Algorithmic Recourse based on User's Feature-order Preference

Manan Singh[*]
IIT Palakkad, India, India
142214003@smail.iitpkd.ac.in

Sai Srinivas Kancheti[*]
IIT Hyderabad, India, India
cs21resch01004@iith.ac.in

Shivam Gupta[*]
IIT Ropar, India, India
shivam.20csz0004@iitrpr.ac.in

Ganesh Ghalme
IIT Hyderabad, India, India
ganeshghalme@ai.iith.ac.in

Shweta Jain
IIT Ropar, India, India
shwetajain@iitrpr.ac.in

Narayanan C. Krishnan
IIT Palakkad, India, India
ckn@iitpkd.ac.in

ABSTRACT

The state-of-the-art recourse generation methods solely rely on the user's profile (feature vector). However, two users having the same profile may still have different preferences. Consequently, the recourse generated from a single profile may not have the same appeal to both the users. For example, one rejected loan applicant may prefer changes in *Savings Amount*, whereas, another - being a financial expert - may prefer changes in *Investment Amount*. Taking into account these preferences in feature-change can be very helpful in generating more user-satisfying recourses. To this end, we propose a simple user-preference representation and design a method to generate a recourse that adheres to the user preference. We empirically demonstrate the effectiveness and ease of the proposed method at generating recourses satisfying user preferences.

KEYWORDS

Algorithmic decision making, Classification, Counterfactuals

ACM Reference Format:
Manan Singh, Sai Srinivas Kancheti, Shivam Gupta, Ganesh Ghalme, Shweta Jain, and Narayanan C. Krishnan. 2023. Algorithmic Recourse based on User's Feature-order Preference. In *6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD) (CODS-COMAD 2023), January 4–7, 2023, Mumbai, India.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3570991.3571039

1 INTRODUCTION

Algorithmic recourses, also known as counterfactual explanations [3–6], are possibilities on what could overturn the unfavourable outcomes received by the users of a decision system. The explanations help the users to contextualize the decision, and improve themselves in the future. The state-of-the-art methods for recourse generation are based only on the feature-values used by the classifier. The features characterize only certain user aspects ignoring other facets such as interests and preferences. As a result, two users having the same feature-profile may not prefer a single recourse recommendation. They may have different preferences on the features permitted to change. For example, one loan applicant may prefer to increase the *Total Savings amount* more than the increase to the *Investment amount*, whereas another user, being an investment expert, may prefer the opposite.

Thus, it is important to consider the preference for relative-ease in feature-change during recourse generation. A recent work has looked at this problem but has represented the user-preference as a set of feature-specific state-transition functions, which are admittedly not intuitive and easily expressible by the user [7]. In our work, we have assumed a very intuitive and easily expressible preference-representation as an ordering over features; and proposed a simple yet effective method that employs this representation to generate recourses with higher user satisfaction. We use ordinal preference ordering over features to capture the relative ease with which an agent is willing to modify each feature. The choice of ordinal (as opposed to the cardinal values) is grounded on the consumer choice literature [1].

Formally, we state our problem as follows: Given a pre-trained classifier h, an unfavourably-predicted user's feature vector $\mathbf{x} \in \mathbb{R}^d$, the user-defined actionable feature-subset M, and the user's preference over features $f_{(1)} \succ \ldots \succ f_{(i)} \succ \ldots \succ f_{(|M|)}$, where $f_{(i)} \in M$ refers the i^{th} top-priority feature, an example $\mathbf{x'}$ having a favourable outcome has to be generated satisfying two properties. Firstly, the example should be as feature-wise similar to the original vector as possible, and, secondly, the higher-priority features should have changed *more* than the lower-priority features (i.e. $\Delta f_{(i)} \geq \Delta f_{(j)}$ if $f_{(i)} \succ \Delta f_{(j)}$, where $\Delta f_{(i)}$ refers to the amount of change in the feature's value [1] i.e. $|x_{f_{(i)}} - x'_{f_{(i)}}|$). An ideal example having no violations of the feature-order can be termed as a *Preference Compatible Recourse*, but in practice, such counterfactual is not always producible, and thus, we propose a method to generate a recourse having the least number of violations.

2 METHODOLOGY

Our proposed solution is based on the commonly used Wachter's method to generate counterfactuals [4, 6] that optimizes an objective comprising of two terms as shown below:

$$\mathbf{x'} = \underset{\mathbf{x'}}{\operatorname{argmin}}\ \ell(h(\mathbf{x'}), +1) + \lambda\ c(\mathbf{x}, \mathbf{x'})$$

Here, $\ell(.,.)$ denotes classification-loss (like binary cross-entropy), $c(.,.)$ being cost (or L1-distance) and λ as trade-off. The $\ell(.,.)$ ensures that the counterfactual has a positive outcome, and $c(.,.)$ enforces it to be close to the original feature vector. Typically such

[*] Doctoral Students

[1] In this initial work, we assume all the features to be continuous and normalized to a range of 0-1.

CODS-COMAD 2023, January 4–7, 2023, Mumbai, India

ACM ISBN 978-1-4503-9797-1/23/01.
https://doi.org/10.1145/3570991.3571039

Method	Validity(↑)	Best UPS(↑)	Avg. UPS(↑)	Worst UPS(↑)	Avg. Cost(↓)	#Avg. features changed(↓)
Wachter	1.0	0.0899	0.0712	0.0601	5.81	20.77
WachterM	0.88	0.6414	0.5142	0.3795	4.93	12.63
Proposed Method	**0.89**	**0.9099**	**0.7393**	**0.3867**	**1.36**	**3.26**

Table 1: Comparison of different methods. UPS is averaged over 100 users from the valid recourses found in 50 trials/user.

objective functions are optimized using gradient-descent with i iteration steps while allowing all actionable features to change.

As a modification, we perform $|M|$ set of updates, each containing i iterations of gradient descent. We allow only the top $K(1 < K < |M|)$ features to change during each update stage. For simplicity, say, in the first update set, allowing only the top one feature to change, followed by the top two features in the next update set, and so on. A list of valid counterfactuals generated in each update set is maintained. In the end, counterfactual with the least preference violations (if multiple exists, then choosing the one with the lowest cost) gets returned.

Our approach has the benefit that when only a subset of higher-priority features are allowed to change, the lower priority-features remain unchanged. This trivially satisfies many of the user's feature-priorities that involve lower-priority features. When such an approach is repeatedly run while gradually growing the actionable feature-set, it is more likely to generate counterfactuals with lesser user-preference violations. In our experiments, we indeed observe that this approach easily generates preference satisfying recourses.

We compare our approach against two versions of Wachter's method. In the first, called *Wachter*, all actionable features known *globally* are allowed to change, whereas, the second, called *WachterM*, allows only the *user-specified* actionable features to change. As the former method enables all features to change, it becomes easier for the optimizer to find an instance with a favorable outcome. But it increases the risk of generating recourses with a higher degree of user-preference violations. On the contrary, *WachterM* restricts the possible counterfactuals by allowing only limited user specified features to change; discouraging non-preferred features from changing and increasing the user satisfiability. But restricting feature sets render finding a valid counterfactual difficult. So there is a trade-off in both the versions of *Wachter* method, which is also evident from our experiments.

3 RESULTS AND DISCUSSION

We present the results of the above methods on HELOC [2] - a standard credit-approval dataset. It contains $11K$ samples of 23 continuous feature attributes describing users' financial information normalized to $[0, 1]$. The pre-trained binary classifier to be explained by counterfactuals is a Multi Layer Perceptron (MLP) with test accuracy of around 70 % (best in the literature).

We evaluate the performance of different methods [2] on following metrics: (i) **Validity**: The fraction of users for which a valid counterfactual is generated i.e. one having a positive prediction; (ii) **User Preference Satisfaction** (UPS) of a counterfactual $\mathbf{x}'$ for an instance $\mathbf{x}$ with feature-order preference p is computed using:

$$\text{UPS}(\mathbf{x}, \mathbf{x}', p) = 1 - \frac{\text{\# preference-order violations}}{\text{\# all possible preference-orders}} \tag{1}$$

where a preference-order, $f_{(i)} \succ f_{(j)}$, between two features $f_{(i)}$ and $f_{(j)}$ is said to be *violated*, if $f_{(j)}$, the less-preferred feature, is altered more than a more-preferred feature $f_{(i)}$, i.e. if $\Delta x_{f_{(j)}} > \Delta x_{f_{(i)}}$. Note that the UPS belongs to $[0, 1]$, and changing of non-preferred feature resets UPS to 0 for a user; (iii) **Cost** of the best counterfactual[3] for a user (measured by L_1 distance), and (iv) **Avg. features changed**.

The results obtained after generating counterfactuals for 100 negatively predicted users are reported in Table 1.

We can see that when all features are allowed to change, *Wachter* always finds a counterfactual, but permitting feature changes beyond the user's choice causes the method to lower user-satisfaction (UPS of around 0.1). Contrary, in *WachterM*, with user specific actionable features, validity decreases, but user satisfaction increases to 64%. Our method achieves much higher user satisfaction because of its refined search. Along with constraining actionable features (like one in *WachterM*), our method performs a dedicated and narrowed search in the solution space where only a subset of top-priority features changes. This prevents counterfactuals suffering from lower-priority feature-violations (promoting higher UPS values). Additionally, our method modifies fewer features, thus offering low-cost counterfactuals.

4 CONCLUSION AND FUTURE WORK

In this work, we introduce the novel problem of generating recourses that are compatible with user-specified ordinal feature preferences. We demonstrate the inadequacies of traditional unconstrained counterfactual generation methods in giving good solutions, while a simple structured search adhering to the user-specified feature preference can more easily achieve significantly better recourses. As future work, we plan to investigate other solutions to the problem and study guarantees on the user-preference satisfaction of the generated recourses.

REFERENCES

[1] William Barnett. 2003. The modern theory of consumer behavior: Ordinal or cardinal? *The Quarterly Journal of Austrian Economics* 6, 1 (2003), 41–65.

[2] FICO. 2018. Explainable machine learning challenge. https://community.fico.com/s/explainable-machine-learning-challenge.

[3] Amir-Hossein Karimi, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera. 2020. A survey of algorithmic recourse: definitions, formulations, solutions, and prospects. *arXiv preprint arXiv:2010.04050* (2020).

[4] Sohini Upadhyay, Shalmali Joshi, and Himabindu Lakkaraju. 2021. Towards robust and reliable algorithmic recourse. *Advances in Neural Information Processing Systems* 34 (2021).

[5] Sahil Verma, John Dickerson, and Keegan Hines. 2020. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596* (2020).

[6] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.* 31 (2017), 841.

[7] Prateek Yadav, Peter Hase, and Mohit Bansal. 2021. Low-cost algorithmic recourse for users with uncertain cost functions. *arXiv preprint arXiv:2111.01235* (2021).

[2] We take number of iterations, trade-off parameter(λ) , learning-rate to be 100, 0.05, 0.01 respectively to generate counterfactuals across all methods. User-specified preferences were randomly simulated for conducting the experiments.

[3] "Best" refers to the one with the least violations, and if multiple exists then the one with the lowest cost.