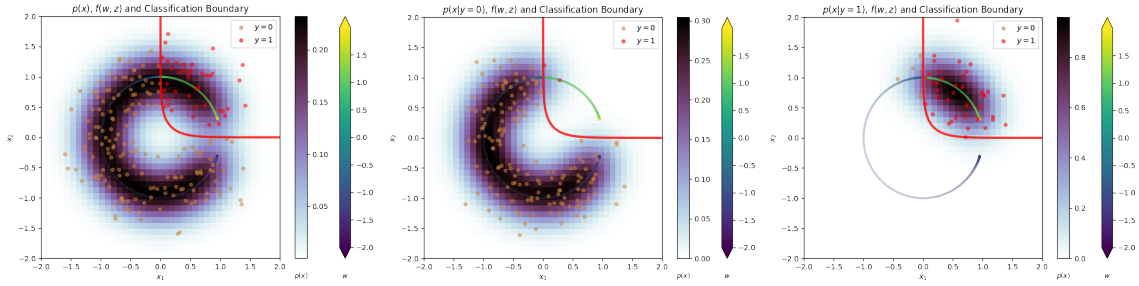
(d) CSVAE is able to partition the space into two regions: $p(x|y = 0)$ on the left and $p(x|y = 1)$ on the right.

Figure 7. **S-Example.** Although the VAE and CSVAE both model $p(x)$ perfectly, because the classification boundary passes in a low-mass region of $p(x)$ near the high-mass region of $p(x)$, REVISE+VAE flips the outcome trivially instead of moving towards where $p(x|y = 1$ is high. CSVAE+Sampling, on the other hand, simply by conditioning on $y = 1$, guarantees that generated counterfactuals are on the correct side of the boundary.



(a) Classifier boundary, and $p(x)$, $f(z)$ of VAE

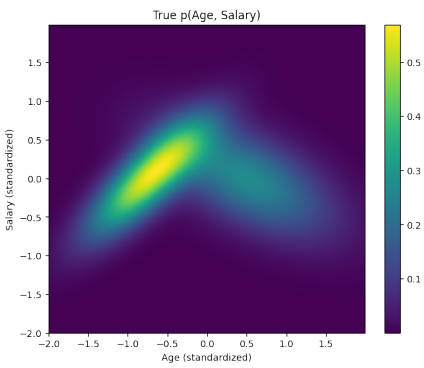
(b) REVISE cannot flip label



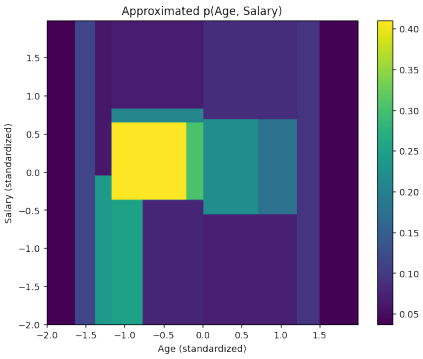
(c) Classifier boundary, $p(x)$, $f(w, z)$ of CSVAE

and (d) CSVAE is able to partition the space into two regions: $p(x|y = 0)$ on the left and $p(x|y = 1)$ on the right.

Figure 8. **C-Example.** Although the VAE and CSVAE both model $p(x)$ perfectly, due to the curvature of the manifold, REVISE+VAE is unable to flip the outcome. CSVAE+Sampling, on the other hand, simply by conditioning on $y = 1$, guarantees that generated counterfactuals are the correct side of the boundary.



(a) Counterfactual distribution $p(x_{\text{CF}}|x, y')$



(b) Approximation of $p(x_{\text{CF}}|x, y')$

Option 1:	Salary ≤ 0.653	Salary > -0.362	Age ≤ -0.219	Age > -1.179
Option 2:	Salary ≤ 0.653	Salary > -0.362	Age ≤ 0.001	Age > -0.219
Option 3:	Salary ≤ -0.362	Age ≤ -0.772	Age > -1.179	
Option 4:	Age ≤ -1.179	Age > -1.380	Salary ≤ -0.043	
Option 5:	Salary ≤ 0.700	Salary > -0.556	Age ≤ 0.704	Age > 0.001
Option 6:	Age ≤ 0.001	Age > -1.179	Salary ≤ 0.839	Salary > 0.653
Option 7:	Salary ≤ 0.700	Salary > -0.556	Age ≤ 1.204	Age > 0.704
Option 8:	Age ≤ -1.380	Age > -1.644		
Option 9:	Age ≤ 1.495	Age > 1.204		
Option 10:	Age ≤ 1.204	Age > 0.001	Salary > 0.700	
Option 11:	Salary ≤ -0.362	Age ≤ 0.001	Age > -0.772	
Option 12:	Age ≤ 1.204	Age > 0.001	Salary ≤ -0.556	
Option 13:	Age ≤ 0.001	Age > -1.179	Salary > 0.839	
Option 14:	Age ≤ -1.179	Age > -1.380	Salary > -0.043	
Option 15:	Age > 1.495			
Option 16:	Age ≤ -1.644			

(c) Rules presented to user. Green means already satisfied by $x = [0.5, 0.5]$, red means is not yet satisfied.

Figure 9. **Example of Counterfactual Distribution Summary.** A target (synthetic) counterfactual distribution $p(x_{\text{CF}}|x, y')$ (left) is discretized using a decision tree (middle) and converted into a set of decision rules (right), sorted in order of most likely to least likely.

D. Summarizing the Counterfactual Distribution $p(x_{\text{CF}}|x, y')$

We need a human-understandable way of communicating to a user x a distribution over counterfactuals x_{CF}, for which the outcome is flipped from y to y'. The problem with having a distribution over x_{CF} (as opposed to a single x_{CF} obtained via optimization), is that a distribution is hard to summarize, especially in a high-dimensional and possibly multi-modal setting. As an example, consider the case in which we draw two samples $x_{\text{CF}}^{(1)}$ and $x_{\text{CF}}^{(2)}$ that are nearly identical. A user looking at these two counterfactuals really only needed to have seen one of them. So how do we summarize the distribution concisely? We do this using the following framework:

1. **Partition:** We partition the space of counterfactuals based on the value of $p(x_{\text{CF}}|x, y')$: ex. region 1 is when $p(x_{\text{CF}}|x, y')$ is high, region 2 is when $p(x_{\text{CF}}|x, y')$ is medium, and region 3 is when $p(x_{\text{CF}}|x, y')$ is low.
2. **Summarize:** For each region, we summarize what kinds of x_{CF} define that region: ex. when $x_{\text{CF}} < 0.5$ or $x_{\text{CF}} > 1.0$, we are in region 1.
3. **Sort and Condense:** We sort the summaries from high-to-low $p(x_{\text{CF}}|x, y')$ – this way, the user will first look at a summary of the region mostly likely to be applicable to them.

In this work, we chose to implement these three steps as follows:

1. **Partition:** We sample S samples from $p(x_{\text{CF}}|x, y')$ and evaluate their probabilities under $p(x_{\text{CF}}|x, y')$. We then discretize the input space as well as the value of the distribution by training a decision tree regressor to map samples to their respective probabilities: $\text{DecisionTree} : x_{\text{CF}} \to p(x_{\text{CF}}|x, y')$. Figure 9a and 9b show an example $p(x_{\text{CF}}|x, y')$ we wish to summarize and its approximation with the decision tree, respectively.
2. **Summarize:** Each path from the root to a leaf correspond to a set of conditions that all must be satisfied for sample x_{CF} to have the probability listed in the leaf node. Each path therefore represents an AND of clauses that, if the user satisfies, will be granted recourse.
3. **Sort and Condense:** We sort the paths by their probabilities (most likely to least likely). We then condense redundant information (ex. if a path contains two conditions, $x_1 < 0.0$ AND $x_1 < 1.0$, we can instead just write $x_1 < 0.0$). Each option presented in the summary now has at most two clauses per feature of x_{CF}. Figure 9c shows the end result of this algorithm.

E. CRUDS-based Baselines

We propose two variants that use a REVISE-like algorithm on a CSVAE as baselines:

p-CRUDS 1 performs gradient ascent steps in W space, maximizing $p(w|y = 1)$. It does so without making use of an external classifier and halts when $p(w|y = 1) > p(w|y = 0)$.

p-CRUDS 2 also performs gradient ascent steps in W space, maximizing $p(w|y = 1)$. It does make use of an external classifier and halts when $\text{classifier}(y = 1|x_{\text{CF}}) > 0.5$.

F. Quantitative Results

For each dataset, we summarize the results of our experiments in two tables: one comparing the evaluation metrics across the REVISE baseline and CRUDS-based methods, and a second presenting a sample data point along with proposed recourse generated using different methods (Tables 1, 2, 3 and 4). In every case, the original features of the sample data point result in an undesirable classification, while the proposed recourse options all result in a desirable classification. The sample recourse is presented for intuitive, qualitative evaluation of the actionability and realism of our proposed methods. A dash indicates that a metric does not apply or was not computed due to lack of domain expertise needed for reasonable evaluation.

F.1. Evaluation Metrics

Let X_F be a factual observation and X_{CF} be its corresponding counterfactual obtained from an algorithm that produces recourse through counterfactuals. Such algorithms as REVISE and p-CRUDS also produce a set of intermediate values that interpolate between X_F and X_{CF}, which we refer to as $\text{path}(X_F, X_{CF})$. Let $\text{clf}(x) = \mathbb{I}[\text{clf\_prob}(x) \geq .5]$ be a classifier. The metrics computed in the experiments are the average of the following functions over all observations in need of recourse under clf on a held out test set.

1. Target Class Validity: $c(X_{CF}) = \text{clf}(X_{CF})$, where $\text{clf}(x) = 1$ is always assumed to be the desirable outcome. Whether or not the recommended counterfactual actually results in the better outcome under clf, a higher target class validity is better.
2. Target Class Probability: $c(X_{CF}) = \text{clf\_prob}(X_{CF})$. This metric provides a sense for how close to the decision boundary the counterfactuals lie on average. Whether closer or farther away is better depends on the application.
3. ℓ_2 distance between the original and counterfactual: $c(X_F, X_{CF}) = \|X_{CF} - X_F\|_2$. This metric provides

us with a naive measure of similarity between factual and counterfactual datapoints. Large values imply that the counterfactual is more different than the original point, but as we show in this paper this is not necessarily worse.

4. Normalized ℓ_1: $c(X_F, X_{CF}) = \sum_i \frac{|X^i_{CF} - X^i_F|}{\text{MAX}_i - \text{MIN}_i}$ where MAX_i and MIN_i are the max and min values for the ith feature determined using the train set (from Karimi et al. (2020)). This metric provides another measure of similarity that weights each feature equally. Large values imply that the counterfactual is more different than the original point, but as we show in this paper this is not necessarily worse.

5. Approximate geodesic length: the geometric notion of shortest distance from the factual example to the counterfactual while traversing the data manifold. This is the sum of the ℓ_2 distance between subsequent points in path(X_F, X_{CF}). The path on the data manifold from the factual to the counterfactual is not necessarily a straight line so this cost metric gives a more realistic measure of distance, since it may not be possible in real life to "move in a straight line" to the counterfactual. Large values imply that the counterfactual is more different than the original point, but as we show in this paper this is not necessarily worse.

6. Total Percentile Shift: $c(X_F, X_{CF}) = \sum_i |\text{ECDF}_i(X^i_{CF}) - \text{ECDF}_i(X^i_F)|$ where ECDF_i is the empirical CDF for the ith feature. Similar to the normalized ℓ_1 metric, the total percentile shift gives us a sense for how different the counterfactual is from the original point relative to the rest of the population. Large values imply that the counterfactual is more different than the original point, but as we show in this paper this is not necessarily worse.

7. Total Log Percentile Shift: $c(X_F, X_{CF}) = \sum_i \left| \frac{\log(1-\text{ECDF}_i(X^i_{CF}))}{\log(1-\text{ECDF}_i(X^i_F))} \right|$. This metric takes into account that the difficulty of changing from one percentile to another depends on where one starts. For example, going from the 50th percentile to the 55th is typically easier than going from the 90th percentile to the 95th. Large values imply that the counterfactual is more different than the original point, but as we show in this paper this is not necessarily worse.

8. Maximum Percentile Shift: $c(X_F, X_{CF}) = \max_i |\text{ECDF}_i(X^i_{CF}) - \text{ECDF}_i(X^i_F)|$. This gives a sense of the largest effort exerted for a single change relative to the entire population. Large values imply more effort but are not necessarily worse.

9. Minimum Percentile Shift: $c(X_F, X_{CF}) = \min_i |\text{ECDF}_i(X^i_{CF}) - \text{ECDF}_i(X^i_F)|$. This gives a sense of the smallest amount of effort exerted for a single change relative to the entire population. Large values imply more effort but are not necessarily worse.

Additionally, we compute the constraint feasibility score as the harmonic mean of the individual causal constraint satisfaction rates. We compute a constraint feasibility score on the Credit Default, simple causal, and hidden confounding examples.

The individual **causal constraints** used for the Credit Default dataset are:

- Age should not decrease too much. The fact that it decreases at all is more a reflection of the error distribution than a suggestion that one should decrease their age
- Education level does not decrease
- Total months overdue going down implies that age goes up to represent the passage of time
- The maximum bill amount over the past 6 months must be at least as large as the most recent bill amount
- The maximum payment amount over the past 6 months must be at least as large as the most recent payment amount

The individual causal constraints used for the simple causal and hidden confounding examples are:

- An increase in income implies an increase in savings

The individual causal constraints used for the South German Credit examples are:

- Age cannot decrease
- The duration of the loan must be positive
- The amount of the loan must be positive

F.2. Datasets

Synthetic We use 7 synthetic datasets, described in Appendix B.

Real We use the Credit Default dataset (Dua & Graff, 2017) with the same pre-processing as in Ustun et al. (2019) and sample 10% of the records at random to facilitate faster experimentation. The Credit Default dataset has mixed continuous and categorical attributes. The CSVAE likelihood assumes that all attributes are continuous. We use a small likelihood variance and observe empirically that CRUDS

and p-CRUDS largely respect the logical structure of the categorical attributes, though we hope to extend CRUDS to explicitly handle features of mixed type as future work. We standardize the continuous attributes and represent the categoricals with a one-hot encoding. The target of prediction is whether the client ultimately defaulted on their credit.

We use the Wine Quality dataset (Dua & Graff, 2017) and standardize the data. All attributes in the Wine Quality dataset are continuous. The target of prediction is whether the wine is of good quality or not.

Finally, we use the South German Credit dataset (Dua & Graff, 2017), which is an updated version of the German Credit dataset from Karimi et al. (2020). The dataset has mixed continuous and categorical attributes. We use only the 7 most important features as determined by SHAP (Lundberg et al., 2020) and permutation importance. Three of these are continuous: amount (of loan), duration (of loan), and age (hoehe, laufzeit, and alter), and the other 4 are categorical: checking account status, savings account status, credit history, and installment rate (laufkont, sparkont, moral, and rate). The categorical attributes have multiple levels, but we binarize them by deriving indicators from them for whether the applicant has the "best" (or "worst" for installment rate) category for that attribute. For checking account status, this means the applicant has at least 200 Deutsche Marks (DM) in their checking account or income for at least one year. For savings account status, this means the applicant has at least 1000 DM in their savings account. For credit history, this means all credits from the bank to the applicant have been paid back duly. For installment rate, this means that the installment rate is less than twenty percent of the applicant's income. We standardize the continuous attributes and represent the remaining features as binary indicators. We treat credit history as being mutable. The target of prediction is whether the applicant is at low or high risk of default.

G. Qualitative Analysis of the South German Credit Dataset

We present here a qualitative analysis of CRUDS applied to the South German Credit dataset. Table 4c details three counterfactuals recommended by REVISE, p-CRUDS 1, and CRUDS, respectively, for an applicant that was deemed high risk due to insufficient checking and savings account funds, problems with credit history, the long duration of the loan, and large amount of money requested. REVISE and p-CRUDS 1, due to a lack of explicit causal constraint enforcement, both recommend decreasing age. REVISE also recommends increasing the installment payments and the loan amount, the latter of which increases loan risk. p-CRUDS 1 and CRUDS both recommend decreasing the loan amount and duration as well as obtaining more capital before seeking the loan, all of which are interventions which we would intuitively expect to decrease loan risk.

Figure 2 details the options produced by step 4 in section 4. A decision tree of depth three was used to partition the sampled counterfactuals. The leftmost conditions correspond to the rules determined by the decision tree for a particular leaf node and the remaining conditions correspond to the values that the unused attributes take on within the set of counterfactuals in that leaf node. Numeric attributes not used in a decision rule are summarized with a range. Binary attributes are included only if they do not vary within the leaf node. We observe that it is almost always necessary to increase available funds within a checking account, but there is choice in how the other attributes are modified. For example, option 1 and option 8 represent a choice between increasing savings or obtaining a better credit history while option 3 suggests that it is sufficient to acquire a better credit history and increase savings while ignoring the checking account funds. We note that there are redundancies in this option set and that an investigation into how the decision tree parameters affect the quality of the options as well as alternative methods for summarizing the counterfactual distribution remain as future work.

The seven attributes used in this qualitative analysis are easily interpretable and their relationship with loan risk can be reconciled with human judgment in a straightforward manner. This example illustrates that CRUDS can provide sensible recourse where other methods do not while also giving the end user choice in how they achieve the recourse.

H. Experimental Details

Architecture The VAE and CSVAE used the same architectures in all experiments. All encoders, decoders and the $q_\delta(y|z)$ inference network consisted of 1 hidden layer with 64 nodes and ReLU activations.

Hyper-parameters In the pedagogical examples in Appendix B, the VAE/CSVAE's parameters were that of the ground truth data generating process. In all other experiments, $\sigma_z^2 = 0.25$, w is 1-dimensional, drawn from a mixture of Gaussians with means $-1, 1$ (corresponding to $y = 0, y = 1$) and with $\sigma_w^2 = 0.25$. For REVISE, we used $\lambda = 10^{-5}$. A logistic regression was used as the classifier for all experiments except the nonlinear decision boundary synthetic example which used a 3 hidden layer MLP with ReLU activations. For the real datasets, the latent dimensions were chosen using the "ELBOW" method.

For the CSVAE model, the β_i's were selected to ensure that the z's are uncorrelated with the y's, and so that the posterior aggregated over the data matched the prior – both checks were performed visually in the simple models. For the remainder of the hyper-parameters, refer to Table 5.

Optimization We used the Adam optimizer (Kingma & Ba, 2014) with a learning rate of $1e-3$ in all experiments.

Data For the synthetic datasets, train sets consisting of 3000 observations and test sets consisting of 100 observations were generated. For the real datasets, an 80-20 train-test split was performed.

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	1.00	1.00	1.00	1.00
Classifier Probability	0.52	0.64	0.52	0.63
$\ell 2$	1.16	1.62	1.14	1.66
normalized $\ell 1$	0.23	0.34	0.22	0.33
Geodesic (Approx.)	1.27	1.84	1.15	-
Max Percentile Shift	0.28	0.43	0.26	0.43
Min Percentile Shift	0.19	0.35	0.19	0.34
Total Percentile Shift	0.47	0.78	0.45	0.77
Total Log Percentile Shift	0.29	0.75	0.34	0.66
Constraint Feasibility Score	1.00	1.00	1.00	1.00

(a) Simple causal example

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	0.95	0.71	0.90	0.90
Classifier Probability	0.50	0.51	0.51	0.55
$\ell 2$	0.73	0.75	0.81	1.54
normalized $\ell 1$	0.12	0.15	0.16	0.30
Geodesic (Approx.)	0.85	0.79	0.86	-
Max Percentile Shift	0.18	0.18	0.19	0.34
Min Percentile Shift	0.03	0.12	0.13	0.26
Total Percentile Shift	0.21	0.39	0.32	0.60
Total Log Percentile Shift	0.32	0.34	0.35	1.11
Constraint Feasibility Score	0.45	0.11	0.02	0.02

(b) Hidden confounding example

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	1.00	0.98	1.00	1.00
Classifier Probability	0.93	0.77	0.77	0.90
$\ell 2$	2.96	2.01	2.02	2.40
normalized $\ell 1$	0.27	0.19	0.19	0.23
Geodesic (Approx.)	3.13	2.07	2.08	-
Max Percentile Shift	0.23	0.23	0.23	0.25
Min Percentile Shift	0.13	0.13	0.13	0.15
Total Percentile Shift	0.36	0.36	0.36	0.41
Total Log Percentile Shift	1.27	0.36	0.36	0.39
Constraint Feasibility Score	-	-	-	-

(c) Nonlinear example

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	1.00	0.98	1.00	1.00
Classifier Probability	0.51	0.52	0.53	0.64
$\ell 2$	0.79	0.89	0.89	1.87
normalized $\ell 1$	0.54	0.59	0.59	0.95
Geodesic (Approx.)	0.93	0.97	0.98	-
Max Percentile Shift	0.45	0.46	0.46	0.73
Min Percentile Shift	0.25	0.28	0.28	0.27
Total Percentile Shift	0.70	0.74	0.74	1.01
Total Log Percentile Shift	1.08	1.13	1.14	1.73
Constraint Feasibility Score	-	-	-	-

(d) Overlap example

Table 1. Evaluation metrics - synthetic datasets

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
Income	67522.0	93354.0	107010.0	102481.0
Savings	21028.0	27967.0	30142.0	30689.0

(a) Simple causal example

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
Income	60000.00	96050.82	87129.77	104434.91
Savings	-1445.94	-4017.33	-10217.36	-17154.71

(b) Hidden confounding example

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
x_1	1.08	2.98	1.88	2.08
x_2	1.38	3.07	2.81	3.14

(c) Nonlinear example

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
x_1	1.16	0.19	0.20	-1.07
x_2	0.12	0.99	1.05	0.17

(d) Overlap example

Table 2. Proposed recourse - synthetic datasets

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	1.00	1.00	1.00	1.00
Classifier Probability	0.52	0.76	0.52	0.88
$\ell 2$	1.41	2.90	1.28	4.03
normalized $\ell 1$	0.68	1.03	0.55	1.43
Geodesic (Approx.)	1.67	3.31	1.53	-
Max Percentile Shift	0.73	0.73	0.70	0.89
Min Percentile Shift	0.00	0.00	0.00	0.00
Total Percentile Shift	2.37	2.72	2.68	3.54
Total Log Percentile Shift	0.13	0.23	0.10	0.32
Constraint Feasibility Score	0.69	0.81	0.82	1.00

(a) Credit default dataset

Metric	REVISE	p-CRUDS 1	p-CRUDS 2	CRUDS
Target Class Validity	1.00	0.95	1.00	1.00
Classifier Probability	0.56	0.59	0.54	0.83
$\ell 2$	1.60	1.77	1.61	2.74
normalized $\ell 1$	0.46	0.51	0.46	0.82
Geodesic (Approx.)	2.08	2.39	2.13	-
Max Percentile Shift	0.35	0.38	0.35	0.53
Min Percentile Shift	0.01	0.01	0.01	0.02
Total Percentile Shift	1.33	1.44	1.31	2.26
Total Log Percentile Shift	0.30	0.36	0.34	0.51
Constraint Feasibility Score	-	-	-	-

(b) Wine quality dataset

Table 3. Evaluation metrics - real datasets

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
Max Bill Amount (last 6 mos.)	3690.0	3656.0	4454.0	4366.0
Max Payment Amount (last 6 mos.)	310.0	251.0	185.0	267.0
Mos. w/ High Spending (last 6 mos.)	6.0	5.0	5.0	5.0
Most Recent Bill Amount	3430.0	3499.0	4141.0	3808.0
Most Recent Payment Amount	0.0	107.0	104.0	149.0
Total Overdue Counts	1.0	1.0	0.0	0.0
Total Months Overdue	11.0	5.0	1.0	0.0
Age	53.0	53.0	53.0	53.0
Married	1.0	1.0	1.0	1.0

(a) Credit default dataset

Attribute	Original	REVISE	p-CRUDS 1	CRUDS
Fixed Acidity	8.20	7.90	7.98	7.75
Volatile Acidity	0.52	0.42	0.46	0.41
Citric Acid	0.34	0.34	0.30	0.31
Residual Sugar	1.20	4.96	3.90	4.92
Chlorides	0.04	0.04	0.04	0.03
Free Sulfur Dioxide	18.00	21.68	25.24	30.95
Total Sulfur dioxide	167.00	140.60	147.29	132.29
Density	0.99	0.99	0.99	0.99
pH	3.24	3.14	3.15	3.13
Sulphates	0.39	0.52	0.50	0.49
Alcohol	10.60	11.56	11.62	12.80

(b) Wine quality dataset. We take the rating threshold for desirable quality to be ≥ 6.

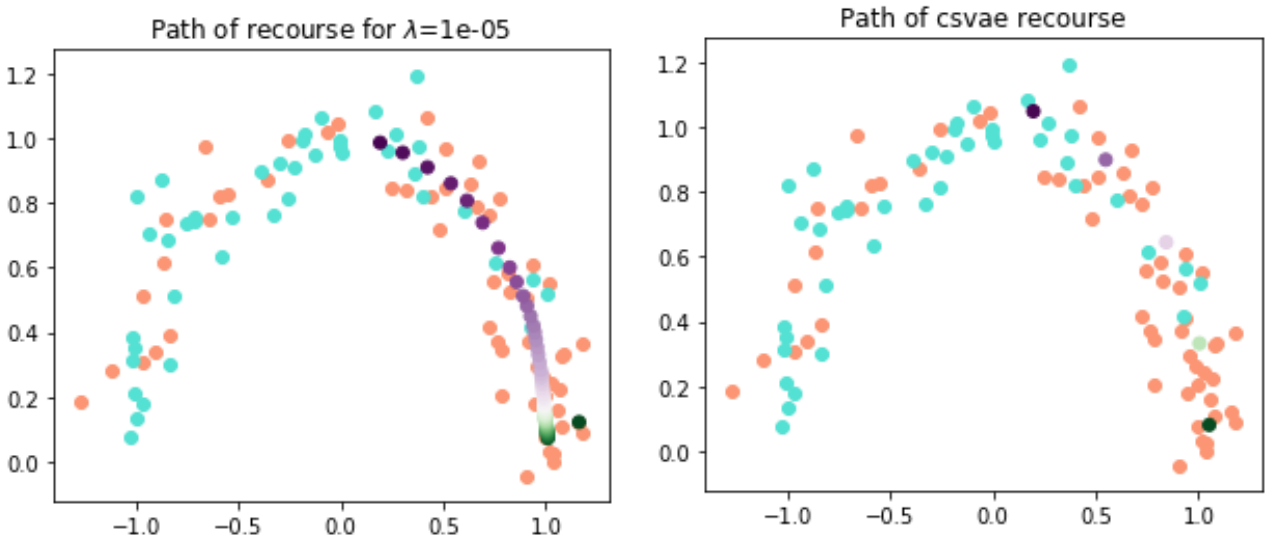
Attribute	Original	REVISE	p-CRUDS 1	CRUDS
Duration of Loan (months)	60	57	49	48
Amount of Loan (DM)	7297	7403	5876	5738
Age of Applicant	36	35	35	36
High Checking Account Funds	0	0	1	1
High Savings Account Funds	0	0	1	1
Good Credit History	0	0	0.0	0.0
Low Installment Rate	1	0	1.0	1

(c) South German credit dataset

Table 4. Proposed recourse - real datasets

Parameter	Simple causal	Hidden confounding	Nonlinear	Overlap	Credit default	Wine quality	South German Credit
VAE z dimensionality	1	2	2	2	6	5	4
VAE Training Epochs	500	500	500	500	500	200	500
VAE σ_ϵ^2	0.25^2	0.1^2	0.25^2	0.1^2	0.1^2	0.25^2	0.1^2
CSVAE z dimensionality	1	1	1	1	6	5	3
CSVAE Training Epochs	500	500	500	500	1000	200	1000
CSVAE σ_ϵ^2	0.25^2	0.1^2	0.25^2	0.1^2	0.1^2	0.25^2	0.1^2
CSVAE β_1	10	5	10	10	10	10	5
CSVAE β_2	5	10	5	5	8	10	15
CSVAE β_3	5	5	5	5	8	5	10
CSVAE β_4	10	100	10	20	100	100	100
CSVAE β_5	1	10	1	2	10	10	10
Classifier	LR	LR	MLP	LR	LR	LR	MLP
REVISE Step Size	0.05	0.05	0.1	0.05	0.05	0.05	0.05
p-CRUDS Step Size	0.01	0.1	0.2	0.05	0.01	0.05	0.1
Train Size	3000	3000	3000	3000	2400	3168	800
Test Size	100	100	100	100	600	793	200

Table 5. Experiment Parameters



(a) **REVISE** takes more gradient steps to produce a counterfactual that flips the classifier's decision in cases where it can't move directly to the decision boundary.

(b) **p-CRUDS** overcomes this problem easily, owing to its optimization with respect to W space which has a clear correlation with the value of y.

Figure 10. **Comparison of Recourse Paths.**. Teal and salmon colored points respectively represent actually observed examples of the two classes and the green to purple gradation represents the transformation from a factual observation to a counterfactual one.

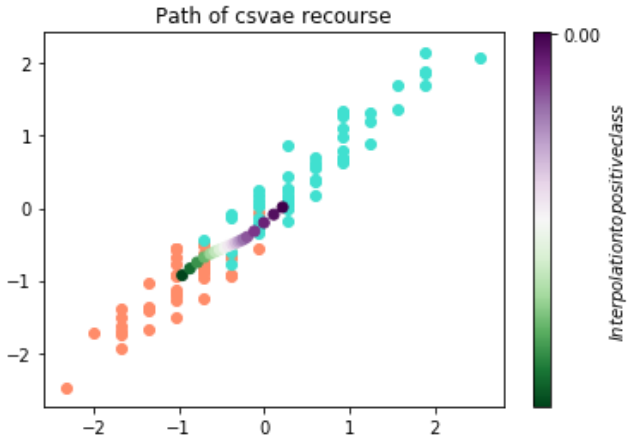
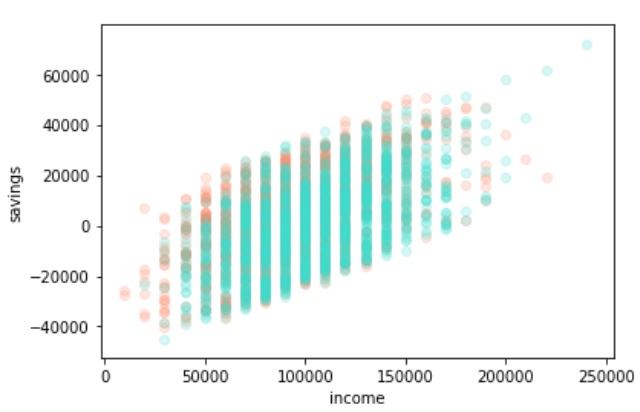
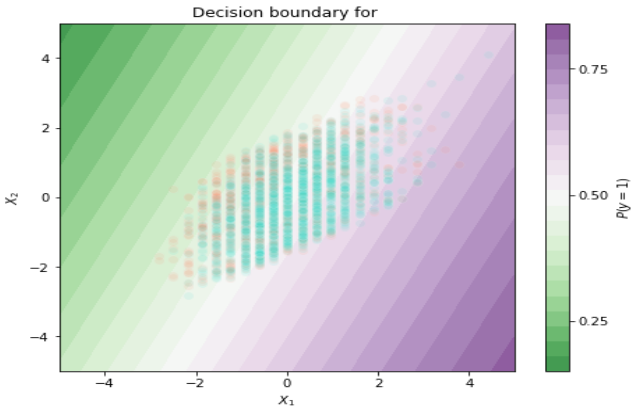


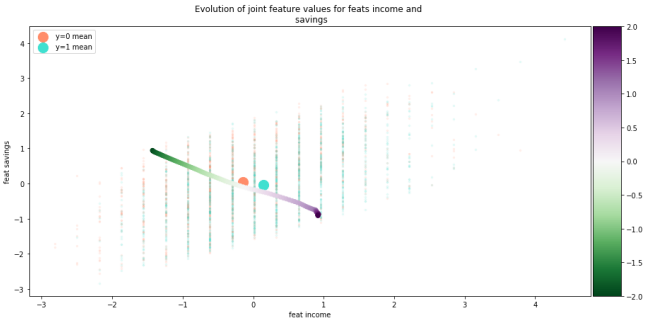
Figure 11. p-CRUDS path for **simple causal** synthetic dataset which follows the generative process in 4a. The path is concise and precise, following the linear structure of the generated data.



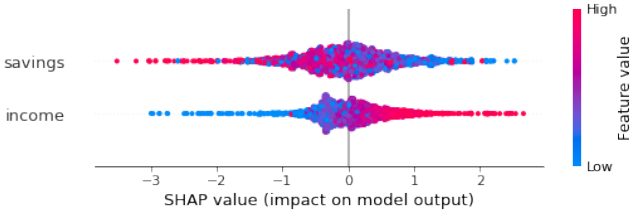
(a) Scatterplot of savings vs income



(b) Linear Decision Rule for hidden confounding dataset



(c) p-CRUDS counterfactuals recommend increasing income and decreasing savings. The true relationship between savings and loan acceptance rate is positive.



(d) SHAP values on XGBoost confirm negative relationship between savings and loan acceptance

Figure 12. **Hidden Confounding Example.** This dataset was constructed to demonstrate that CRUDS only recovers correlations in the dataset and not the underlying causal structure.

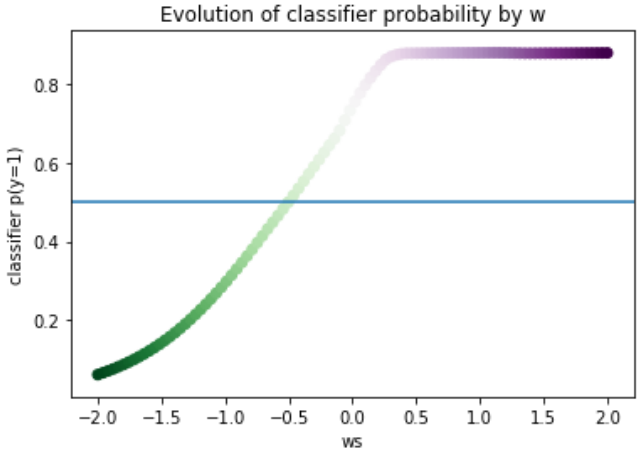


Figure 13. Evolution of $p_{\text{classifier}}(y = 1|x_{\text{CF}})$ for decoded values x_{CF} as latent representation w varies while z is fixed. w is strongly correlated with y.

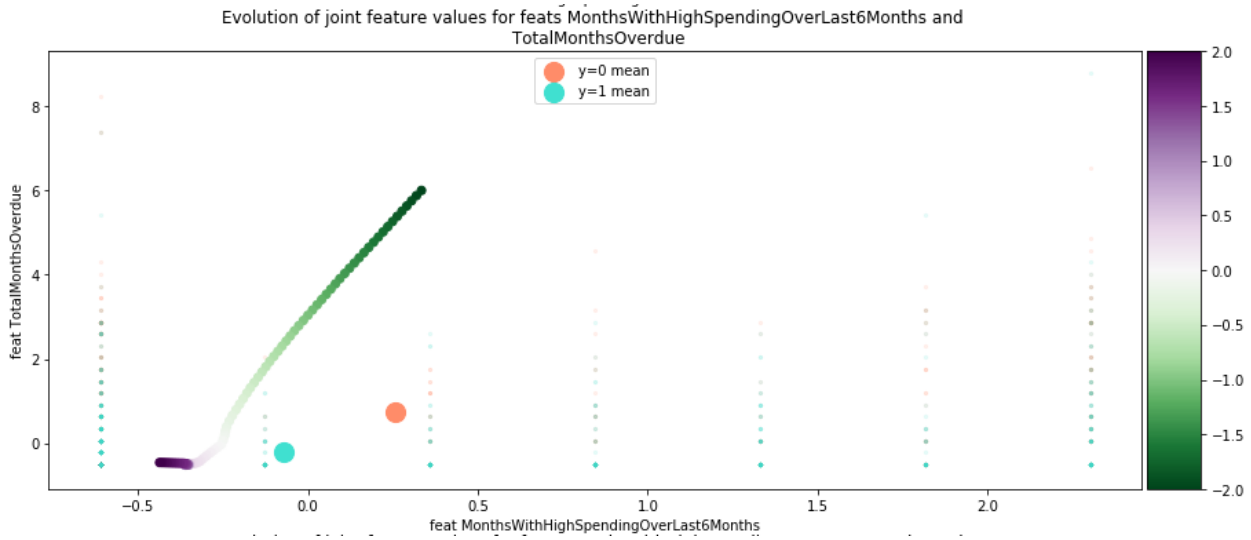


Figure 14. Evolution of the two most important features, total months overdue and months with high spending, for determining whether someone will default on their loan under p-CRUDS for a randomly chosen case in need of recourse. The green to purple path follows the counterfactuals generated from decoding the latent attributes w and z as z is fixed and w is moved from regions of high density for $p(w|y = 0)$ to $p(w|y = 1)$. The salmon and turquoise points represent the joint distribution for these two features. p-CRUDS recommends decreasing both as much as possible. The feature values in this example are normalized.

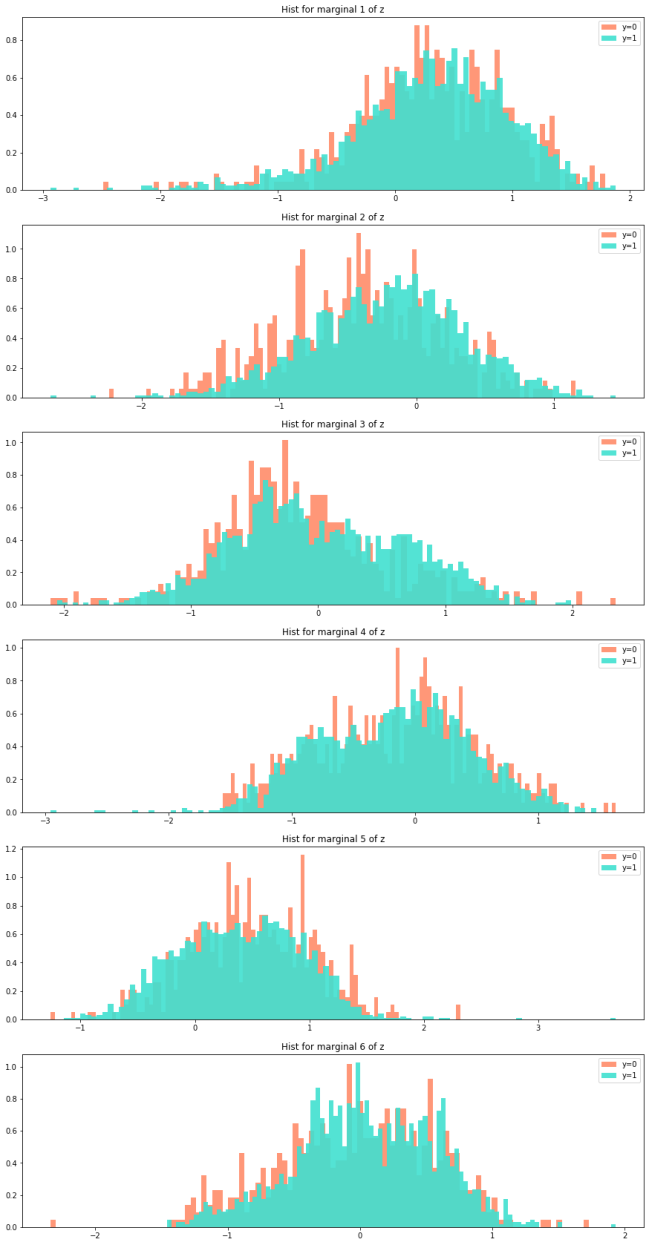


Figure 15. Distribution of encoded z marginals for the **credit default** dataset. As seen from the tightly overlapping distributions, z contains negligible information about y. The marginals also match the prior well.

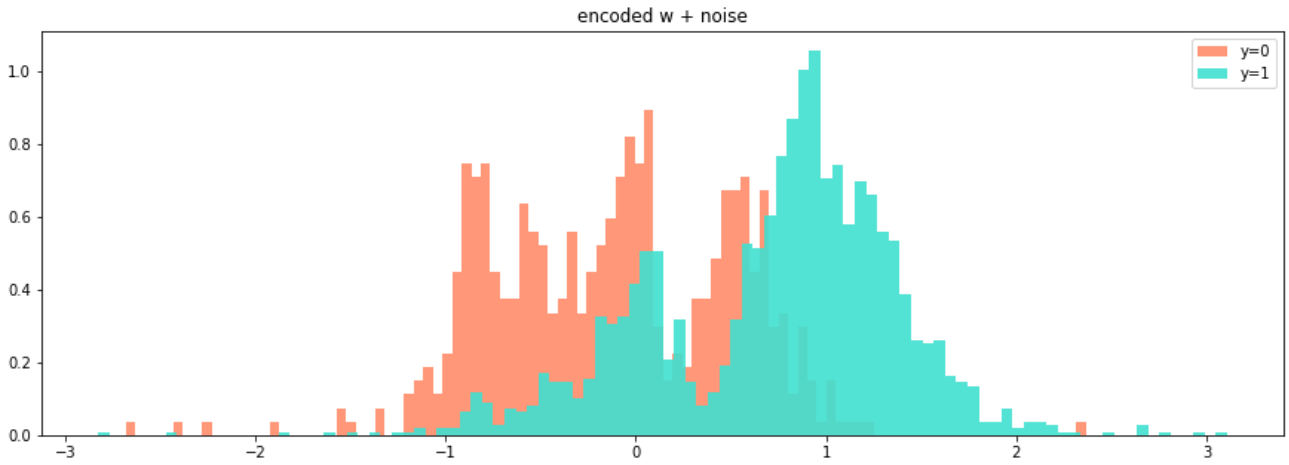


Figure 16. Distribution of encoded w values for the **credit default** dataset. As seen from the notable distance between the distributions, w contains nearly all of the information about y. The encoded w values also match the prior distribution well.