COUNTERFACTUAL TRAINING: TEACHING MODELS PLAUSIBLE AND ACTIONABLE EXPLANATIONS

A PREPRINT

Patrick Altmeyer
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

p.altmeyer@tudelft.nl

Aleksander Buszydlik
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Arie van Deursen
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Cynthia C. S. Liem
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

March 14, 2025

ABSTRACT

We propose a novel training regime termed counterfactual training that leverages counterfactual explanations to increase the explanatory capacity of models. Counterfactual explanations have emerged as a popular post-hoc explanation method for opaque machine learning models: they inform how factual inputs would need to change in order for a model to produce some desired output. To be useful in real-word decision-making systems, counterfactuals should be plausible with respect to the underlying data and actionable with respect to the stakeholder requirements. Much existing research has therefore focused on developing post-hoc methods to generate counterfactuals that meet these desiderata. In this work, we instead hold models directly accountable for the desired end goal: counterfactual training employs counterfactuals ad-hoc during the training phase to minimize the divergence between learned representations and plausible, actionable explanations. We demonstrate empirically and theoretically that our proposed method facilitates training models that deliver inherently desirable explanations while maintaining high predictive performance.

Keywords Counterfactual Training • Counterfactual Explanations • Algorithmic Recourse • Explainable AI • Representation Learning

1 Introduction

Today's prominence of artificial intelligence (AI) has largely been driven by **representation learning**: instead of relying on features and rules that are carefully hand-crafted by humans, modern machine learning (ML) models are tasked

with learning representations directly from data, guided by narrow objectives such as predictive accuracy (Goodfellow, Bengio, and Courville 2016). Modern advances in computing have made it possible to provide such models with ever-growing degrees of freedom to achieve that task, which frequently allows them to outperform traditionally more parsimonious models. Unfortunately, in doing so, models learn increasingly complex and highly sensitive representations that humans can no longer easily interpret.

The trend towards complexity for the sake of performance has come under serious scrutiny in recent years. At the very cusp of the deep learning (DL) revolution, Szegedy et al. (2014) showed that artificial neural networks (ANN) are sensitive to adversarial examples (AEs): perturbed versions of data instances that yield vastly different model predictions despite being "imperceptible" in that they are semantically indifferent from their factual counterparts. Even though some partially effective mitigation strategies have been proposed—most notably **adversarial training** (Goodfellow, Shlens, and Szegedy 2015)—truly robust deep learning remains unattainable even for models that are considered "shallow" by today's standards (Kolter 2023).

Part of the problem is that the high degrees of freedom provide room for many solutions that are locally optimal with respect to narrow objectives (Wilson 2020).[1] Indeed, recent work on the so-called "lottery ticket hypothesis" suggests that modern neural networks can be pruned by up to 90% while preserving their predictive performance (Frankle and Carbin 2019). Similarly, Zhang et al. (2021) showed that state-of-the-art neural networks are expressive enough to fit randomly labeled data. Thus, looking at the predictive performance alone, the solutions may seem to provide compelling explanations for the data, when in fact they are based on purely associative, semantically meaningless patterns. This poses two challenges. Firstly, there is no dependable way to verify if representations correspond to meaningful, plausible explanations. Secondly, even if we could resolve the first challenge, it remains undecided how to ensure that models can *only* learn valuable explanations.

The first challenge has attracted an abundance of research on **explainable AI** (XAI), a paradigm that focuses on the development of tools to derive (post-hoc) explanations from complex model representations. Such explanations should mitigate a scenario in which practitioners deploy opaque models and blindly rely on their predictions. On countless occasions, this has happened in practice and caused real harms to people who were adversely and unfairly affected by automated decision-making (ADM) systems involving opaque models (O'Neil 2016; McGregor 2021). Effective XAI tools can aid us in monitoring models and providing recourse to individuals to turn negative outcomes (e.g., "loan application rejected") into positive ones (e.g., "application accepted"). Our work builds upon **counterfactual explanations** (CE) proposed by Wachter, Mittelstadt, and Russell (2017) as an effective approach to achieve this goal. CEs prescribe minimal changes for factual inputs that, if implemented, would prompt some fitted model to produce a desired output.

To our surprise, the second challenge has not yet attracted major research interest. Specifically, there has been no concerted effort towards improving the "explanatory capacity" of models, i.e., the degree to which learned representations correspond to explanations that are **interpretable** and deemed **plausible** by humans (see Def. 3.1). Instead, the choice has generally been to improve the ability of XAI tools to identify the subset of explanations that are both plausible and valid for any given model, independent of whether the learned representations are also compatible with plausible explanations (Altmeyer et al. 2024). Fortunately, recent findings indicate that improved explanatory capacity can arise as a consequence of regularization techniques aimed at other training objectives such as robustness, generalization, and generative capacity (Schut et al. 2021; Augustin, Meinke, and Hein 2020; Altmeyer et al. 2024). As further discussed in Section 2, our work consolidates these findings within a single objective.

Specifically, we propose counterfactual training (CT): a novel training regime that aligns learned representations with plausible explanations respecting actionability constraints. The remainder of this paper is structured as follows. Section 2 presents related work, focusing on the link between adversarial examples and counterfactual explanations. Then follow our two principal contributions. In Section 3, we introduce our methodological framework and show theoretically that it can be employed to enforce global actionability constraints. In Section 4, through extensive experiments, we demonstrate that CT substantially improves explainability without sacrificing predictive performance. We discuss the challenges in Section 5 and conclude in Section 6 that CT is a promising approach towards making opaque models more trustworthy.

2 Related Literature

To the best of our knowledge, the proposed framework for counterfactual training represents the first attempt to use counterfactual explanations during training to improve model explainability. In high-level terms, we define model explainability as the extent to which valid explanations derived for an opaque model are also deemed plausible with respect to the underlying data and (global) actionability constraints. To make the desiderata for our framework more

[1] We follow the standard ML convention, where "degrees of freedom" refer to the number of parameters estimated from data.

concrete, we follow Augustin, Meinke, and Hein (2020) in tying the concept of explainability to the quality of CEs that can be generated for a given model. The authors show that CEs—understood as minimal input perturbations that yield some desired model prediction—tend to be more meaningful if the underlying model is more robust to adversarial examples. We can make intuitive sense of this finding when looking at adversarial training (AT) through the lens of representation learning with high degrees of freedom. As argued before, learned representations may be sensitive to producing implausible explanations and mispredicting for worst-case counterfactuals (i.e., AEs). Thus, by inducing models to "unlearn" susceptiblity to such examples, AT can effectively remove implausible explanations from the solution space.

2.1 Adversarial Examples are Counterfactual Explanations

This interpretation of the link between explainability through counterfactuals on one side and robustness to adversarial examples on the other is backed by empirical evidence. Sauer and Geiger (2021) demonstrate that using counterfactual images during classifier training improves model robustness. Similarly, Abbasnejad et al. (2020) argue that counterfactuals represent potentially useful training data in machine learning, especially in supervised settings where inputs may be reasonably mapped to multiple outputs. They, too, demonstrate that augmenting the training data of image classifiers can improve generalization. Finally, Teney, Abbasnedjad, and Hengel (2020) propose an approach using counterfactuals in training that does not rely on data augmentation: they argue that counterfactual pairs typically already exist in training datasets. Specifically, their approach relies on identifying similar input samples with different annotations and ensuring that the gradient of the classifier aligns with the vector between such pairs of counterfactual inputs using the cosine distance as the loss function.

In the natural language processing (NLP) domain, CEs have also been used to improve models through data augmentation. Wu et al. (2021) propose *Polyjuice*, a general-purpose counterfactual generator for language models. The authors empirically demonstrate that the augmentation of training data with their method improves robustness in a number of NLP tasks. Balashankar et al. (2023) similarly use *Polyjuice* to augment NLP datasets through diverse counterfactuals and show that classifier robustness improves by up to 20%. Finally, Luu and Inoue (2023) introduce Counterfactual Adversarial Training (CAT), which also aims at improving generalization and robustness of language models through a three-step procedure: the authors identify training samples that are subject to high predictive uncertainty, generate CEs for those samples, and then fine-tune the language model on a dataset augmented with the CEs.

There have also been several attempts at formalizing the relationship between counterfactual explanations and adversarial examples. Pointing to clear similarities in how CEs and AEs are generated, Freiesleben (2022) makes the case for jointly studying the opaqueness and robustness problems in representation learning. Formally, AEs can be seen as the subset of CEs for which misclassification is achieved (Freiesleben 2022). Similarly, Pawelczyk et al. (2022) show that CEs and AEs are equivalent under certain conditions and derive theoretical upper bounds on distances between them.

Two recent works are closely related to ours in that they use counterfactuals during training with the explicit goal of affecting certain properties of the post-hoc counterfactual explanations. Firstly, Ross, Lakkaraju, and Bastani (2024) propose a way to train models that guarantee individual recourse to some positive target class with high probability. Their approach builds on adversarial training by explicitly inducing susceptibility to targeted adversarial examples for the positive class. Additionally, the proposed method allows for imposing a set of actionability constraints ex-ante. For example, users can specify that certain features are immutable. Secondly, Guo, Nguyen, and Yadav (2023) are the first to propose an end-to-end training pipeline that includes CEs as part of the training procedure. In particular, they propose a specific network architecture that includes a predictor and CE generator network, where the parameters of the CE generator network are learnable. Counterfactuals are generated during each training iteration and fed back to the predictor network. In contrast to Guo, Nguyen, and Yadav (2023), we impose no restrictions on the ANN architecture at all.

2.2 Beyond Robustness

Improving the adversarial robustness of models is not the only path towards aligning representations with plausible explanations. In a work closely related to this one, Altmeyer et al. (2024) show that explainability can be improved through model averaging and refined model objectives. The authors propose a way to generate counterfactuals that are maximally faithful to the model in that they are consistent with what the model has learned about the underlying data. Formally, they rely on tools from energy-based modelling to minimize the divergence between the distribution of counterfactuals and the conditional posterior over inputs learned by the model. Their proposed counterfactual explainer, *ECCCo*, yields plausible explanations if and only if the underlying model has learned representations that align with them. The authors find that both deep ensembles (Lakshminarayanan, Pritzel, and Blundell 2017) and joint energy-based models (JEMs) (Grathwohl et al. 2020) tend to do well in this regard.

Once again it helps to look at these findings through the lens of representation learning with high degrees of freedom. Deep ensembles are approximate Bayesian model averages, which are most called for when models are underspecified by the available data (Wilson 2020). Averaging across solutions mitigates the aforementioned risk of relying on a single locally optimal representations that corresponds to semantically meaningless explanations for the data. Previous work by Schut et al. (2021) similarly found that generating plausible ("interpretable") counterfactual explanations is almost trivial for deep ensembles that have also undergone adversarial training. The case for JEMs is even clearer: they involve a hybrid objective that induces both high predictive performance and generative capacity (Grathwohl et al. 2020). This is closely related to the idea of aligning models with plausible explanations and has inspired our proposed CT objective, as we explain in Section 3.

3 Counterfactual Training

Counterfactual training combines ideas from adversarial training, energy-based modelling and counterfactuals explanations with the explicit goal of aligning representations with plausible explanations that comply with user requirements. In the context of CEs, plausibility has broadly been defined as the degree to which counterfactuals comply with the underlying data-generating process (Poyiadzi et al. 2020; Guidotti 2022; Altmeyer et al. 2024). Plausibility is a necessary but insufficient condition for using CEs to provide algorithmic recourse (AR) to individuals (negatively) affected by opaque models. For AR recommendations to be actionable, they need to not only result in plausible counterfactuals but also be attainable. A plausible CE for a rejected 20-year-old loan applicant, for example, might reveal that their application would have been accepted, if only they were 20 years older. Ignoring all other features, this would comply with the definition of plausibility if 40-year-old individuals were in fact more credit-worthy on average than young adults. But of course this CE does not qualify for providing actionable recourse to the applicant since *age* is not a (directly) mutable feature. CT aims to improve model explainability by aligning models with counterfactuals that meet both desiderata: plausibility and actionability. Formally, we define explainability as follows:

Definition 3.1 (Model Explainability)**.** Let $\mathbf{M}_\theta : \mathcal{X} \mapsto \mathcal{Y}$ denote a supervised classification model that maps from the D-dimensional input space $\mathcal{X}$ to representations $\phi(\mathbf{x};\theta)$ and finally to the K-dimensional output space $\mathcal{Y}$. Assume that for any given input-output pair $\{\mathbf{x},\mathbf{y}\}_i$ there exists a counterfactual $\mathbf{x}' = \mathbf{x} + \Delta : \mathbf{M}_\theta(\mathbf{x}') = \mathbf{y}^+ \neq \mathbf{y} = \mathbf{M}_\theta(\mathbf{x})$ where $\arg\max_y \mathbf{y}^+ = y^+$ and y^+ denotes the index of the target class.

We say that $\mathbf{M}_\theta$ is **explainable** to the extent that faithfully generated counterfactuals are plausible and actionable. We define these properties as follows:

1. (Plausibility) $\int^A p(\mathbf{x}'|\mathbf{y}^+)d\mathbf{x} \to 1$ where A is some small region around $\mathbf{x}'$.
2. (Actionability) Permutations Δ are subject to some actionability constraints.
3. (Faithfulness) $\int^A p_\theta(\mathbf{x}'|\mathbf{y}^+)d\mathbf{x} \to 1$ where A is defined as above.

where $p_\theta(\mathbf{x}|\mathbf{y}^+)$ denotes the conditional posterior over inputs.

The characterization of faithfulness and plausibility in Def. 3.1 is the same as in Altmeyer et al. (2024), with adapted notation. Intuitively, plausible counterfactuals are consistent with the data and faithful counterfactuals are consistent with what the model has learned about input data. Actionability constraints in Def. 3.1 vary and depend on the context in which $\mathbf{M}_\theta$ is deployed. In this work, we focus on domain and mutability constraints for individual features x_d for $d = 1, ..., D$. We limit ourselves to classification tasks for reasons discussed in Section 5.

3.1 Our Proposed Objective

Let $\mathbf{x}'_t$ for $t = 0, ..., T$ denote a counterfactual explanation generated through gradient descent over T iterations as initially proposed by Wachter, Mittelstadt, and Russell (2017). For our purposes, we let T vary and consider the counterfactual search as converged as soon as the predicted probability for the target class has reached a pre-determined threshold, τ: $\mathcal{S}(\mathbf{M}_\theta(\mathbf{x}'))[y^+] \geq \tau$, where $\mathcal{S}$ is the softmax function.[2]

To train models with high explainability as defined in Def. 3.1, we propose to leverage counterfactuals in the following objective:

$$\min_\theta \text{yloss}(\mathbf{M}_\theta(\mathbf{x}), \mathbf{y}) + \lambda_{\text{div}}\text{div}(\mathbf{x}^+, \mathbf{x}'_T, y; \theta) + \lambda_{\text{adv}}\text{advloss}(\mathbf{M}_\theta(\mathbf{x}'_{t\leq T}), \mathbf{y}) + \lambda_{\text{reg}}\text{ridge}(\mathbf{x}^+, \mathbf{x}'_T, y; \theta) \tag{1}$$

[2]For detailed background information on gradient-based counterfactual search and convergence see supplementary appendix.

where yloss$(\cdot)$ is a classification loss that induces discriminative performance (e.g., cross-entropy). The second and third terms are explained in detail below. For now, they can be summarized as inducing explainability directly and indirectly by penalizing: (1) the contrastive divergence, div$(\cdot)$, between mature counterfactuals $\mathbf{x}'_T$ and observed samples $\mathbf{x}^+ \in \mathcal{X}^+ = \{\mathbf{x} : y = y^+\}$ in the target class y^+, and, (2) the adversarial loss, advloss$(.)$, with respect to nascent counterfactuals $\mathbf{x}'_{t\leq T}$. Finally, ridge$(\cdot)$ denotes a Ridge penalty (ℓ_2-norm) that regularizes the magnitude of the energy terms involved in div$(\cdot)$ (Du and Mordatch 2020). The trade-off between the components can governed through penalties λ_{div}, λ_{adv} and λ_{reg}.

3.2 Directly Inducing Explainability with Contrastive Divergence

As observed by Grathwohl et al. (2020), any classifier can be re-interpreted as a joint energy-based model (JEM) that learns to discriminate output classes conditional on the observed (training) samples from $p(\mathbf{x})$ and the generated samples from $p_\theta(\mathbf{x})$. The authors show that JEMs can be trained to perform well at both tasks by directly maximizing the joint log-likelihood factorized as $\log p_\theta(\mathbf{x}, \mathbf{y}) = \log p_\theta(\mathbf{y}|\mathbf{x}) + \log p_\theta(\mathbf{x})$. The first term can be optimized using conventional cross-entropy as in Equation 1. Then, to optimize $\log p_\theta(\mathbf{x})$ Grathwohl et al. (2020) minimize the contrastive divergence between these observed samples from $p(\mathbf{x})$ and generated samples from $p_\theta(\mathbf{x})$.

A key empirical finding in Altmeyer et al. (2024) was that JEMs tend to do well with respect to the plausibility objective in Def. 3.1. This follows directly if we consider samples drawn from $p_\theta(\mathbf{x})$ as counterfactuals because the JEM objective effectively minimizes the divergence between the conditional posterior and $p(\mathbf{x}|\mathbf{y}^+)$. To generate samples, Grathwohl et al. (2020) rely on Stochastic Gradient Langevin Dynamics (SGLD) using an uninformative prior for initialization but we depart from their methodology. Instead of SGLD, we propose to use counterfactual explainers to generate counterfactuals of observed training samples. Specifically, we have:

$$\text{div}(\mathbf{x}^+, \mathbf{x}'_T, y; \theta) = \mathcal{E}_\theta(\mathbf{x}^+, y) - \mathcal{E}_\theta(\mathbf{x}'_T, y) \tag{2}$$

where $\mathcal{E}_\theta(\cdot)$ denotes the energy function. We set $\mathcal{E}_\theta(\mathbf{x}, y) = -\mathbf{M}_\theta(\mathbf{x})[y^+]$ where y^+ denotes the index of the randomly drawn target class, $y^+ \sim p(y)$. Conditional on the target class y^+, $\mathbf{x}'_T$ denotes a mature counterfactual for a randomly sampled factual from a non-target class generated with a gradient-based CE generator for up to T iterations. Mature counterfactuals are ones that have either reached convergence wrt. the decision threshold τ or exhausted T.

Intuitively, the gradient of Equation 2 decreases the energy of observed training samples (positive samples) while increasing the energy of counterfactuals (negative samples) (Du and Mordatch 2020). As the counterfactuals get more plausible (Def. 3.1) during training, these opposing effects gradually balance each other out (Lippe 2024).

The departure from SGLD allows us to tap into the vast repertoire of explainers that have been proposed in the literature to meet different desiderata. For example, many methods facilitate the imposition of domain and mutability constraints. In principle, any existing approach for generating counterfactual explanations is viable, so long as it does not violate the faithfulness condition. Like JEMs (Murphy 2022), CT can be considered a form of contrastive representation learning.

3.3 Indirectly Inducing Explainability with Adversarial Robustness

Based on our analysis in Section 2, counterfactuals $\mathbf{x}'$ can be repurposed as additional training samples (Luu and Inoue 2023; Balashankar et al. 2023) or AEs (Freiesleben 2022; Pawelczyk et al. 2022). This leaves some flexibility with respect to the choice for advloss$(\cdot)$ in Equation 1. An intuitive functional form, but likely not the only sensible choice, is inspired by adversarial training:

$$\begin{aligned} \text{advloss}(\mathbf{M}_\theta(\mathbf{x}'_{t\leq T}), \mathbf{y}; \varepsilon) &= \text{yloss}(\mathbf{M}_\theta(\mathbf{x}'_{t_\varepsilon}), \mathbf{y}) \\ t_\varepsilon &= \max_t \{t : ||\Delta_t||_\infty < \varepsilon\} \end{aligned} \tag{3}$$

Under this choice, we consider nascent counterfactuals $\mathbf{x}'_{t\leq T}$ as AEs as long as the magnitude of the perturbation to any single feature is at most ε. This is closely aligned with Szegedy et al. (2014) who define an adversarial attack as an "imperceptible non-random perturbation". Thus, we choose to work with a different distinction between CE and AE than Freiesleben (2022) who consider misclassification as the key distinguishing feature of AE. One of the key observations in this work is that we can leverage CEs during training and get adversarial examples essentially for free.

3.4 Encoding Actionability Constraints

Many existing counterfactual explainers support domain and mutability constraints out-of-the-box. In fact, both types of constraints can be implemented for any counterfactual explainer that relies on gradient descent in the feature space for optimization (Altmeyer, Deursen, and Liem 2023). In this context, domain constraints can be imposed by simply projecting counterfactuals back to the specified domain, if the previous gradient step resulted in updated feature values

that were out-of-domain. Mutability constraints can similarly be enforced by setting partial derivatives to zero to ensure that features are only perturbed in the allowed direction, if at all.

Since such actionability constraints are binding at test time, we should also impose them when generating $\mathbf{x}'$ during each training iteration to inform model representations. Through their effect on $\mathbf{x}'$, both types of constraints influence model outcomes via Equation 2. Here it is crucial that we avoid penalizing implausibility that arises due to mutability constraints. For any mutability-constrained feature d this can be achieved by enforcing $\mathbf{x}^+[d] - \mathbf{x}'[d] := 0$ whenever perturbing $\mathbf{x}'[d]$ in the direction of $\mathbf{x}^+[d]$ would violate mutability constraints. Specifically, we set $\mathbf{x}^+[d] := \mathbf{x}'[d]$ if:

1. Feature d is strictly immutable in practice.
2. We have $\mathbf{x}^+[d] > \mathbf{x}'[d]$, but feature d can only be decreased in practice.
3. We have $\mathbf{x}^+[d] < \mathbf{x}'[d]$, but feature d can only be increased in practice.

From a Bayesian perspective, setting $\mathbf{x}^+[d] := \mathbf{x}'[d]$ can be understood as assuming a point mass prior for $p(\mathbf{x}^+)$ with respect to feature d. Intuitively, we think of this simply in terms ignoring implausibility costs with respect to immutable features, which effectively forces the model to instead seek plausibility with respect to the remaining features. This in turn results in lower overall sensitivity to immutable features, which we demonstrate empirically for different classifiers in Section 4. Under certain conditions, this result holds theoretically:[3]

Proposition 3.1 (Protecting Immutable Features)**.** *Let $f_\theta(\mathbf{x}) = \mathcal{S}(\mathbf{M}_\theta(\mathbf{x})) = \mathcal{S}(\Theta\mathbf{x})$ denote a linear classifier with softmax activation $\mathcal{S}$ where $y \in \{1, ..., K\} = \mathcal{K}$ and $\mathbf{x} \in \mathbb{R}^D$. If we assume multivariate Gaussian class densities with common diagonal covariance matrix $\Sigma_k = \Sigma$ for all $k \in \mathcal{K}$, then protecting an immutable feature from the contrastive divergence penalty will result in lower classifier sensitivity to that feature relative to the remaining features, provided that at least one of those is discriminative and mutable.*

It is worth highlighting that Proposition 3.1 assumes independence of features. This raises a valid concern about the effect of protecting immutable features in the presence of proxies that remain unprotected. We address this in Section 5.

3.5 Example (Prediction of Consumer Credit Default)

Suppose we are interested in predicting the likelihood that loan applicants default on their credit. We have access to historical data on previous loan takers comprised of a binary outcome variable ($y \in \{1 = \text{default}, 2 = \text{no default}\}$) with two input features: (1) the subjects' *age*, which we define as immutable, and (2) the subjects' existing level of *debt*, which we define as mutable.

We have simulated this scenario using synthetic data with two independent features and Gaussian class-conditional densities in Figure 1. The four panels show the outcomes for different training procedures using the same model architectures (a linear classifier). In panels (a) and (c) we have trained the models conventionally, while in panels (b) and (d) we used CT. Only in panels (c) and (d) do we impose the mutability constraint on *age* at test time. In each case, we show the decision boundary (in green) and the training data colored according to their ground-truth label: orange points belong to the target class, $y^+ = 2$, blue points belong to the non-target class, $y^- = 1$. Stars indicate CEs in the target class generated at test time using generic gradient descent until convergence.

In all cases the counterfactuals are valid but their quality differs. In panel (a), they are not plausible: they do not comply with the distribution of the factuals in y^+ to the point where they form a clearly distinguishable cluster. In panel (b), they are highly plausible, meeting the first objective of Def. 3.1. In panel (c), the CEs involve substantial reductions in *debt* for younger applicants. By comparison, counterfactual paths are shorter on average in panel (d) where we have protected the immutable *age* as described in Section 3.4. Due to the classifier's lower sensitivity to *age*, recommendations with respect to *debt* are much more homogenous and do not unfairly punish younger individuals. These counterfactuals are also plausible with respect to the mutable feature. Thus, we consider the model in panel (d) as the most explainable according to Def. 3.1.

4 Experiments

Here we present the experiments conducted to answer our research questions:

1. To what extent does our counterfactual training objective as defined in Equation 1 induce models to learn plausible explanations?
2. To what extent does the CT objective produce more favorable algorithmic recourse outcomes in the presence of actionability constraints?
3. What are the effects of hyperparameter selection wrt. the CT objective?

[3] For the proof, see the supplementary appendix.

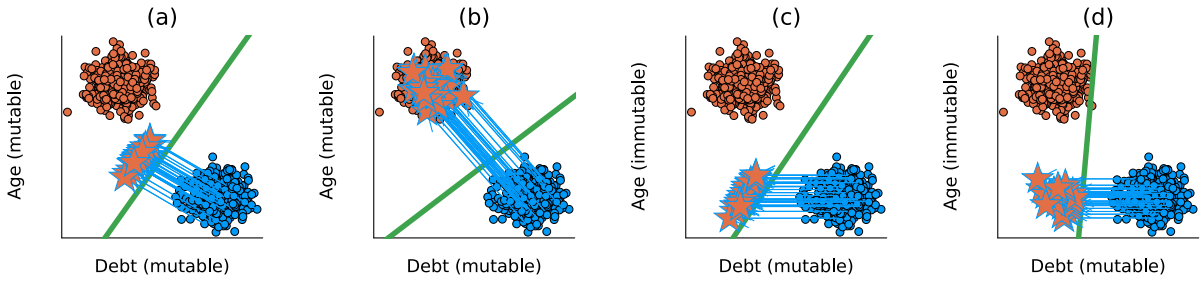

Figure 1: Illustration of how CT improves model explainability.

4.1 Experimental Setup

Our key outcome of interest is how well do models perform with respect to explainability (Def. 3.1). To this end, we focus primarily on the plausibility and cost of faithfully generated counterfactuals at test time. To measure the cost of counterfactuals, we follow the standard convention of using distances (ℓ_1-norm) between factuals and counterfactuals as a proxy. For plausibility, we assess how similar counterfactuals are to observed samples in the target domain, $\mathbf{X}^+ \subset \mathcal{X}^+$. We rely on the distance-based metric used by Altmeyer et al. (2024),

$$\text{IP}(\mathbf{x}', \mathbf{X}^+) = \frac{1}{|\mathbf{X}^+|} \sum_{\mathbf{x} \in \mathbf{X}^+} \text{dist}(\mathbf{x}', \mathbf{x}) \tag{4}$$

and introduce a novel divergence metric,

$$\text{IP}^*(\mathbf{X}', \mathbf{X}^+) = \text{MMD}(\mathbf{X}', \mathbf{X}^+) \tag{5}$$

where $\mathbf{X}'$ denotes a collection of counterfactuals and $\text{MMD}(\cdot)$ is an unbiased estimate of the squared population maximum mean discrepancy (Gretton et al. 2012). The metric in Equation 5 is equal to zero iff the two distributions are the same, $\mathbf{X}' = \mathbf{X}^+$.

In addition to cost and plausibility, we also compute other standard metrics to evaluate counterfactuals at test time including validity and redundancy. Finally, we also assess the predictive performance of models using standard metrics.

We run the experiments with three gradient-based generators: *Generic* of Wachter, Mittelstadt, and Russell (2017) as a simple baseline approach, *REVISE* (Joshi et al. 2019) that aims to generate plausible counterfactuals using a surrogate Variational Autoencoder (VAE), and *ECCo*—the generator of Altmeyer et al. (2024) but without the conformal prediction component—as a method that directly targets both faithfulness and plausibility of the CEs.

4.2 Experimental Results

4.2.1 Plausibility.

Table 1 presents our main empirical findings. The top five rows show the percentage reduction in implausibility according to Equation 4 for varying degrees of the energy penalty used for *ECCo* at test time. Row 6 shows the reduction in implausibility as measured by Equation 5 and aggregated across all test specifications of *ECCo*. Rows 7 and 8 show the test accuracies for the model trained with CT and conventionally trained models ("vanilla").

For all datasets except *OL* and across all test settings, the average distance of CEs from observed samples in the target class is reduced, indicating improved plausibility. The magnitude of improvements varies. For the simple synthetic datasets, distance reductions range from around 20-40% (*LS*, *Moon*) to almost 60% (*Circ*). For the real-world tabular datasets, improvements tend to be smaller but still substantial, with around 10-15% for *CH*, 11-28% for *GMSC*, 7-8% for *Cred* and around 3% for *Adult*. For our only vision dataset (*MNIST*), distances are reduced by up to 9%. The results for our proposed divergence metric are qualitatively similar, but generally even more pronounced: for the *Circ* dataset, implausibility is reduced by almost 94% to virtually zero as we verified by the absolute outcome. Improvements for other datasets range from 28% (*Moon*) to 78% (*GMSC*). For *OL* the reduction is negative, consistent with the distance-based metric. *MNIST* is the only dataset for which the two metrics disagree.

These broad and substantial improvements in plausibility generally do not come at the cost of decreased predictive performance: test accuracy for CT is virtually identical to the baseline for *Adult*, *Circ*, *LS*, *Moon* and *OL*, and even slightly improved for *Cred*. Exceptions to this general pattern are *MNIST*, *CH* and *GMSC*, for which we observe reduction in test accuracy of 2, 5 and 15 percentage points, respectively. We note in this context, that we have not

Table 1: Key explainability and predictive performance metrics for all datasets. **Plausibility**: the top five rows show the percentage reduction in implausibility according to Equation 4 for varying degrees of the energy penalty used for *ECCo* at test time. The following row shows the reduction in implausibility as measured by Equation 5 and aggregated across all test specifications. **Accuracy**: The following two rows show the test accuracies for the models trained with CT and the baseline. **Actionability**: The final row present the average reduction in costs when imposing mutability constraints.

Measure	λ_{egy}	Adult	CH	Circ	Cred	GMSC	LS	MNIST	Moon	OL
IP $(-\Delta\%)$	0.1	2.93	9.59	56.5	6.7	11	27.1	9.11	20.4	-6.72
IP $(-\Delta\%)$	0.5	3.4	9.26	57.1	6.18	13.4	26.7	8.26	21.4	-6.19
IP $(-\Delta\%)$	1	3.53	10.4	56.5	7.19	13.4	26.6	8.07	21.6	-6.1
IP $(-\Delta\%)$	5	2.88	11.9	58.5	7.01	21.4	27.1	6.1	19	-2.77
IP $(-\Delta\%)$	10	3.15	14.6	49.3	7.78	27.9	38.6	3.53	19.8	-1.44
IP* $(-\Delta\%)$	(agg.)	34.8	66.6	93.4	51.6	77.9	54.5	-2.28	27.6	-25.5
Acc. (CT)		0.848	0.794	0.997	0.712	0.608	1	0.902	0.999	0.918
Acc. (vanilla)		0.854	0.85	0.999	0.706	0.751	1	0.922	1	0.914
Cost $(-\Delta\%)$				35			26.3		23.4	15.5

optimized our models for predictive performance at all and worked with very small networks. In summary, we find that CT can substantially improve the quality of explanations learned by models without sacrificing predictive accuracy.

4.2.2 Actionability.

In Section 3, we show that our proposed way for encoding mutability constraints leads to lower classifier sensitivity wrt. immutable features for linear models, tilting the decision boundary in favour of mutable features instead. For binding constraints at test time, this has the effect of shorter counterfactual paths and hence smaller average costs to individuals. To extend this to the non-linear case, we test the effect of imposing mutability constraints empirically for our synthetic data using the same evaluation scheme as above. The final row in Table 1 reports the average reduction in costs for CT compared to the baseline, when imposing that either the first or the second feature is immutable. In all cases, costs are reduced substantially indicating that classifiers trained with CT are indeed more sensitive to mutable features.

4.2.3 Impact of hyperparameter settings.

We test the impact of three types of hyperparameters; our complete results are in the supplementary appendix.

Hyperparameters of the CE generators. First, we observe that CT is highly sensitive to hyperparameter settings but (a) there are manageable patterns and (b) we can typically identify settings that improve either plausibility or cost, and commonly both of them at the same time. Second, we note that the choice of a CE generator has a major impact on the results. For example, *REVISE* tends to perform the worst, most likely because it uses a surrogate VAE to generate counterfactuals which impedes faithfulness (Altmeyer et al. 2024). Third, increasing T, the maximum number of steps, generally yields better outcomes because more CEs can mature in each training epoch. Fourth, the impact of τ, the required decision threshold is more difficult to predict. On "harder" datasets it may be difficult to satisfy high τ for any given sample (i.e., also factuals) and so increasing this threshold does not seem to correlate with better outcomes. In fact, we have generally found that a choice of $\tau = 0.5$ leads to optimal results because it is associated with high proportions of mature counterfactuals.

Hyperparameters for penalties. We find that the strength of the energy regularization, λ_{reg} is highly impactful; energy must be sufficiently regularized to avoid poor performance in terms of decreased plausibility and increased costs. The sensitivity with respect to λ_{div} and λ_{adv} is much less evident. While high values of λ_{reg} may increase the variability in outcomes when combined with high values of λ_{div} or λ_{adv}, this effect is not very pronounced.

Other hyperparameters. We observe that the effectiveness and stability of CT is positively associated with the number of counterfactuals generated during each training epoch. We also confirm that a higher number of training epochs is beneficial. Interestingly, we observed desired improvements in explainability when CT was combined with conventional training and applied only for the final 50% of epochs of the complete training process. Put differently, CT may be a way to improve the explainability of models in a fine-tuning manner.

5 Discussion

As our results indicate, counterfactual training produces models that are more explainable. However, our approach is not without limitations.

CT increases the training time of models. CT can be more time-consuming than conventional training regimes.

While higher numbers of CEs per iteration positively impact the quality of solutions, they also increase the amount of computations. Relatively small grids with 270 settings can take almost four hours for more demanding datasets on a high-performance computing cluster with 34 2GB CPUs.[4] Three factors attenuate this effect. First, CT amortizes the cost of CEs for the training samples. Second, we find that it can retain its value when used as a "fine-tuning" technique for conventionally-trained models. Third, it yields itself to parallel execution, which we have leveraged for our own experiments.

Immutable features may have proxies. We propose an approach to protect immutable features and thus increase the actionability of the generated CEs. However, it requires that model owners define the mutability constraints for (all) features considered by the model. Even if all immutable features are protected, there may exist proxies that are mutable (and hence should not be protected) but preserve enough information about the principals to hinder the protections. Delineating actionability is a major undecided challenge in the AR literature (see, e.g., Venkatasubramanian and Alfano 2020) impacting the capacity of CT to fulfill its intended goal.

Interventions on features may impact fairness. We provide a tool that allows practitioners to modify the sensitivity of a model with respect to certain features, which may have implication for the fair and equitable treatment of decision subjects. As protecting a set of features leads the model to assign higher relative importance to unprotected features, model owners could misuse our solution by enforcing explanations based on features that are more difficult to modify by some (group of) individuals. For example, consider the Adult dataset used in our experiments, where *workclass* or *education* may be more difficult to change for underpriviledged groups. When applied irresponsibly, CT could result in an unfairly assigned burden of recourse (e.g., Sharma, Henderson, and Ghosh 2020), threatening the equality of opportunity in the system (Bell et al. 2024). Still, these phenomena are not specific to CT.

We also highlight several interesting directions for future research.

Extending CT beyond classification settings. Our formulation relies on the distinction between non-target class(es) y^- and target class(es) y^+ to generate counterfactuals through Equation 1. While y^- and y^+ can be arbitrarily defined, CT requires the output space $\mathcal{Y}$ to be discrete. Thus, it does not apply to ML tasks where the change in outcome cannot be readily quantified. Focus on classification models is a common restriction in research on CEs and AR. Other settings have attracted some interest (e.g., regression in (Spooner et al. 2021; Zhao, Broelemann, and Kasneci 2023)), but there is little consensus how to robustly extend the notion of CEs.

Addressing the training instabilities. JEMs are susceptible to instabilities during training (Grathwohl et al. 2020) and even though we depart from the SGLD-based sampling, we still encounter major variability in the outcomes. CT is exposed to two potential sources of instabilities: (1) the energy-based contrastive divergence term in Equation 2, and (2) the underlying counterfactual explainers. Still, we find that training instabilities can be successfully mitigated by regularizing energy (λ_{reg}), generating sufficiently many counterfactuals during each training epoch, and including only mature counterfactuals for contrastive divergence.

Improving hyperparameter selection procedures. Our method benefits from the tuning of certain key hyperparameters (see Section 4.2.3). In this work, we have relied exclusively on grid search for this task. Future work on CT could benefit from investigating more sophisticated approaches towards hyperparameter tuning. Notably, CT is iterative which makes a variety of methods applicable, including Bayesian (e.g., Snoek, Larochelle, and Adams 2012) or gradient-based (e.g., Franceschi et al. 2017) optimization.

6 Conclusion

State-of-the-art machine learning models are prone to learning complex representations that cannot be interpreted by humans and existing post-hoc explainability approaches cannot guarantee that the explanations agree with the model's learned representation of data. As a step towards addressing this challenge, we introduced counterfactual training, a novel training regime that incentivizes highly-explainable models. Our approach leads to explanations that are both plausible—compliant with the underlying data-generating process—and actionable—compliant with user-specified mutability constraints—and thus meaningful to their recipients. Through extensive experiments we demonstrate that CT satisfies its objectives while preserving the predictive performance of the models. Our approach can also be used to fine-tune conventionally-trained models and achieve similar gains in explainability. Finally, this work showcases that it is practical to improve models *and* their explanations at the same time.

References

Abbasnejad, Ehsan, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. 2020. "Counterfactual Vision and Language Learning." In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10041–51. https://doi.org/10.1109/CVPR42600.2020.01006.

Altmeyer, Patrick, Arie van Deursen, and Cynthia C. S. Liem. 2023. "Explaining Black-Box Models through Counterfactuals." In *Proceedings of the JuliaCon Conferences*, 1:130. 1.

[4] See supplementary appendix for computational details.

Altmeyer, Patrick, Mojtaba Farmanbar, Arie van Deursen, and Cynthia C. S. Liem. 2024. "Faithful Model Explanations through Energy-Constrained Conformal Counterfactuals." In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence*, 38:10829–37. 10. https://doi.org/10.1609/aaai.v38i10.28956.

Augustin, Maximilian, Alexander Meinke, and Matthias Hein. 2020. "Adversarial Robustness on In- and Out-Distribution Improves Explainability." In *Computer Vision – ECCV 2020*, edited by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, 228–45. Cham: Springer.

Balashankar, Ananth, Xuezhi Wang, Yao Qin, Ben Packer, Nithum Thain, Ed Chi, Jilin Chen, and Alex Beutel. 2023. "Improving Classifier Robustness through Active Generative Counterfactual Data Augmentation." In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 127–39. ACL. https://doi.org/10.18653/v1/2023.findings-emnlp.10.

Bell, Andrew, Joao Fonseca, Carlo Abrate, Francesco Bonchi, and Julia Stoyanovich. 2024. "Fairness in Algorithmic Recourse Through the Lens of Substantive Equality of Opportunity." https://arxiv.org/abs/2401.16088.

Bezanson, Jeff, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. "Julia: A Fresh Approach to Numerical Computing." *SIAM Review* 59 (1): 65–98. https://doi.org/10.1137/141000671.

Bouchet-Valat, Milan, and Bogumi Kamiski. 2023. "DataFrames.jl: Flexible and Fast Tabular Data in Julia." *Journal of Statistical Software* 107 (4): 1–32. https://doi.org/10.18637/jss.v107.i04.

Byrne, Simon, Lucas C. Wilcox, and Valentin Churavy. 2021. "MPI.jl: Julia Bindings for the Message Passing Interface." *Proceedings of the JuliaCon Conferences* 1 (1): 68. https://doi.org/10.21105/jcon.00068.

Chagas, Ronan Arraes Jardim, Ben Baumgold, Glen Hertz, Hendrik Ranocha, Mark Wells, Nathan Boyer, Nicholas Ritchie, et al. 2024. "Ronisbr/PrettyTables.jl: V2.4.0." Zenodo. https://doi.org/10.5281/zenodo.13835553.

Christ, Simon, Daniel Schwabeneder, Christopher Rackauckas, Michael Krabbe Borregaard, and Thomas Breloff. 2023. "Plots.jl – a User Extendable Plotting API for the Julia Programming Language." https://doi.org/https://doi.org/10.5334/jors.431.

Danisch, Simon, and Julius Krumbiegel. 2021. "Makie.jl: Flexible High-Performance Data Visualization for Julia." *Journal of Open Source Software* 6 (65): 3349. https://doi.org/10.21105/joss.03349.

Du, Yilun, and Igor Mordatch. 2020. "Implicit Generation and Generalization in Energy-Based Models." https://arxiv.org/abs/1903.08689.

Franceschi, Luca, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. 2017. "Forward and Reverse Gradient-Based Hyperparameter Optimization." In *Proceedings of the 34th International Conference on Machine Learning*, 1165–73. ICML'17. Sydney, NSW, Australia: JMLR.org.

Frankle, Jonathan, and Michael Carbin. 2019. "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks." In *International Conference on Learning Representations*.

Freiesleben, Timo. 2022. "The Intriguing Relation Between Counterfactual Explanations and Adversarial Examples." *Minds and Machines* 32 (1): 77–109.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Goodfellow, Ian, Jonathon Shlens, and Christian Szegedy. 2015. "Explaining and Harnessing Adversarial Examples." https://arxiv.org/abs/1412.6572.

Grathwohl, Will, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. 2020. "Your Classifier Is Secretly an Energy Based Model and You Should Treat It Like One." In *International Conference on Learning Representations*.

Gretton, Arthur, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. "A Kernel Two-Sample Test." *The Journal of Machine Learning Research* 13 (1): 723–73.

Guidotti, Riccardo. 2022. "Counterfactual Explanations and How to Find Them: Literature Review and Benchmarking." *Data Mining and Knowledge Discovery* 38 (5): 2770–2824. https://doi.org/10.1007/s10618-022-00831-6.

Guo, Hangzhi, Thanh H. Nguyen, and Amulya Yadav. 2023. "CounterNet: End-to-End Training of Prediction Aware Counterfactual Explanations." In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 577--589. KDD '23. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3580305.3599290.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. Springer New York. https://doi.org/10.1007/978-0-387-84858-7.

Innes, Michael, Elliot Saba, Keno Fischer, Dhairya Gandhi, Marco Concetto Rudilosso, Neethu Mariya Joy, Tejan Karmali, Avik Pal, and Viral Shah. 2018. "Fashionable Modelling with Flux." https://arxiv.org/abs/1811.01457.

Innes, Mike. 2018. "Flux: Elegant Machine Learning with Julia." *Journal of Open Source Software* 3 (25): 602. https://doi.org/10.21105/joss.00602.

Joshi, Shalmali, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. "Towards Realistic Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems." https://arxiv.org/abs/1907.09615.

Kolter, Zico. 2023."Keynote Addresses: SaTML 2023 ." In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/SaTML54575.2023.00009.

Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. 2017. "Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles." In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 6405–16. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.

Lippe, Phillip. 2024. "UvA Deep Learning Tutorials." https://uvadlc-notebooks.readthedocs.io/en/latest/.

Luu, Hoai Linh, and Naoya Inoue. 2023. "Counterfactual Adversarial Training for Improving Robustness of Pretrained Language Models." In *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation*, 881–88. ACL. https://aclanthology.org/2023.paclic-1.88/.

McGregor, Sean. 2021. "Preventing repeated real world AI failures by cataloging incidents: The AI incident database." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:15458–63. 17.

Murphy, Kevin P. 2022. *Probabilistic Machine Learning: An Introduction*. MIT Press.

O'Neil, Cathy. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown.

Pawelczyk, Martin, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. 2022. "Exploring Counterfactual Explanations Through the Lens of Adversarial Examples: A Theoretical and Empirical Analysis." In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, edited by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, 151:4574–94. Proceedings of Machine Learning Research. PMLR. https://proceedings.mlr.press/v151/pawelczyk22a.html.

Poyiadzi, Rafael, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. "FACE: Feasible and Actionable Counterfactual Explanations." In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 344–50.

Ross, Alexis, Himabindu Lakkaraju, and Osbert Bastani. 2024. "Learning Models for Actionable Recourse." In *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc.

Sauer, Axel, and Andreas Geiger. 2021. "Counterfactual Generative Networks." https://arxiv.org/abs/2101.06046.

Schut, Lisa, Oscar Key, Rory McGrath, Luca Costabello, Bogdan Sacaleanu, Yarin Gal, et al. 2021. "Generating Interpretable Counterfactual Explanations By Implicit Minimisation of Epistemic and Aleatoric Uncertainties." In *International Conference on Artificial Intelligence and Statistics*, 1756–64. PMLR.

Sharma, Shubham, Jette Henderson, and Joydeep Ghosh. 2020. "CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-box Models." In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 166–72. AIES '20. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3375627.3375812.

Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams. 2012. "Practical Bayesian Optimization of Machine Learning Algorithms." In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, edited by F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger. Vol. 25. NIPS'12. Curran Associates, Inc.

Spooner, Thomas, Danial Dervovic, Jason Long, Jon Shepard, Jiahao Chen, and Daniele Magazzeni. 2021. "Counterfactual Explanations for Arbitrary Regression Models." https://arxiv.org/abs/2106.15212.

Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. "Intriguing Properties of Neural Networks." https://arxiv.org/abs/1312.6199.

Teney, Damien, Ehsan Abbasnedjad, and Anton van den Hengel. 2020. "Learning What Makes a Difference from Counterfactual Examples and Gradient Supervision." In *Computer Vision - ECCV 2020*, 580–99. Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-030-58607-2_34.

Venkatasubramanian, Suresh, and Mark Alfano. 2020. "The Philosophical Basis of Algorithmic Recourse." In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 284–93. FAT* '20. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3351095.3372876.

Wachter, Sandra, Brent Mittelstadt, and Chris Russell. 2017. "Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR." *Harv. JL & Tech.* 31: 841. https://doi.org/10.2139/ssrn.3063289.

Wilson, Andrew Gordon. 2020. "The Case for Bayesian Deep Learning." https://arxiv.org/abs/2001.10995.

Wu, Tongshuang, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2021. "Polyjuice: Generating Counterfactuals for Explaining, Evaluating, and Improving Models." In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, edited by Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, 6707–23. Online: ACL. https://doi.org/10.18653/v1/2021.acl-long.523.

Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. "Understanding Deep Learning (Still) Requires Rethinking Generalization." *Commun. ACM* 64 (3): 107–15. https://doi.org/10.1145/3446776.

Zhao, Xuan, Klaus Broelemann, and Gjergji Kasneci. 2023. "Counterfactual Explanation for Regression via Disentanglement in Latent Space." In *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, 976–84. Los Alamitos, CA, USA: IEEE Computer Society. https://doi.org/10.1109/ICDMW60847.2023.00130.

Appendix A Notation

- y^+: The target class and also the index of the target class.
- y^-: The non-target class and also the index of non-the target class.
- $\mathbf{y}^+$: The one-hot encoded output vector for the target class.
- θ: Model parameters (unspecified).
- Θ: Matrix of parameters.

A.1 Other Technical Details

$$
\begin{aligned}
MMD(X', \tilde{X}') = & \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} k(x_i, x_j) \\
& + \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k(\tilde{x}_i, \tilde{x}_j) \\
& - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k(x_i, \tilde{x}_j)
\end{aligned} \tag{6}
$$

In our implementation, Equation 6 is by default applied to the entire subset of the training data for which $y = y^+$.

Appendix B Technical Details of Our Approach

B.1 Generating Counterfactuals through Gradient Descent

In this section, we provide some background on gradient-based counterfactual generators (Section B.1.1) and discuss how we define convergence in this context (Section B.1.2).

B.1.1 Background

Gradient-based counterfactual search was originally proposed by Wachter, Mittelstadt, and Russell (2017). It generally solves the following unconstrained objective,

$$
\min_{\mathbf{z}' \in \mathcal{Z}^L} \{\text{yloss}(\mathbf{M}_\theta(g(\mathbf{z}')), \mathbf{y}^+) + \lambda \text{cost}(g(\mathbf{z}'))\}
$$

where $g : \mathcal{Z} \mapsto \mathcal{X}$ is an invertible function that maps from the L-dimensional counterfactual state space to the feature space and $\text{cost}(\cdot)$ denotes one or more penalties that are used to induce certain properties of the counterfactual outcome. As above, $\mathbf{y}^+$ denotes the target output and $\mathbf{M}_\theta(\mathbf{x})$ returns the logit predictions of the underlying classifier for $\mathbf{x} = g(\mathbf{z})$.

For all generators used in this work we use standard logit crossentropy loss for $\text{yloss}(\cdot)$. All generators also penalize the distance (ℓ_1-norm) of counterfactuals from their original factual state. For *Generic* and *ECCo*, we have $\mathcal{Z} := \mathcal{X}$ and $g(\mathbf{z}) = g(\mathbf{z})^{-1} = \mathbf{z}$, that is counterfactual are searched directly in the feature space. Conversely, *REVISE* traverses the latent space of a variational autoencoder (VAE) fitted to the training data, where $g(\cdot)$ corresponds to the decoder (Joshi et al. 2019). In addition to the distance penalty, *ECCo* uses an additional penalty component that regularizes the energy associated with the counterfactual, $\mathbf{x}'$ (Altmeyer et al. 2024).

B.1.2 Convergence

An important consideration when generating counterfactual explanations using gradient-based methods is how to define convergence. Two common choices are to 1) perform gradient descent over a fixed number of iterations T, or 2) conclude the search as soon as the predicted probability for the target class has reached a pre-determined threshold, τ: $\mathcal{S}(\mathbf{M}_\theta(\mathbf{x}'))[y^+] \geq \tau$. We prefer the latter for our purposes, because it explicitly defines convergence in terms of the black-box model, $\mathbf{M}(\mathbf{x})$.

Defining convergence in this way allows for a more intuitive interpretation of the resulting counterfactual outcomes than with fixed T. Specifically, it allows us to think of counterfactuals as explaining 'high-confidence' predictions by the model for the target class y^+. Depending on the context and application, different choices of τ can be considered as representing 'high-confidence' predictions.

B.2 Protecting Mutability Constraints with Linear Classifiers

In Section 3.4 we explain that to avoid penalizing implausibility that arises due to mutability constraints, we impose a point mass prior on $p(\mathbf{x})$ for the corresponding feature. We argue in Section 3.4 that this approach induces models to

be less sensitive to immutable features and demonstrate this empirically in Section 4. Below we derive the analytical results in Prp.~3.1.

Proof. Let d_{mtbl} and d_{immtbl} denote some mutable and immutable feature, respectively. Suppose that $\mu_{y^-,d_{\text{immtbl}}} < \mu_{y^+,d_{\text{immtbl}}}$ and $\mu_{y^-,d_{\text{mtbl}}} > \mu_{y^+,d_{\text{mtbl}}}$, where $\mu_{k,d}$ denotes the conditional sample mean of feature d in class k. In words, we assume that the immutable feature tends to take lower values for samples in the non-target class y^- than in the target class y^+. We assume the opposite to hold for the mutable feature.

Assuming multivariate Gaussian class densities with common diagonal covariance matrix $\Sigma_k = \Sigma$ for all $k \in \mathcal{K}$, we have for the log likelihood ratio between any two classes $k, m \in \mathcal{K}$ (Hastie, Tibshirani, and Friedman 2009):

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \mathbf{x}^\top \Sigma^{-1}(\mu_k - \mu_m) + \text{const} \tag{7}$$

By independence of $x_1, ..., x_D$, the full log-likelihood ratio decomposes into:

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^{D} \frac{\mu_{k,d} - \mu_{m,d}}{\sigma_d^2} x_d + \text{const} \tag{8}$$

By the properties of our classifier (*multinomial logistic regression*), we have:

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^{D} \left(\theta_{k,d} - \theta_{m,d}\right) x_d + \text{const} \tag{9}$$

where $\theta_{k,d} = \Theta[k, d]$ denotes the coefficient on feature d for class k.

Based on Equation 8 and Equation 9 we can identify that $(\mu_{k,d} - \mu_{m,d}) \propto (\theta_{k,d} - \theta_{m,d})$ under the assumptions we made above. Hence, we have that $(\theta_{y^-,d_{\text{immtbl}}} - \theta_{y^+,d_{\text{immtbl}}}) < 0$ and $(\theta_{y^-,d_{\text{mtbl}}} - \theta_{y^+,d_{\text{mtbl}}}) > 0$

Let $\mathbf{x}'$ denote some randomly chosen individual from class y^- and let $y^+ \sim p(y)$ denote the randomly chosen target class. Then the partial derivative of the contrastive divergence penalty Equation 2 with respect to coefficient $\theta_{y^+,d}$ is equal to

$$\frac{\partial}{\partial \theta_{y^+,d}} \left(\text{div}(\mathbf{x}, \mathbf{x}', \mathbf{y}; \theta)\right) = \frac{\partial}{\partial \theta_{y^+,d}} \left((-\mathbf{M}_\theta(\mathbf{x})[y^+]) - (-\mathbf{M}_\theta(\mathbf{x}')[y^+])\right) = x'_d - x_d \tag{10}$$

and equal to zero everywhere else.

Since $(\mu_{y^-,d_{\text{immtbl}}} < \mu_{y^+,d_{\text{immtbl}}})$ we are more likely to have $(x'_{d_{\text{immtbl}}} - x_{d_{\text{immtbl}}}) < 0$ than vice versa at initialization. Similarly, we are more likely to have $(x'_{d_{\text{mtbl}}} - x_{d_{\text{mtbl}}}) > 0$ since $(\mu_{y^-,d_{\text{mtbl}}} > \mu_{y^+,d_{\text{mtbl}}})$.

This implies that if we do not protect feature d_{immtbl}, the contrastive divergence penalty will decrease $\theta_{y^-,d_{\text{immtbl}}}$ thereby exacerbating the existing effect $(\theta_{y^-,d_{\text{immtbl}}} - \theta_{y^+,d_{\text{immtbl}}}) < 0$. In words, not protecting the immutable feature would have the undesirable effect of making the classifier more sensitive to this feature, in that it would be more likely to predict class y^- as opposed to y^+ for lower values of d_{immtbl}.

By the same rationale, the contrastive divergence penalty can generally be expected to increase $\theta_{y^-,d_{\text{mtbl}}}$ exacerbating $(\theta_{y^-,d_{\text{mtbl}}} - \theta_{y^+,d_{\text{mtbl}}}) > 0$. In words, this has the effect of making the classifier more sensitive to the mutable feature, in that it would be more likely to predict class y^- as opposed to y^+ for higher values of d_{mtbl}.

Thus, our proposed approach of protecting feature d_{immtbl} has the net affect of decreasing the classifier's sensitivity to the immutable feature relative to the mutable feature (i.e. no change in sensitivity for d_{immtbl} relative to increased sensitivity for d_{mtbl}). □

B.3 Domain Constraints

We apply domain constraints on counterfactuals during training and evaluation. There are at least two good reasons for doing so. Firstly, within the context of explainability and algorithmic recourse, real-world attributes are often domain constrained: the *age* feature, for example, is lower bounded by zero and upper bounded by the maximum human

Table 2: Final hyperparameters used for the main results for the different datasets.

Data	No. Train	No. Test	Batchsize	Domain	Decision Threshold	No. Counterfactuals	λ_{reg}
Adult	$2.6 \cdot 10^4$	$5.01 \cdot 10^3$	$1 \cdot 10^3$	none	0.75	$5 \cdot 10^3$	0.25
CH	$1.65 \cdot 10^4$	$3.1 \cdot 10^3$	$1 \cdot 10^3$	none	0.5	$5 \cdot 10^3$	0.25
Circ	$3.6 \cdot 10^3$	600	30	none	0.5	$1 \cdot 10^3$	0.5
Cred	$1.06 \cdot 10^4$	$1.92 \cdot 10^3$	$1 \cdot 10^3$	none	0.5	$5 \cdot 10^3$	0.25
GMSC	$1.34 \cdot 10^4$	$2.47 \cdot 10^3$	$1 \cdot 10^3$	none	0.5	$5 \cdot 10^3$	0.5
LS	$3.6 \cdot 10^3$	600	30	none	0.5	$1 \cdot 10^3$	0.01
MNIST	$1.1 \cdot 10^4$	$2 \cdot 10^3$	$1 \cdot 10^3$	(-1.0, 1.0)	0.5	$5 \cdot 10^3$	0.01
Moon	$3.6 \cdot 10^3$	600	30	none	0.9	$1 \cdot 10^3$	0.25
OL	$3.6 \cdot 10^3$	600	30	none	0.5	$1 \cdot 10^3$	0.25

lifespan. Secondly, domain constraints help mitigate training instabilities commonly associated with energy-based modelling (Grathwohl et al. 2020; Altmeyer et al. 2024).

For our image datasets, features are pixel values and hence the domain is constrained by the lower and upper bound of values that pixels can take depending on how they are scaled (in our case $[-1, 1]$). For all other features d in our synthetic and tabular datasets, we automatically infer domain constraints $[x_d^{\text{LB}}, x_d^{\text{UB}}]$ as follows,

$$
\begin{aligned}
x_d^{\text{LB}} &= \arg\min_{x_d} \{\mu_d - n_{\sigma_d}\sigma_d, \arg\min_{x_d} x_d\} \\
x_d^{\text{UB}} &= \arg\max_{x_d} \{\mu_d + n_{\sigma_d}\sigma_d, \arg\max_{x_d} x_d\}
\end{aligned}
\tag{11}
$$

where μ_d and σ_d denote the sample mean and standard deviation of feature d. We set $n_{\sigma_d} = 3$ across the board but higher values and hence wider bounds may be appropriate depending on the application.

B.4 Training Details

In this section, we describe the training procedure in detail. While the details laid out here are not crucial for understanding our proposed approach, they are of importance to anyone looking to implement counterfactual training.

Appendix C Details on Main Experiments

C.1 Final Hyperparameters

As discussed Section 4, CT is sensitive to certain hyperparameter choices. We study the effect of many hyperparameters extensively in Section D. For the main results, we tune a small set of key hyperparameters (Section E). The final choices for the main results are presented for each data set in Table 2 along with training, test and batch sizes.

C.2 Qualitative Findings for Image Data

Figure 2 shows much more plausible (faithful) counterfactuals for a model with CT than the model with conventional training (Figure 3).

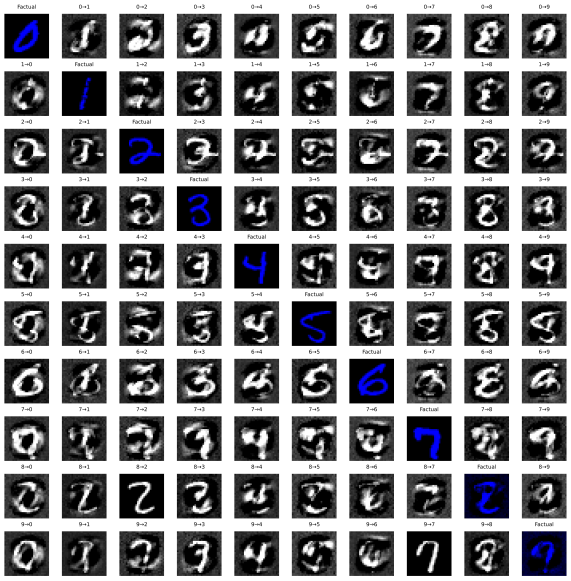

Figure 2: Counterfactual images for *MLP* with counterfactual training. Factual images are shown on the diagonal, with the corresponding counterfactual for each target class (columns) in that same row. The underlying generator, *ECCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

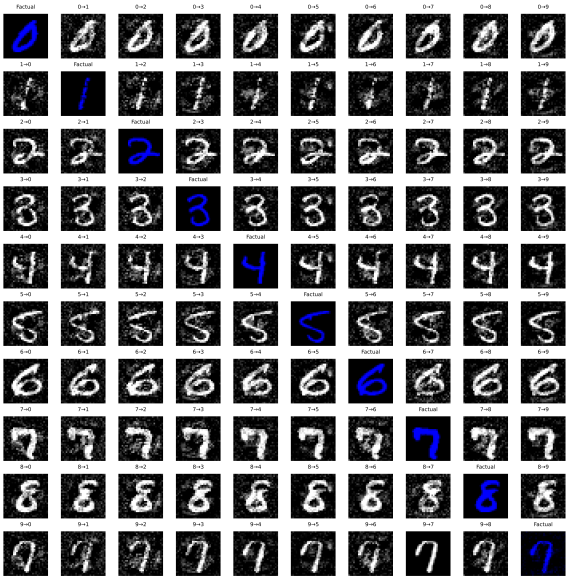

Figure 3: The same setup, factuals, model architecture and generator as in Figure 2, but the model was trained with CT.

Appendix D Grid Searches

To assess the hyperparameter sensitivity of our proposed training regime we ran multiple large grid searches for all of our synthetic datasets. We have grouped these grid searches into multiple categories:

1. **Generator Parameters** (Section D.2): Investigates the effect of changing hyperparameters that affect the counterfactual outcomes during the training phase.
2. **Penalty Strengths** (Section D.3): Investigates the effect of changing the penalty strengths in out proposed objective (Equation 1).
3. **Other Parameters** (Section D.4): Investigates the effect of changing other training parameters, including the total number of generated counterfactuals in each epoch.

We begin by summarizing the high-level findings in Section D.1.2. For each of the categories, Section D.2 to Section D.4 then present all details including the exact parameter grids, average predictive performance outcomes and key evaluation metrics for the generated counterfactuals.

D.1 Evaluation Details

To measure predictive performance, we compute the accuracy and F1-score for all models on test data (Table 3, Table 4, Table 5). With respect to explanatory performance, we report here our findings for the (im)plausibility and cost of counterfactuals at test time. Since the computation of our proposed divergence metric (Equation 5) is memory-intensive, we rely on the distance-based metric for the grid searches. For the counterfactual evaluation, we draw factual samples from the training data for the grid searches to avoid data leakage with respect to our final results reported in the body of the paper. Specifically, we want to avoid choosing our default hyperparameters based on results on the test data. Since we are optimizing for explainability, not predictive performance, we still present test accuracy and F1-scores.

D.1.1 Predictive Performance

We find that CT is associated with little to no decrease in average predictive performance for our synthetic datasets: test accuracy and F1-scores decrease by at most ~1 percentage point, but generally much less (Table 3, Table 4, Table 5). Variation across hyperparameters is negligible as indicated by small standard deviations for these metrics across the board.

D.1.2 Counterfactual Outcomes

Overall, we find that counterfactual training (CT) achieves it key objectives consistently across all hyperparameter settings and also broadly across datasets: plausibility is improved by up to ~60 percent (%) for the *Circles* data (e.g. Figure 4), ~25-30% for the *Moons* data (e.g. Figure 6) and ~10-20% for the *Linearly Separable* data (e.g. Figure 5). At the same time, the average costs of faithful counterfactuals are reduced in many cases by around ~20-25% for *Circles* (e.g. Figure 8) and up to ~50% for *Moons* (e.g. Figure 10). For the *Linearly Separable* data, costs are generally increased although typically by less than 10% (e.g. Figure 9), which reflects a common tradeoff between costs and plausibility (Altmeyer et al. 2024).

We do observe strong sensitivity to certain hyperparameters, with clear an manageable patterns. Concerning generator parameters, we firstly find that using *REVISE* to generate counterfactuals during training typically yields the worst outcomes out of all generators, often leading to a substantial decrease in plausibility. This finding can be attributed to the fact that *REVISE* effectively assigns the task of learning plausible explanations from the model itself to a surrogate VAE. In other words, counterfactuals generated by *REVISE* are less faithful to the model that *ECCo* and *Generic*, and hence we would expect them to be a less effective and, in fact, potentially detrimental role in our training regime. Secondly, we observe that allowing for a higher number of maximum steps T for the counterfactual search generally yields better outcomes. This is intuitive, because it allows more counterfactuals to reach maturity in any given iteration. Looking in particular at the results for *Linearly Separable*, it seems that higher values for T in combination with higher decision thresholds (τ) yields the best results when using *ECCo*. But depending on the degree of class separability of the underlying data, a high decision-threshold can also affect results adversely, as evident from the results for the *Overlapping* data (Figure 7): here we find that CT generally fails to achieve its objective because only a tiny proportion of counterfactuals ever reaches maturity.

Regarding penalty strengths, we find that the strength of the energy regularization, λ_{reg} is a key hyperparameter, while sensitivity with respect to λ_{div} and λ_{adv} is much less evident. In particular, we observe that not regularizing energy enough or at all typically leads to poor performance in terms of decreased plausibility and increased costs, in particular for *Circles* (Figure 12), *Linearly Separable* (Figure 13) and *Overlapping* (Figure 15). High values of λ_{reg} can increase the variability in outcomes, in particular when combined with high values for λ_{div} and λ_{adv}, but this effect is less pronounced.

Finally, concerning other hyperparameters we observe that the effectiveness and stability of CT is positively associated with the number of counterfactuals generated during each training epoch, in particular for *Circles* (Figure 20) and *Moons* (Figure 22). We further find that a higher number of training epochs is beneficial as expected, where we tested training models for 50 and 100 epochs. Interestingly, we find that it is not necessary to employ CT during the entire training phase to achieve the desired improvements in explainability: specifically, we have tested training models conventionally during the first half of training before switching to CT after this initial burn-in period.

D.2 Generator Parameters

The hyperparameter grid with varying generator parameters during training is shown in Note 1. The corresponding evaluation grid used for these experiments is shown in Note 2.

Note 1: Training Phase

- Generator Parameters:
  - Decision Threshold: `0.75, 0.9, 0.95`
  - λ_{egy}: `0.1, 0.5, 5.0, 10.0, 20.0`
  - Maximum Iterations: `5, 25, 50`
- Generator: `ecco, generic, revise`
- Model: `mlp`
- Training Parameters:
  - Objective: `full, vanilla`

Note 2: Evaluation Phase

- Generator Parameters:
  - λ_{egy}: `0.1, 0.5, 1.0, 5.0, 10.0`

D.2.1 Predictive Performance

Predictive performance measures for this grid search are shown in Table 3.

Table 3: Predictive performance measures by dataset and objective averaged across training-phase parameters (Note 1) and evaluation-phase parameters (Note 2).

Dataset	Variable	Objective	Mean	Std
Circ	Accuracy	Full	0.997	0.00309
Circ	Accuracy	Vanilla	0.998	0.000557
Circ	F1-score	Full	0.997	0.00309
Circ	F1-score	Vanilla	0.998	0.000558
LS	Accuracy	Full	0.999	0.00201
LS	Accuracy	Vanilla	1	0
LS	F1-score	Full	0.999	0.00201
LS	F1-score	Vanilla	1	0
Moon	Accuracy	Full	0.999	0.000696
Moon	Accuracy	Vanilla	1	0.00111
Moon	F1-score	Full	0.999	0.000696
Moon	F1-score	Vanilla	1	0.00111
OL	Accuracy	Full	0.915	0.00477
OL	Accuracy	Vanilla	0.917	0.00123
OL	F1-score	Full	0.915	0.00478
OL	F1-score	Vanilla	0.917	0.00124

D.2.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 4 to Figure 7.

D.2.3 Cost

The results with respect to the cost measure are shown in Figure 8 to Figure 11.

D.3 Penalty Strengths

The hyperparameter grid with varying penalty strengths during training is shown in Note 3. The corresponding evaluation grid used for these experiments is shown in Note 4.

Note 3: Training Phase

- Generator: `ecco, generic, revise`
- Model: `mlp`
- Training Parameters:
  - λ_{adv}: `0.1, 0.25, 1.0`
  - λ_{div}: `0.01, 0.1, 1.0`
  - λ_{reg}: `0.0, 0.01, 0.1, 0.25, 0.5`
  - Objective: `full, vanilla`

Note 4: Evaluation Phase

- Generator Parameters:
  - λ_{egy}: `0.1, 0.5, 1.0, 5.0, 10.0`

D.3.1 Predictive Performance

Predictive performance measures for this grid search are shown in Table 4.

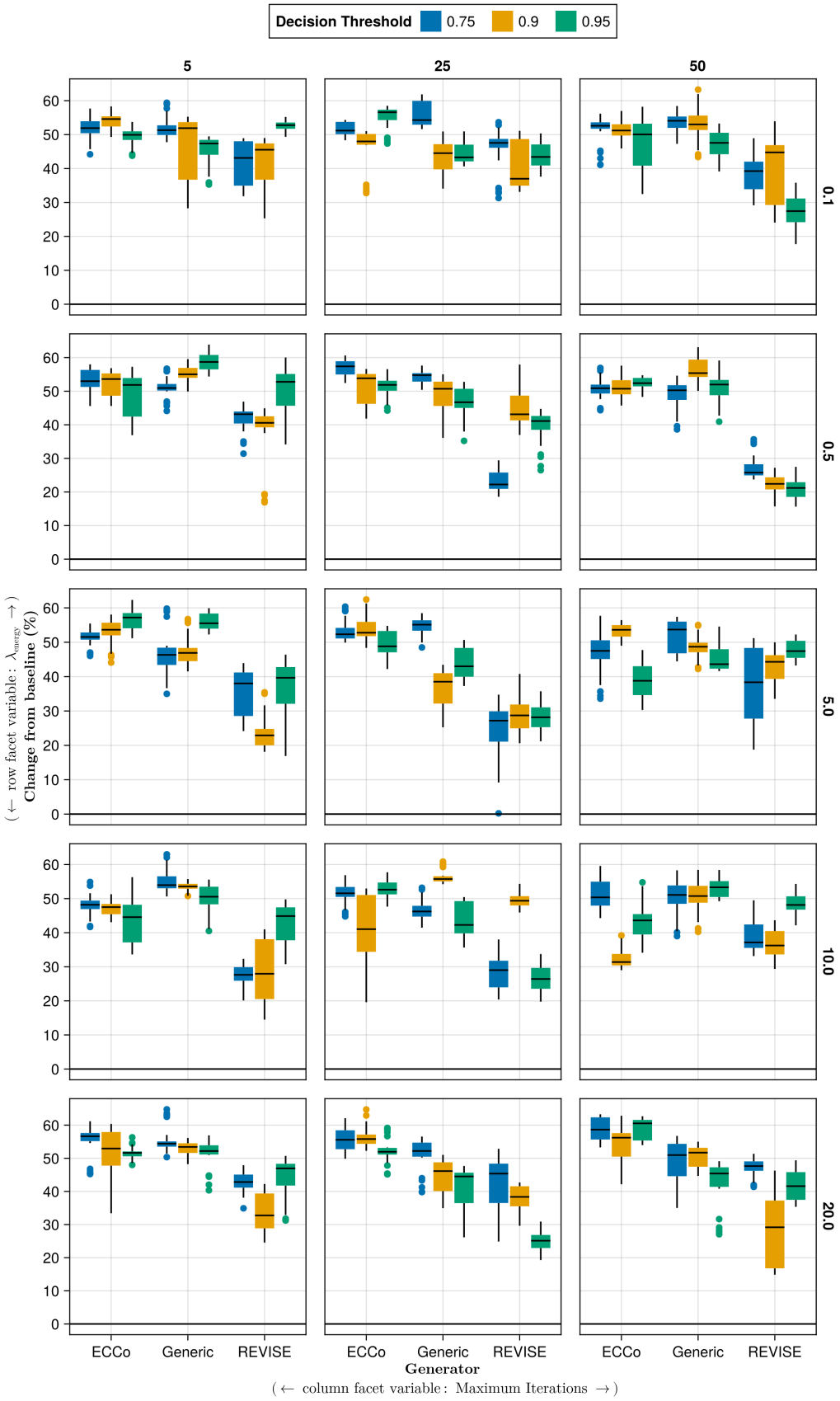

Figure 4: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

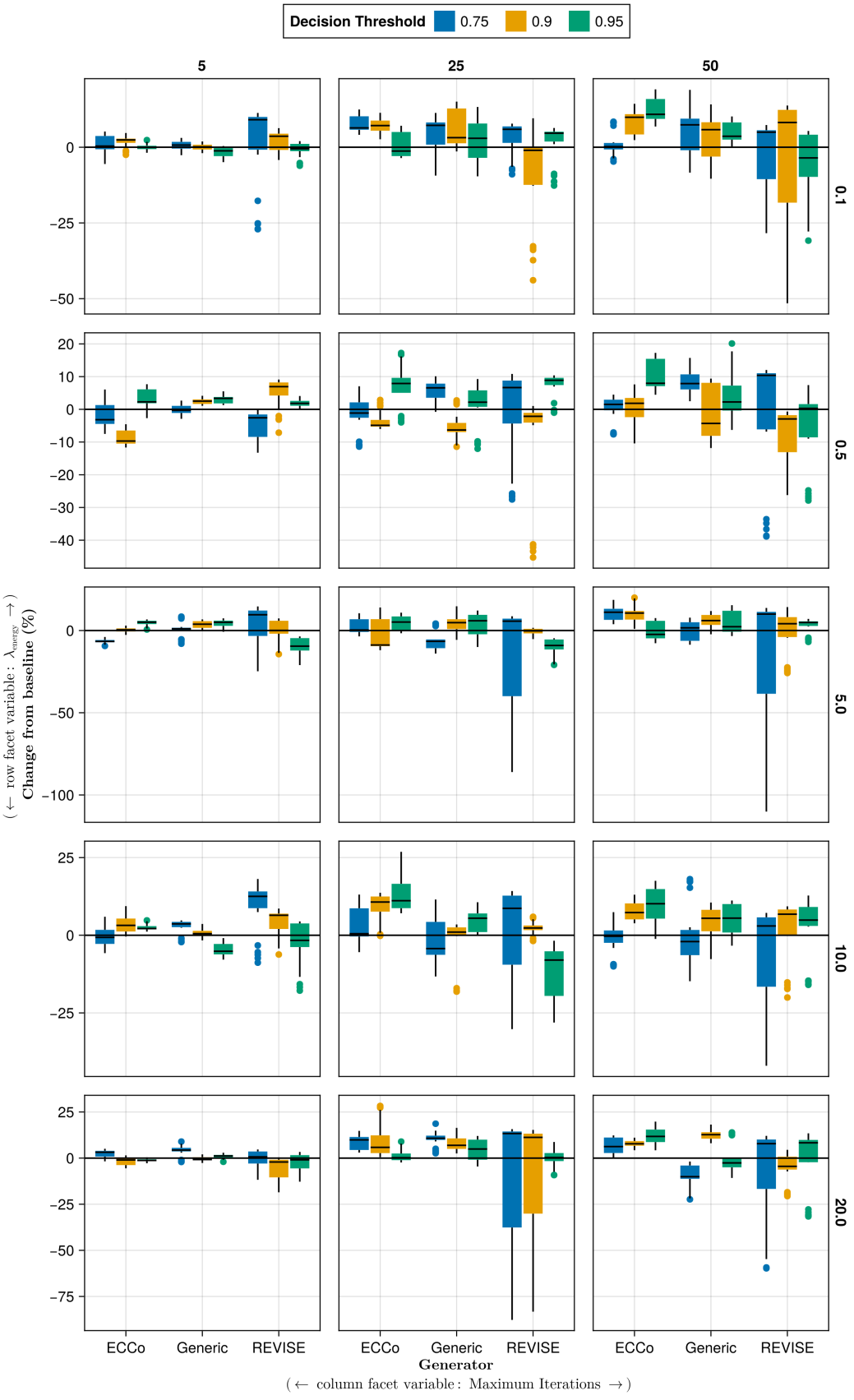

Figure 5: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

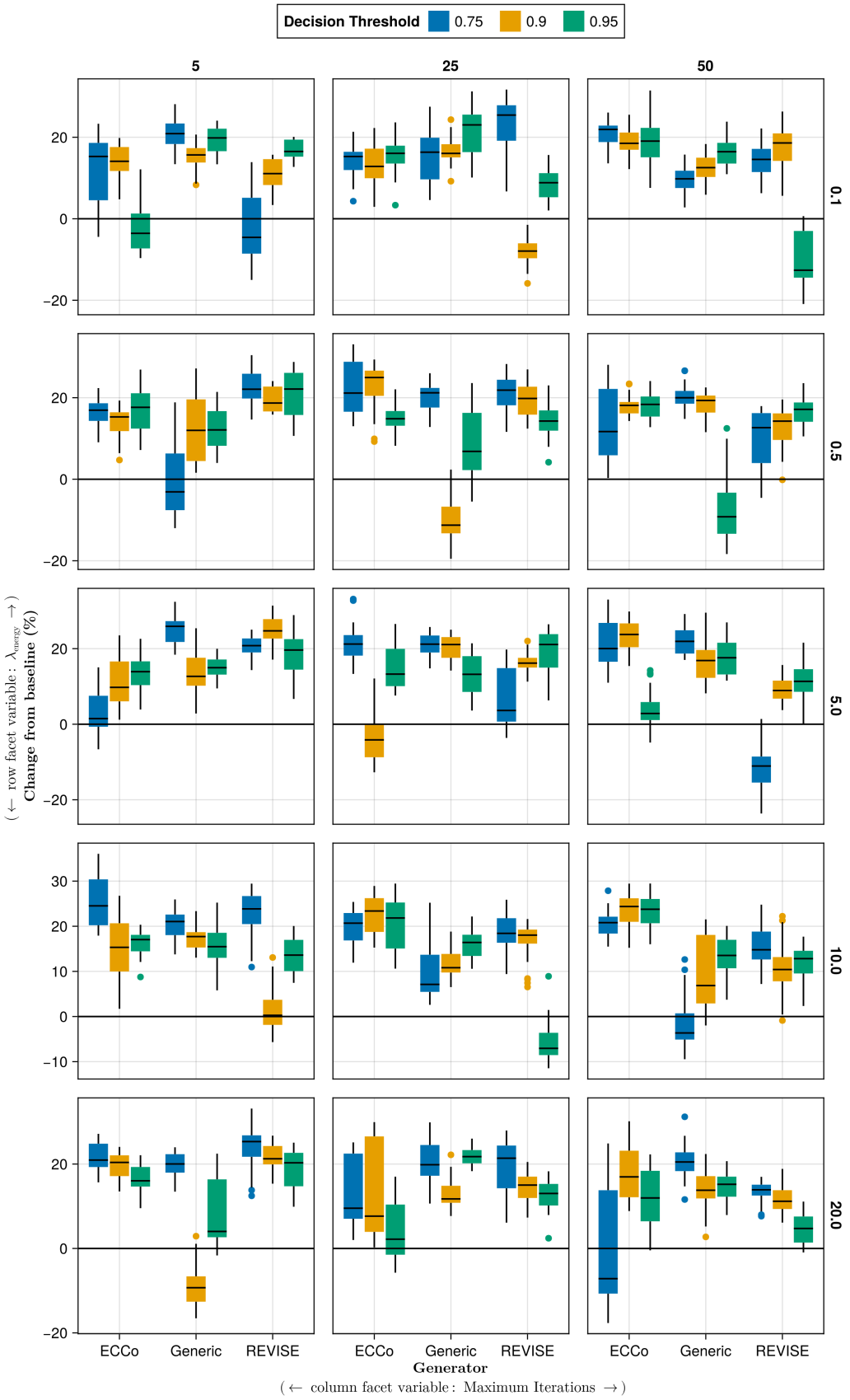

Figure 6: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

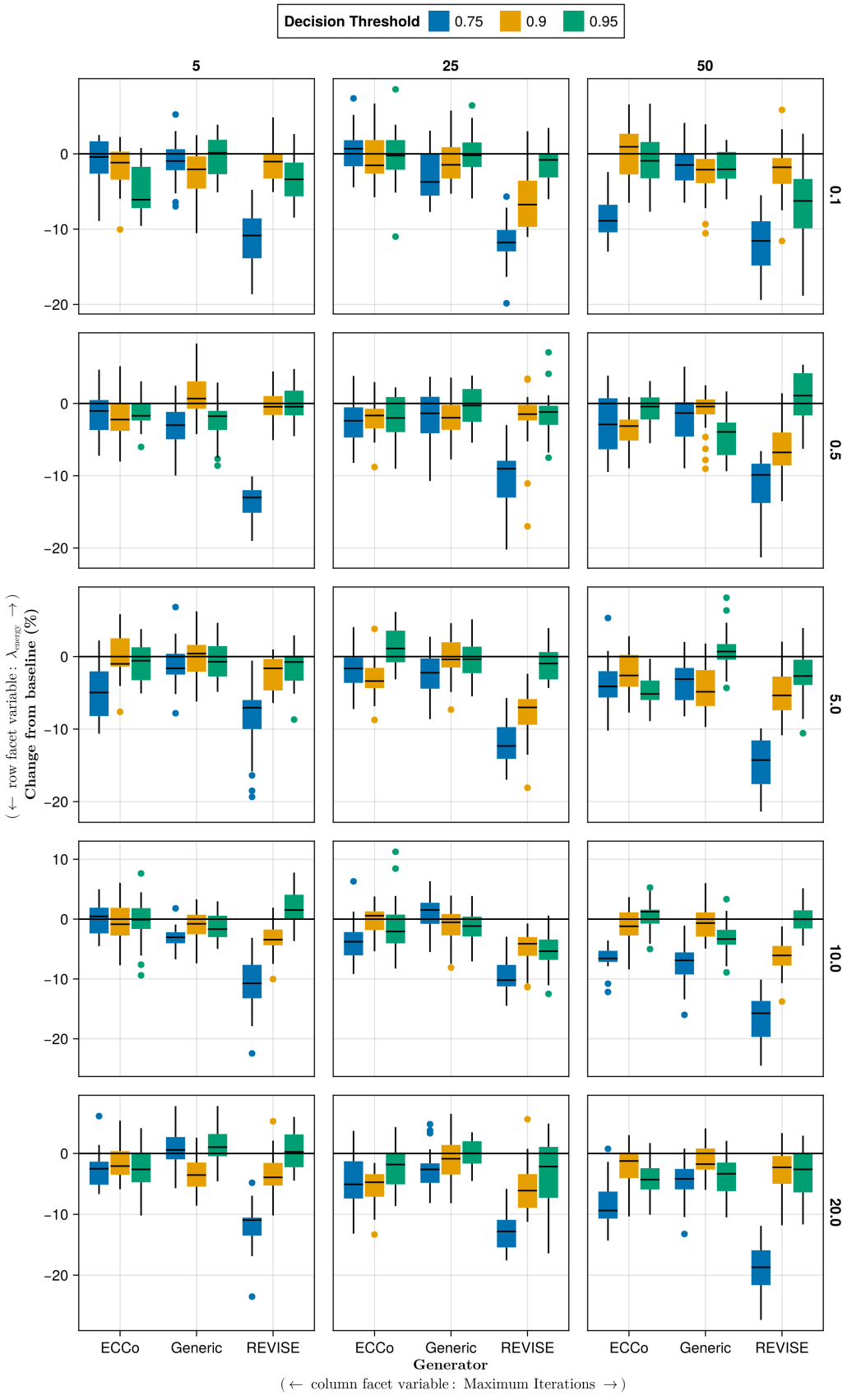

Figure 7: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

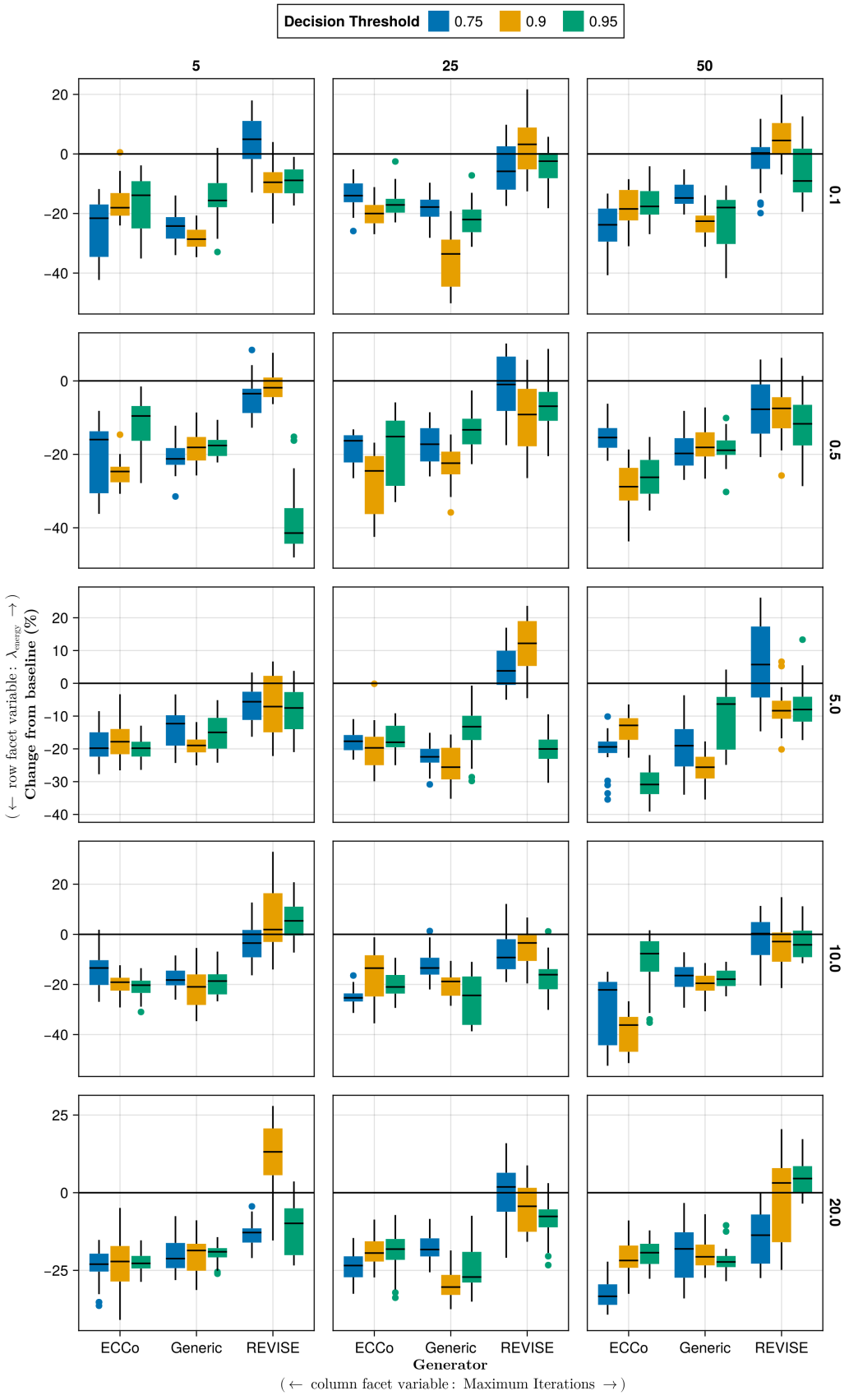

Figure 8: Average outcomes for the cost measure across hyperparameters. Data: Circles.

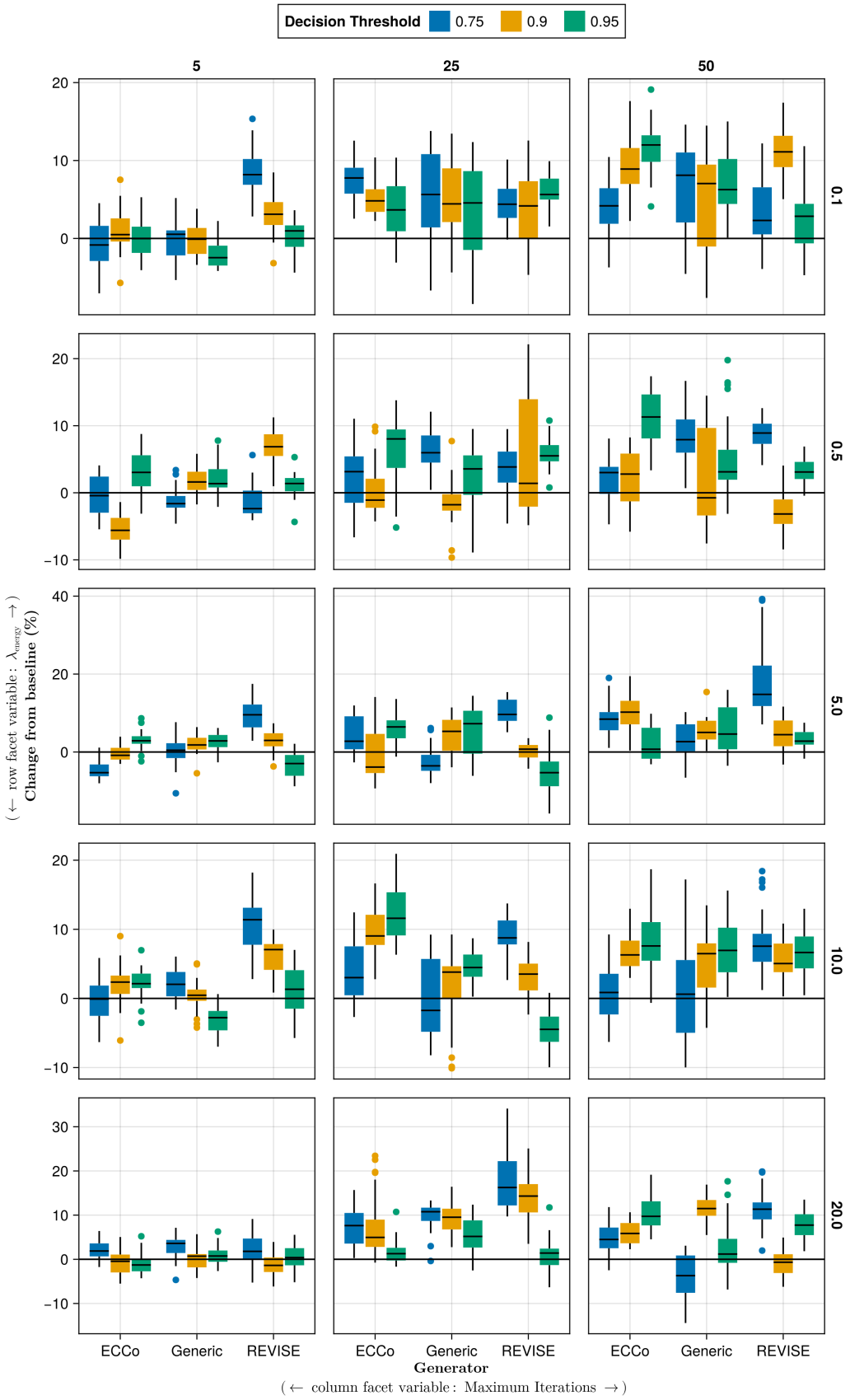

Figure 9: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

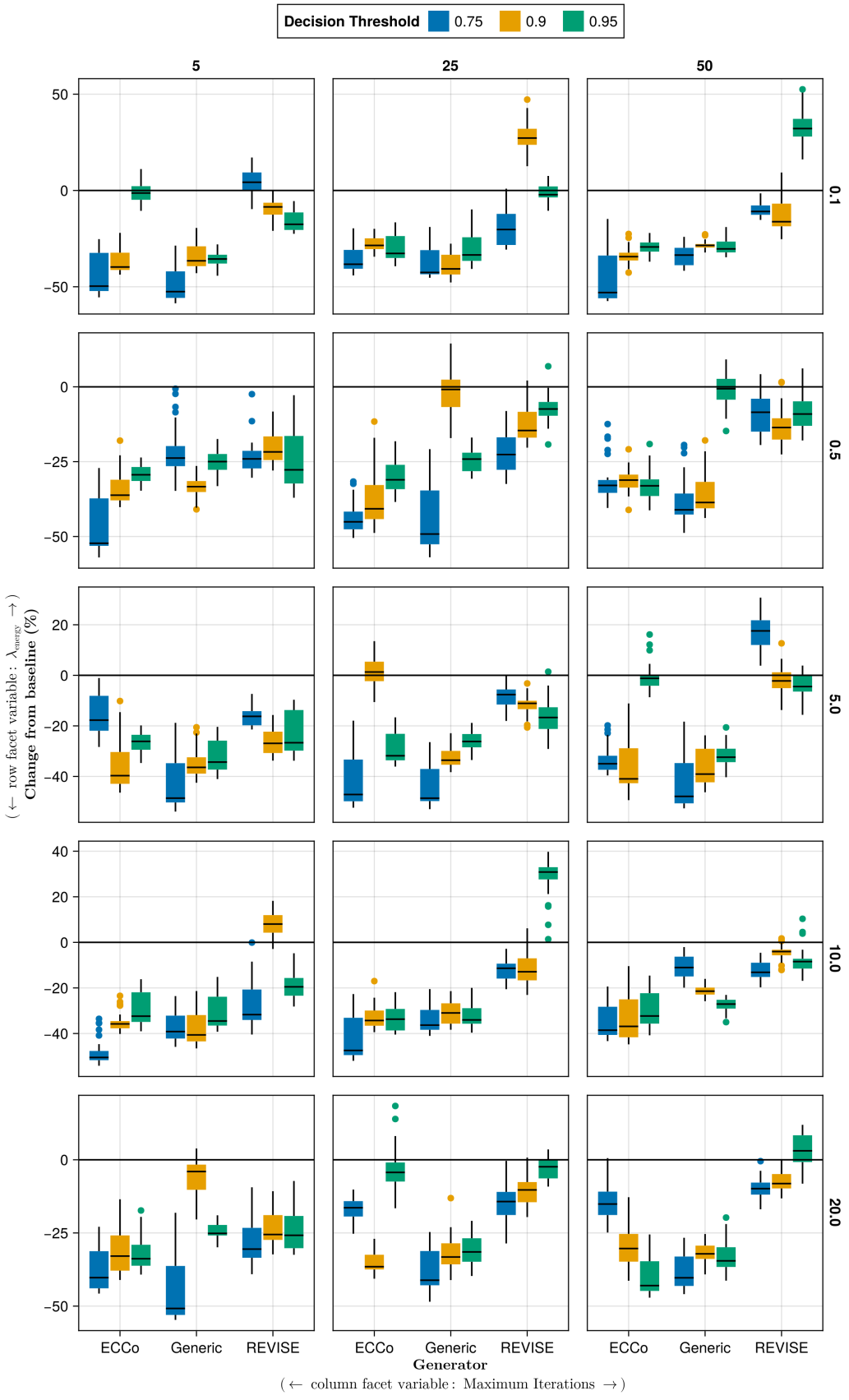

Figure 10: Average outcomes for the cost measure across hyperparameters. Data: Moons.

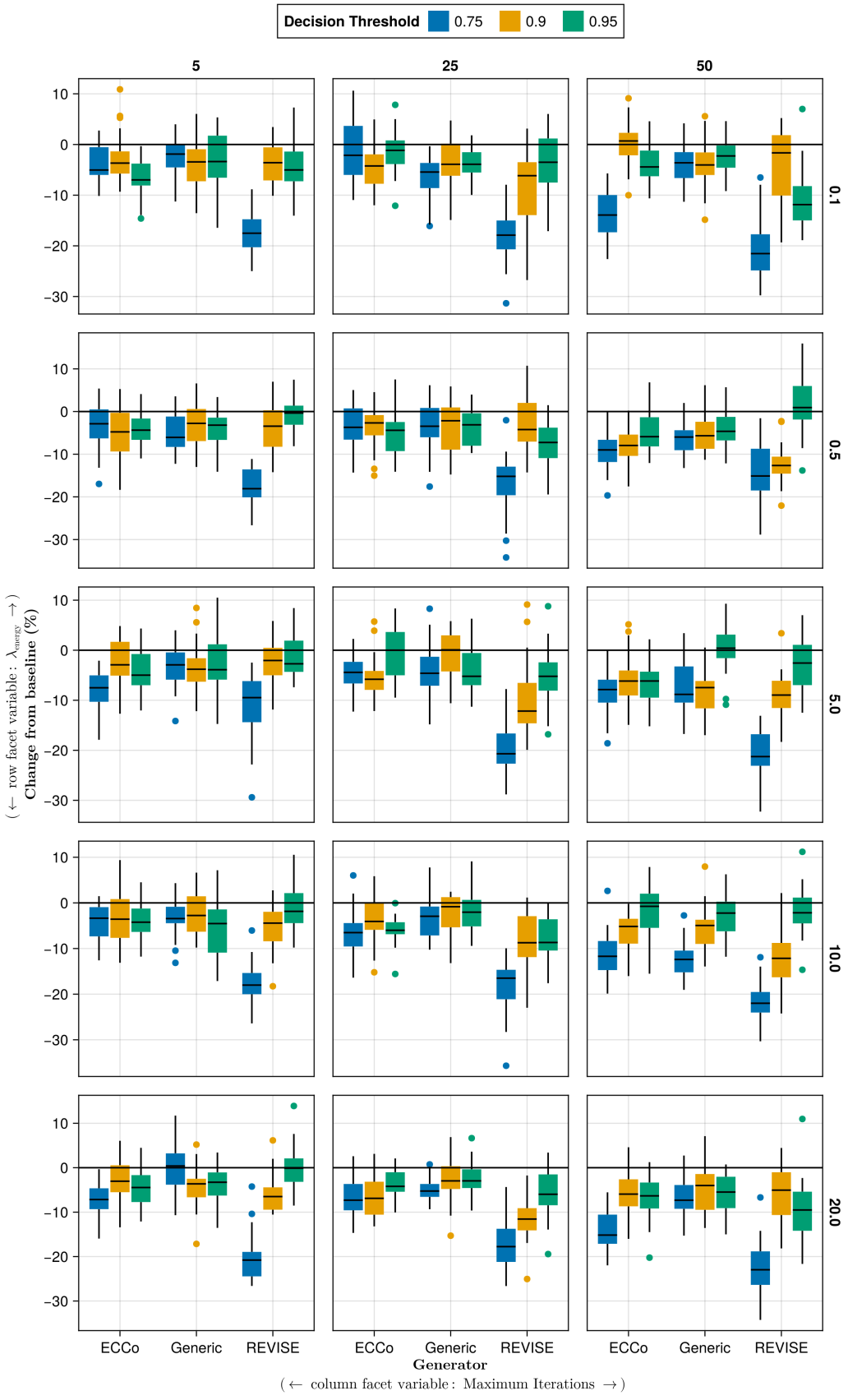

Figure 11: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.

Table 4: Predictive performance measures by dataset and objective averaged across training-phase parameters (Note 3) and evaluation-phase parameters (Note 4).

Dataset	Variable	Objective	Mean	Std
Circ	Accuracy	Full	0.994	0.0144
Circ	Accuracy	Vanilla	0.998	0.000875
Circ	F1-score	Full	0.994	0.0145
Circ	F1-score	Vanilla	0.998	0.000875
LS	Accuracy	Full	0.998	0.00772
LS	Accuracy	Vanilla	1	0
LS	F1-score	Full	0.998	0.00773
LS	F1-score	Vanilla	1	0
Moon	Accuracy	Full	0.987	0.0351
Moon	Accuracy	Vanilla	0.998	0.0101
Moon	F1-score	Full	0.987	0.0352
Moon	F1-score	Vanilla	0.998	0.0102
OL	Accuracy	Full	0.911	0.0217
OL	Accuracy	Vanilla	0.916	0.00236
OL	F1-score	Full	0.911	0.0219
OL	F1-score	Vanilla	0.916	0.00236

D.3.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 12 to Figure 15.

D.3.3 Cost

The results with respect to the cost measure are shown in Figure 16 to Figure 19.

D.4 Other Parameters

The hyperparameter grid with other varying training parameters is shown in Note 5. The corresponding evaluation grid used for these experiments is shown in Note 6.

Note 5: Training Phase

- Generator: `ecco, generic, revise`
- Model: `mlp`
- Training Parameters:
  - Burnin: `0.0, 0.5`
  - No. Counterfactuals: `100, 1000`
  - No. Epochs: `50, 100`
  - Objective: `full, vanilla`

Note 6: Evaluation Phase

- Generator Parameters:
  - λ_{egy}: `0.1, 0.5, 1.0, 5.0, 10.0`

D.4.1 Predictive Performance

Predictive performance measures for this grid search are shown in Table 5.

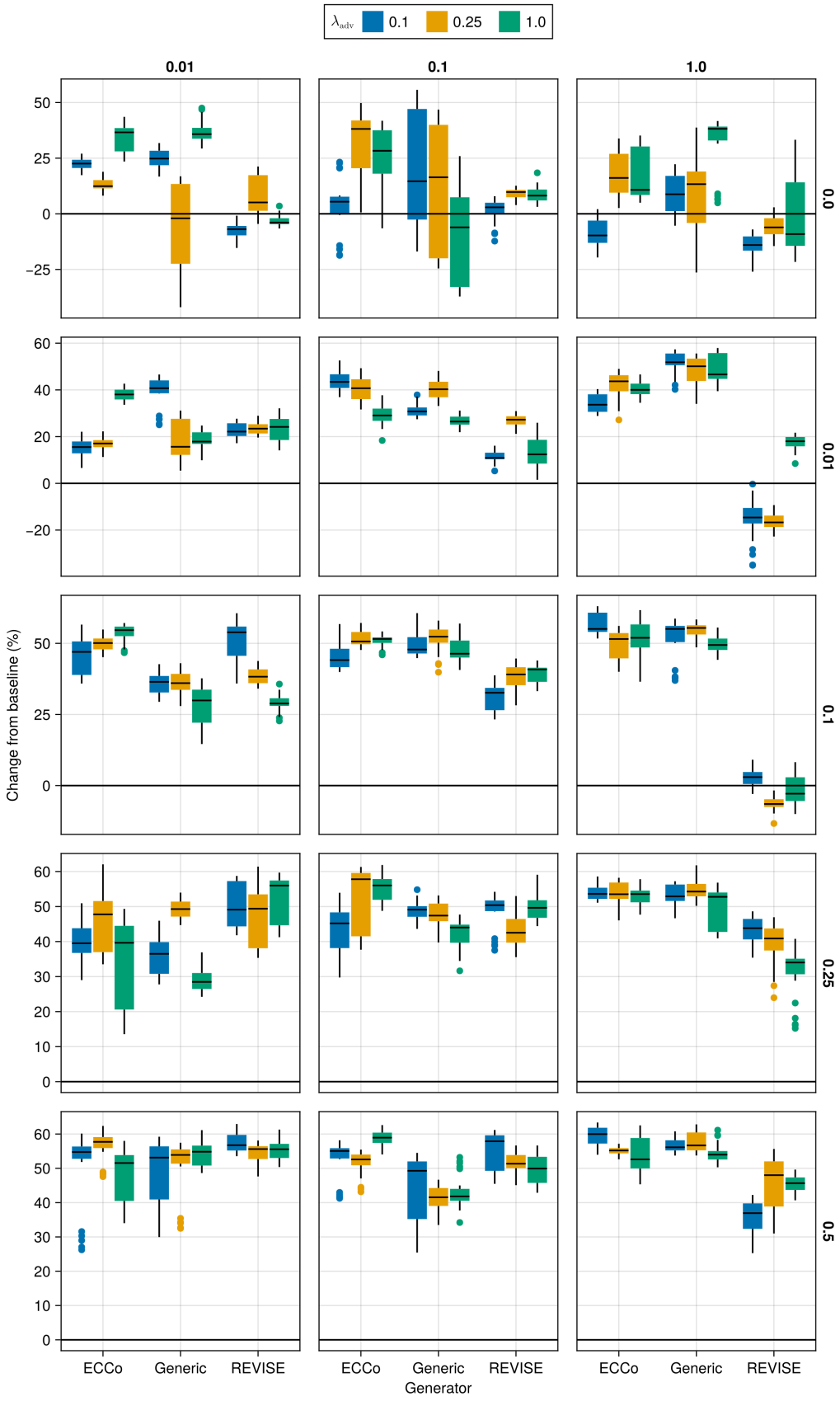

Figure 12: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

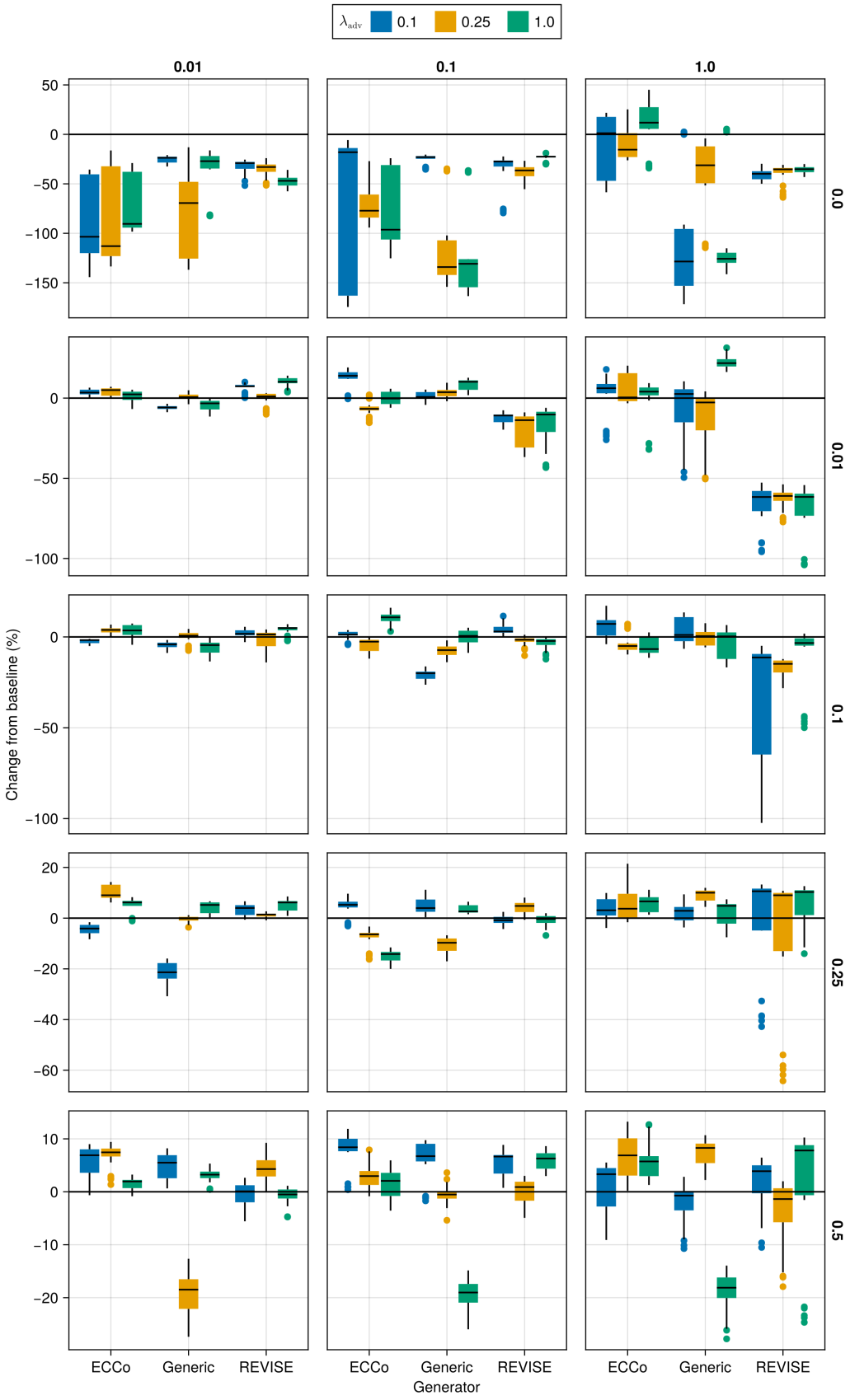

Figure 13: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

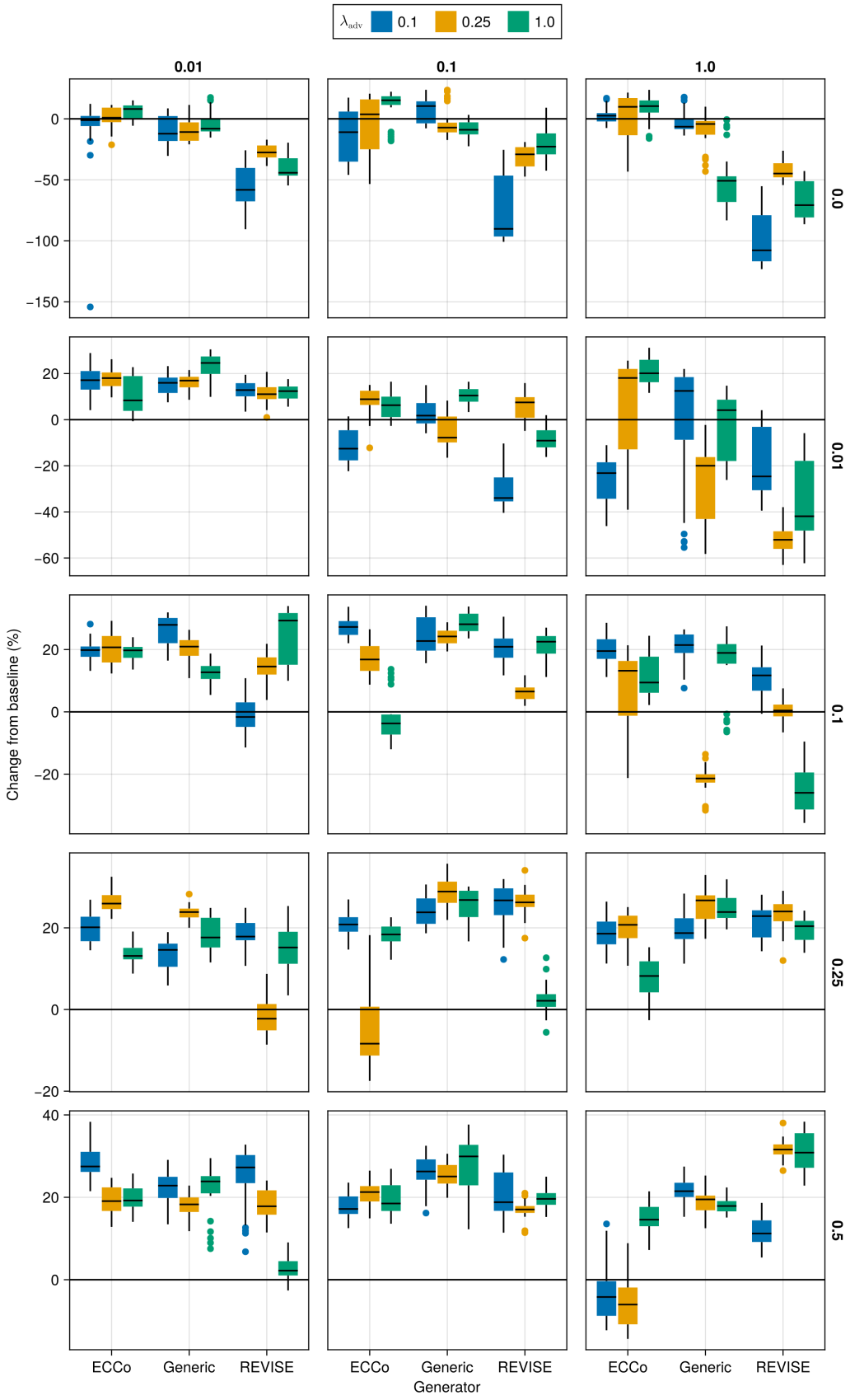

Figure 14: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

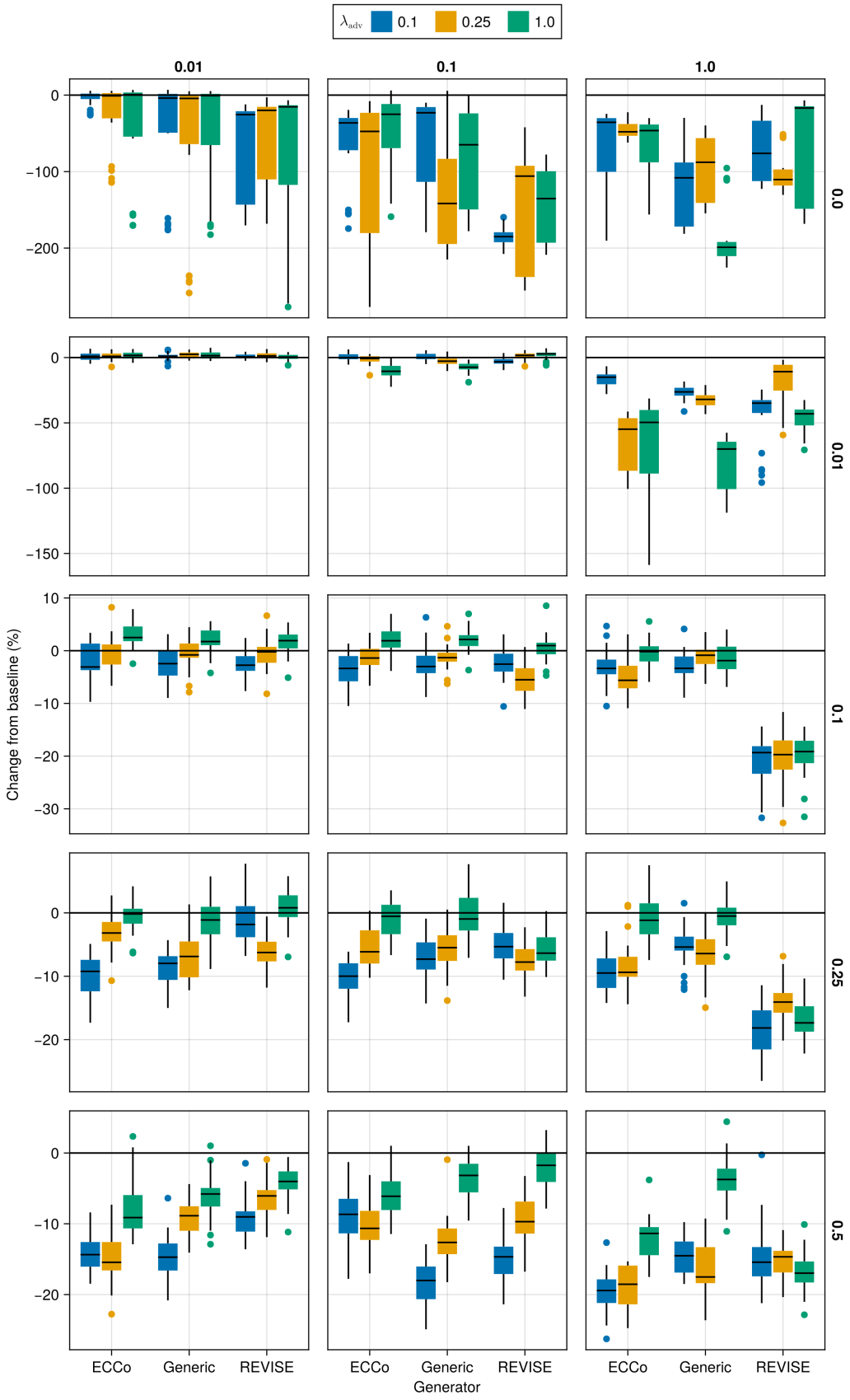

Figure 15: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

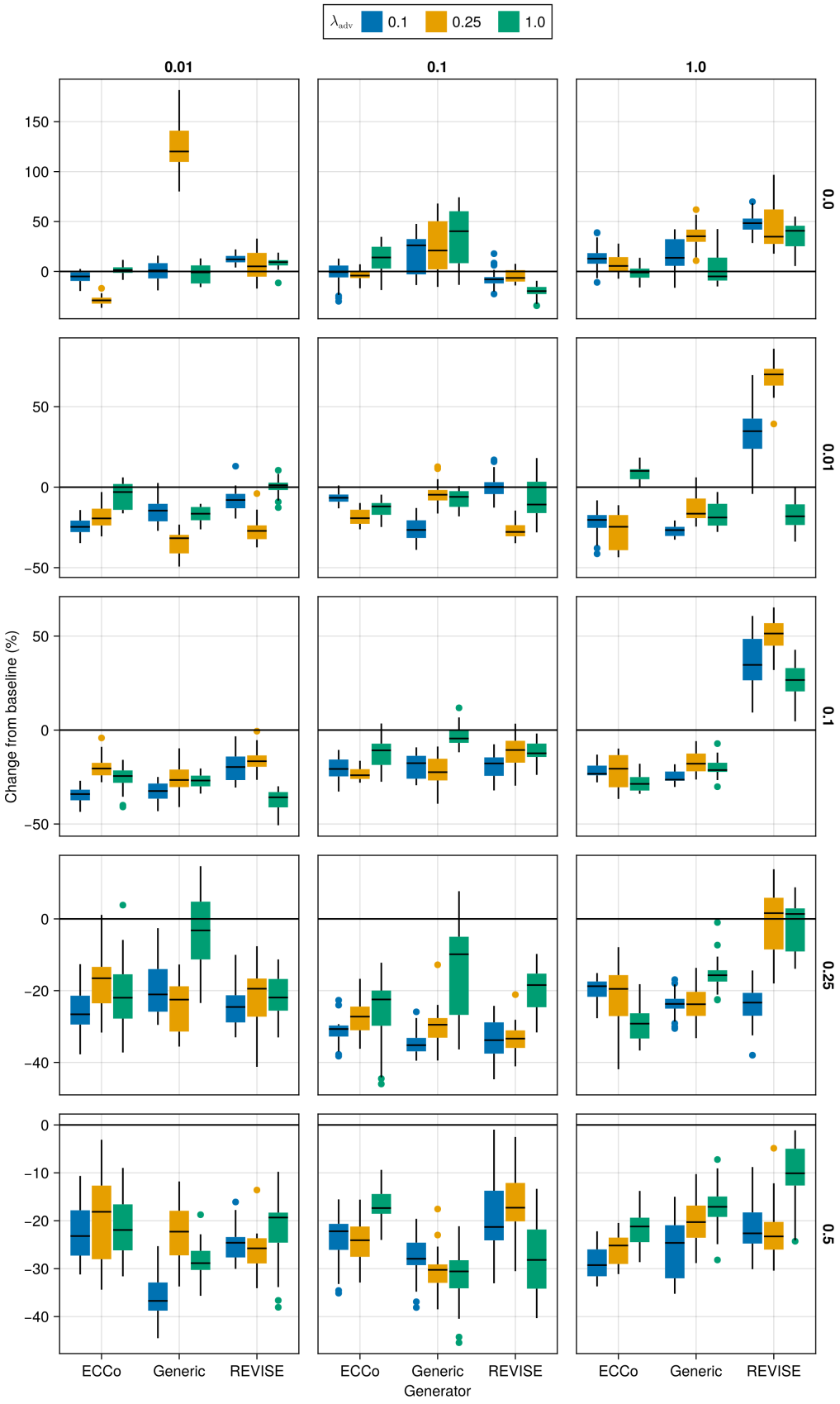

Figure 16: Average outcomes for the cost measure across hyperparameters. Data: Circles.

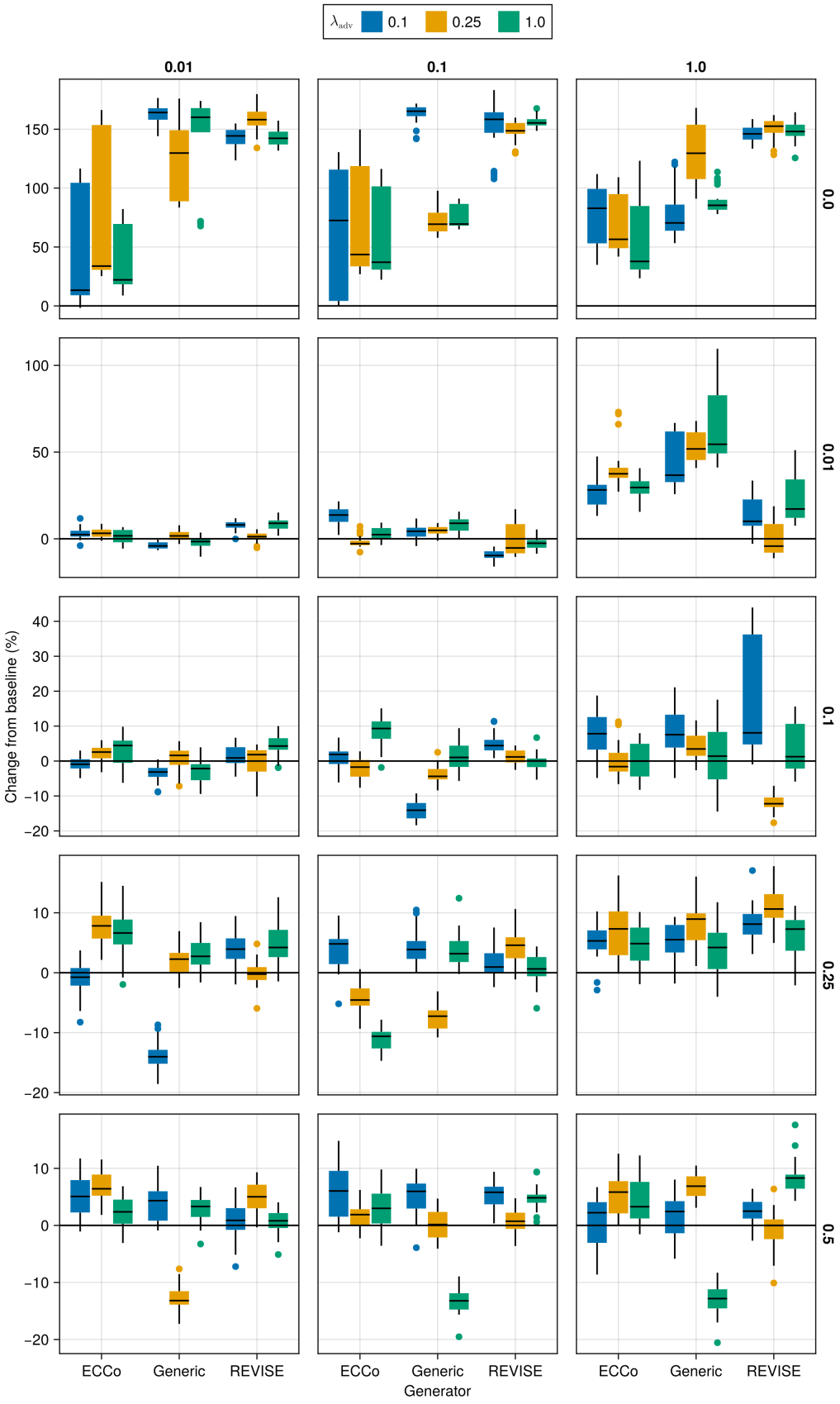

Figure 17: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

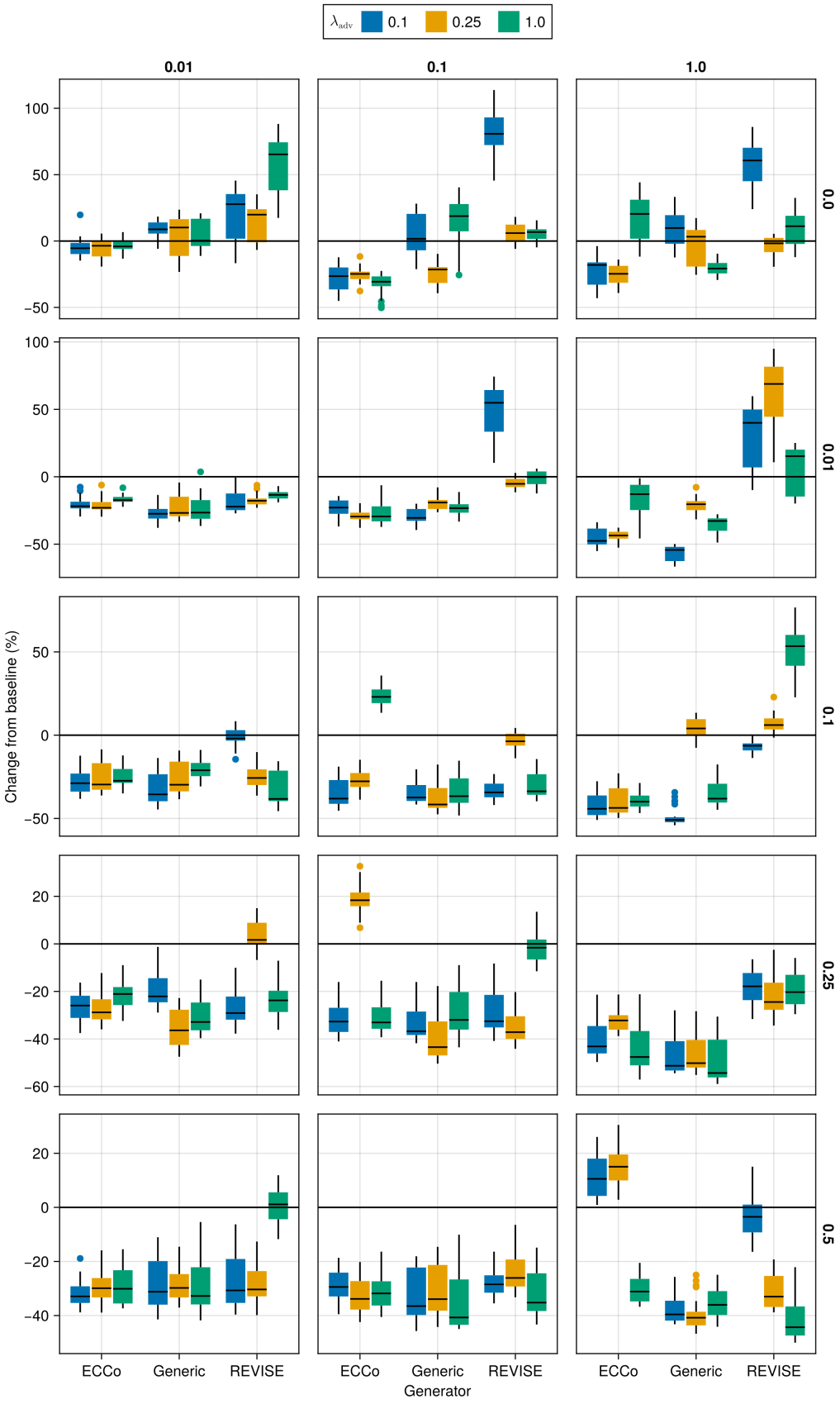

Figure 18: Average outcomes for the cost measure across hyperparameters. Data: Moons.

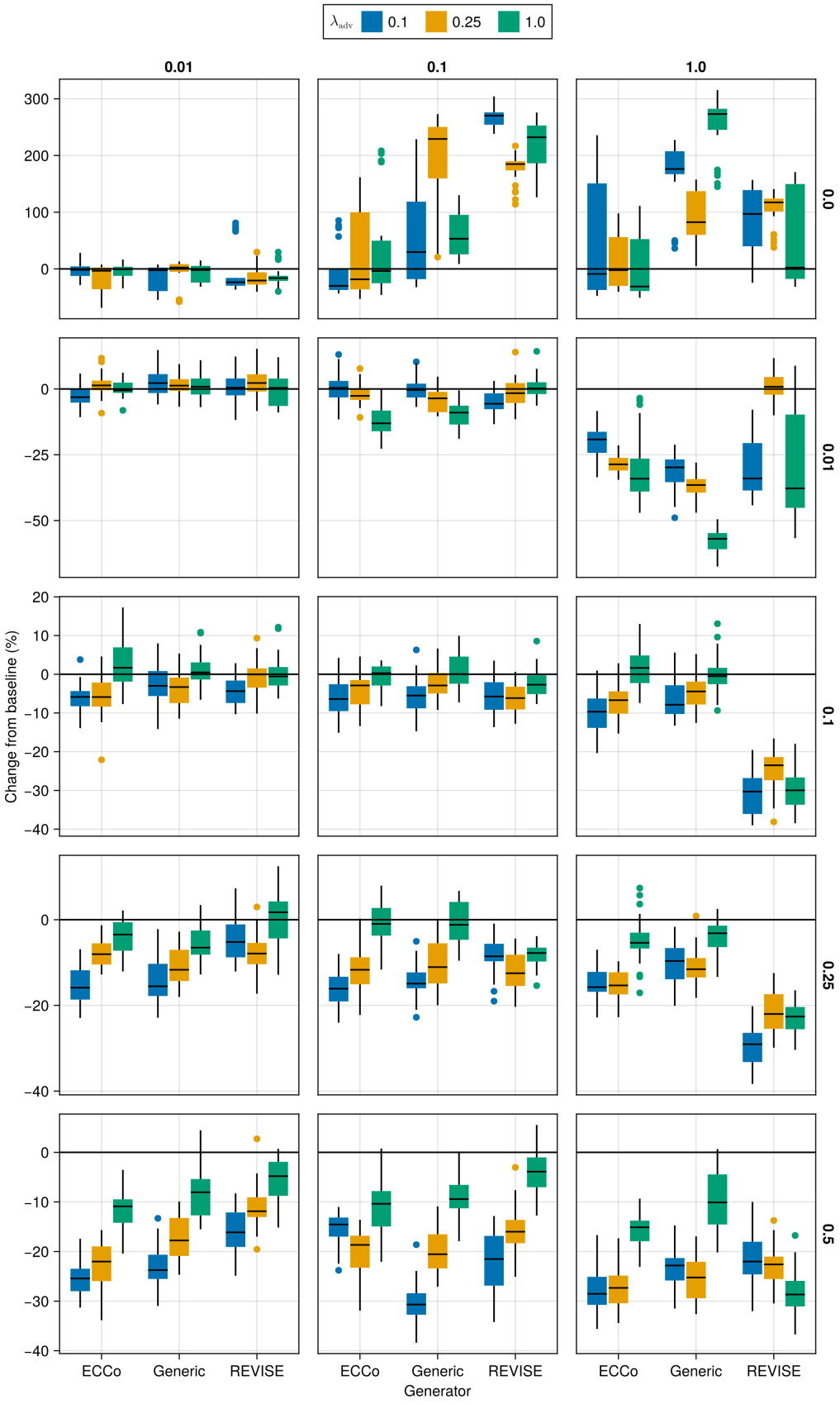

Figure 19: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.

Table 5: Predictive performance measures by dataset and objective averaged across training-phase parameters (Note 5) and evaluation-phase parameters (Note 6).

Dataset	Variable	Objective	Mean	Std
Circ	Accuracy	Full	0.995	0.00431
Circ	Accuracy	Vanilla	0.998	0.000566
Circ	F1-score	Full	0.995	0.00432
Circ	F1-score	Vanilla	0.998	0.000566
LS	Accuracy	Full	0.999	0.00231
LS	Accuracy	Vanilla	1	0
LS	F1-score	Full	0.999	0.00231
LS	F1-score	Vanilla	1	0
Moon	Accuracy	Full	0.996	0.0136
Moon	Accuracy	Vanilla	0.988	0.022
Moon	F1-score	Full	0.996	0.0136
Moon	F1-score	Vanilla	0.988	0.022
OL	Accuracy	Full	0.914	0.00563
OL	Accuracy	Vanilla	0.918	0.00116
OL	F1-score	Full	0.914	0.0057
OL	F1-score	Vanilla	0.918	0.00116

D.4.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 20 to Figure 23.

D.4.3 Cost

The results with respect to the cost measure are shown in Figure 24 to Figure 27.

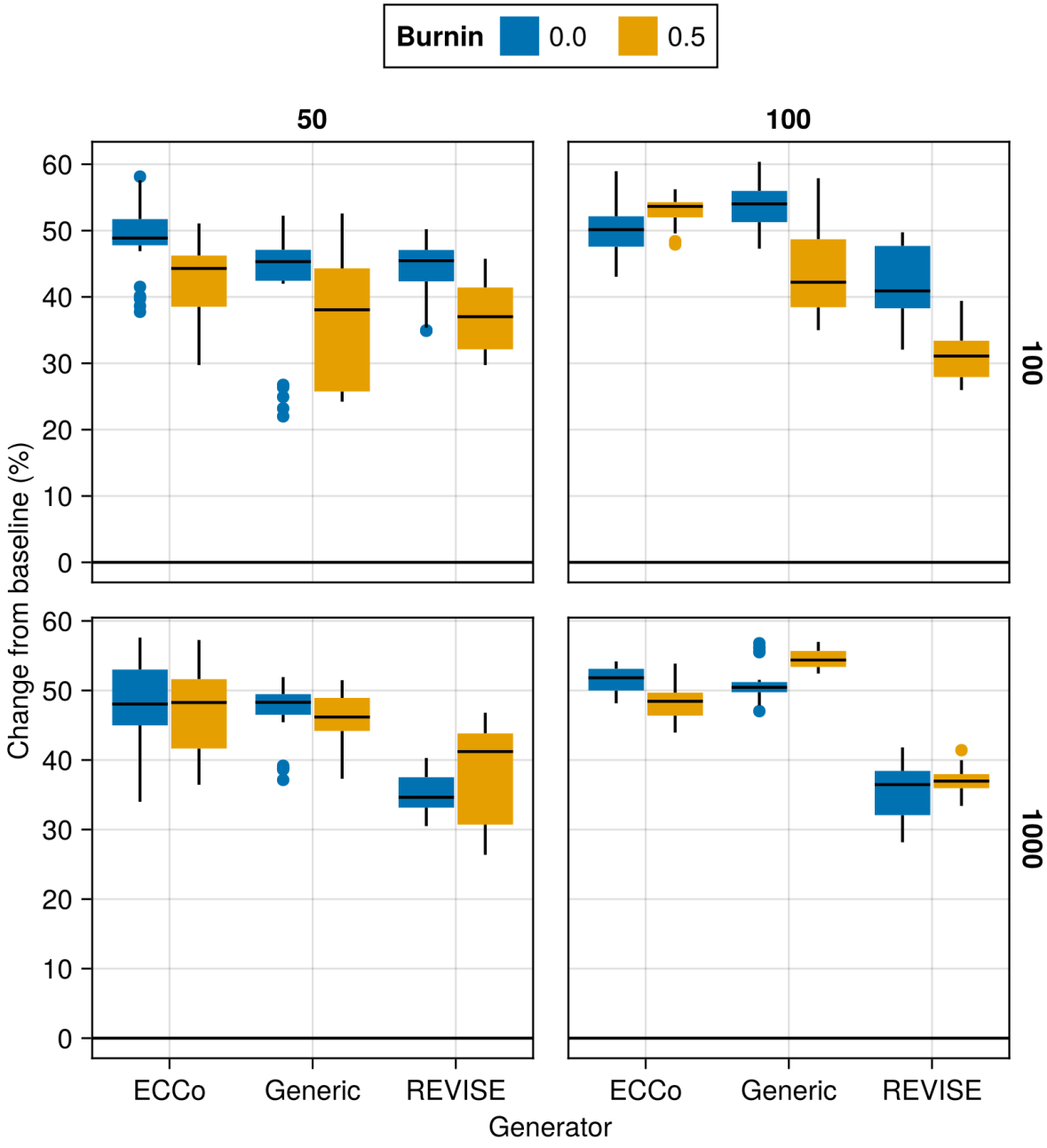

Figure 20: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

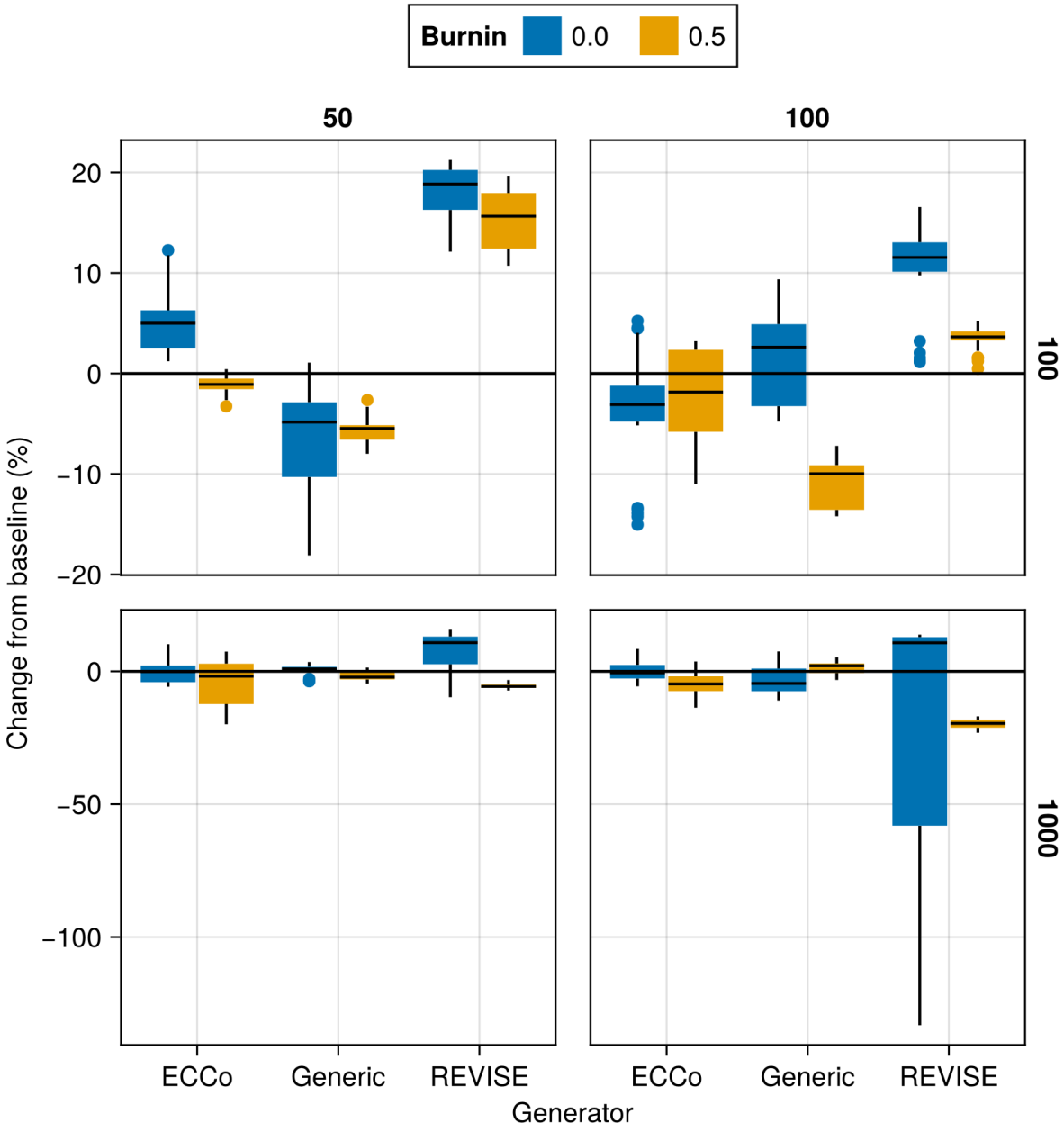

Figure 21: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

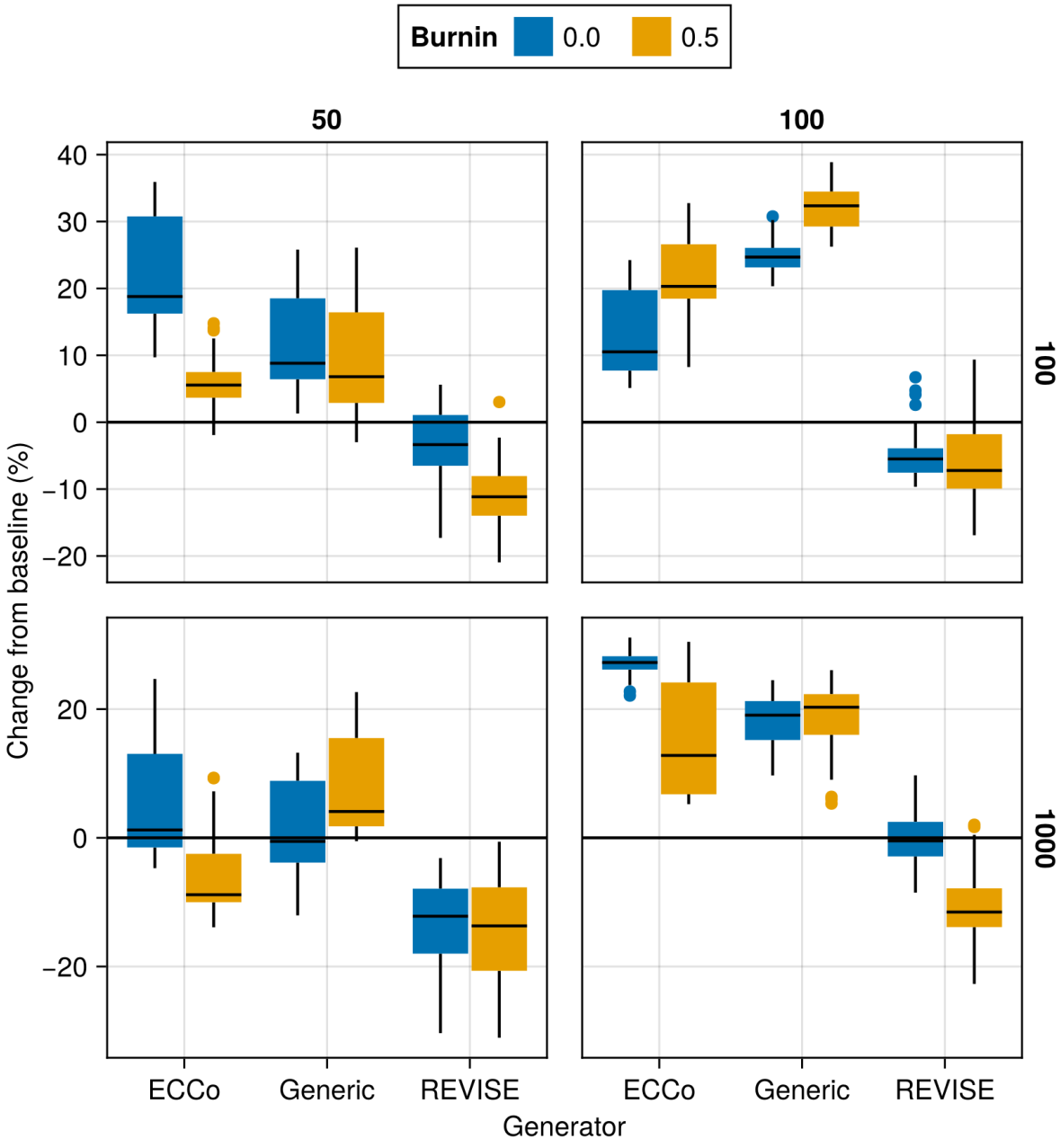

Figure 22: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

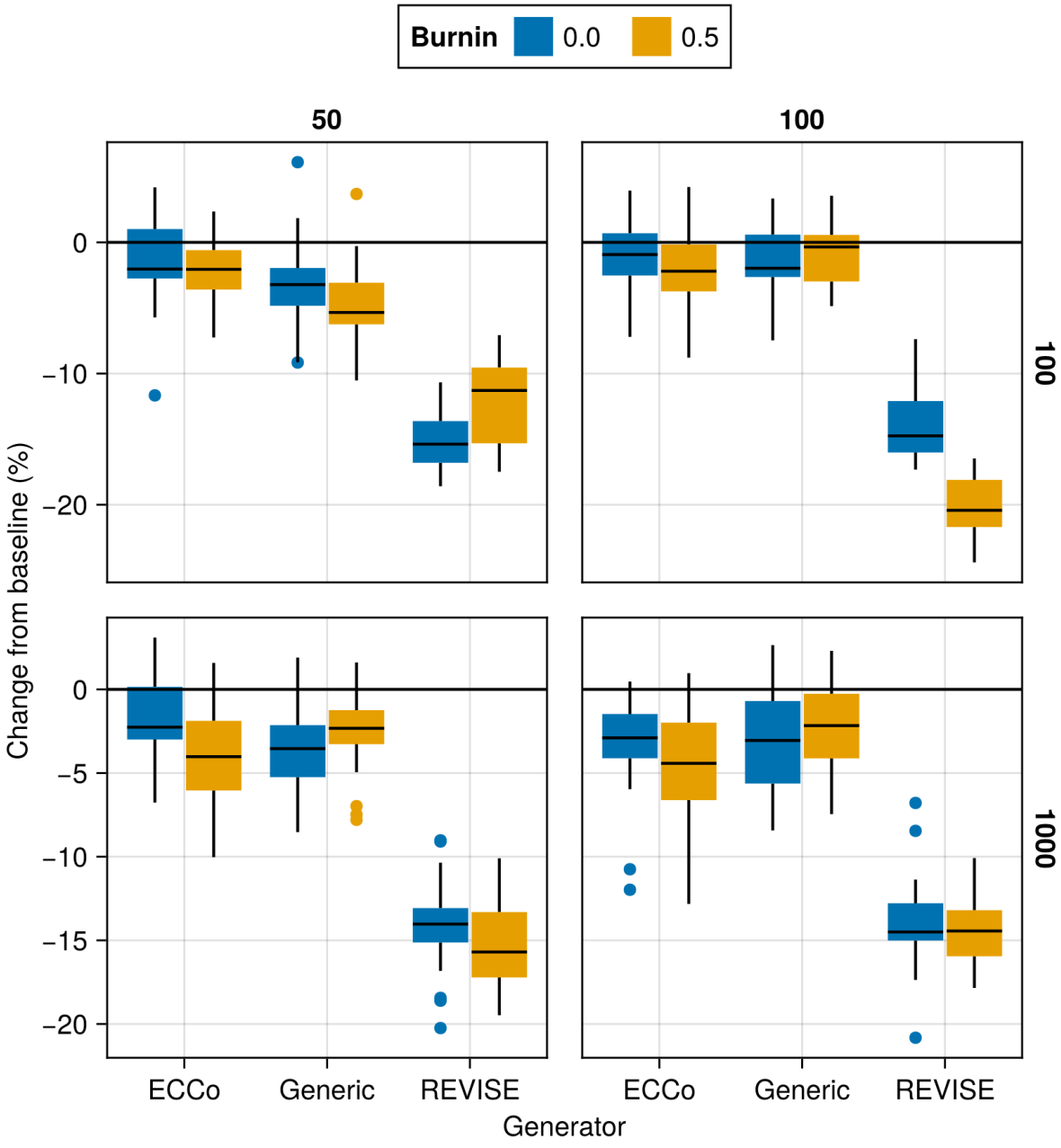

Figure 23: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

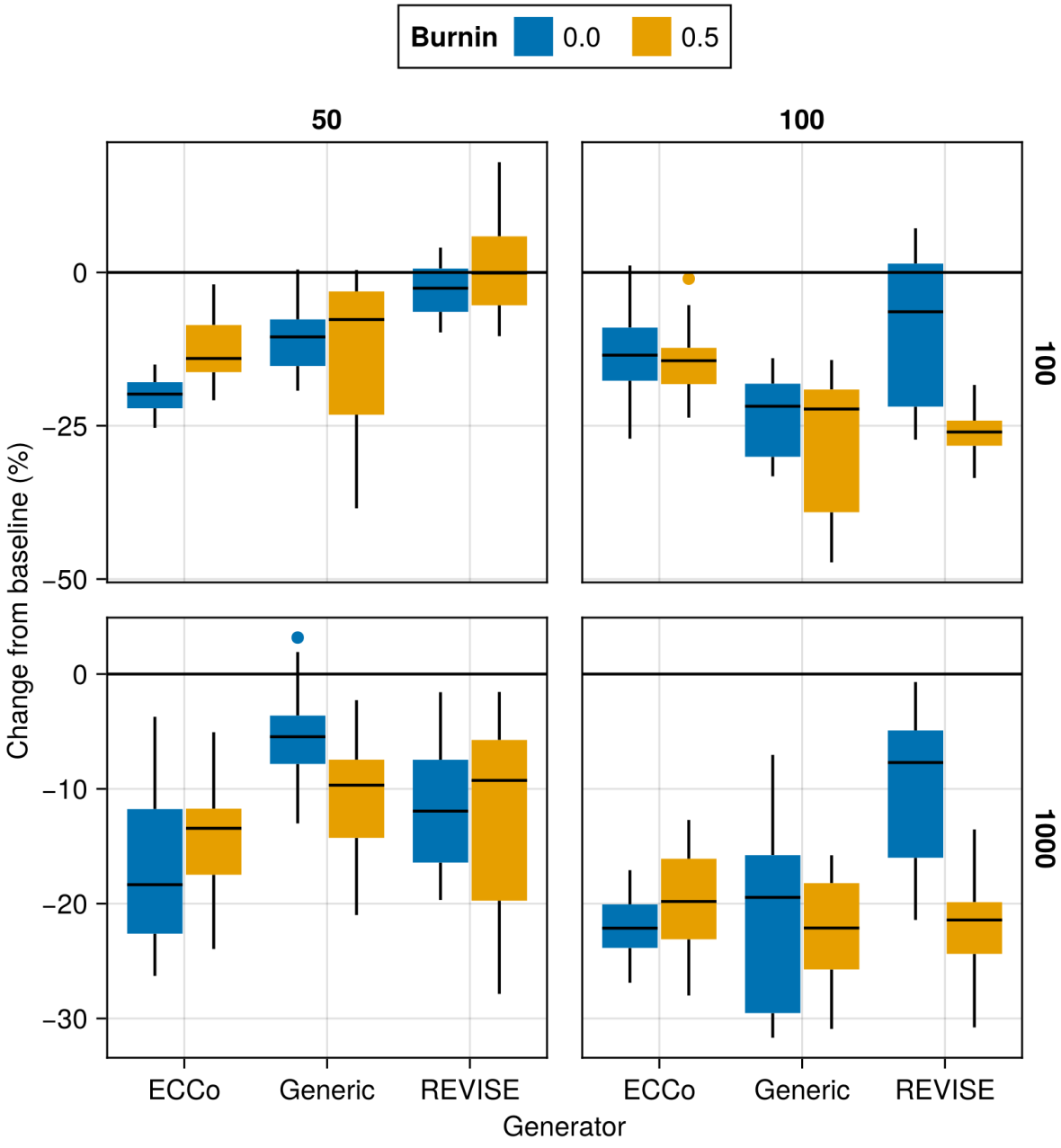

Figure 24: Average outcomes for the cost measure across hyperparameters. Data: Circles.

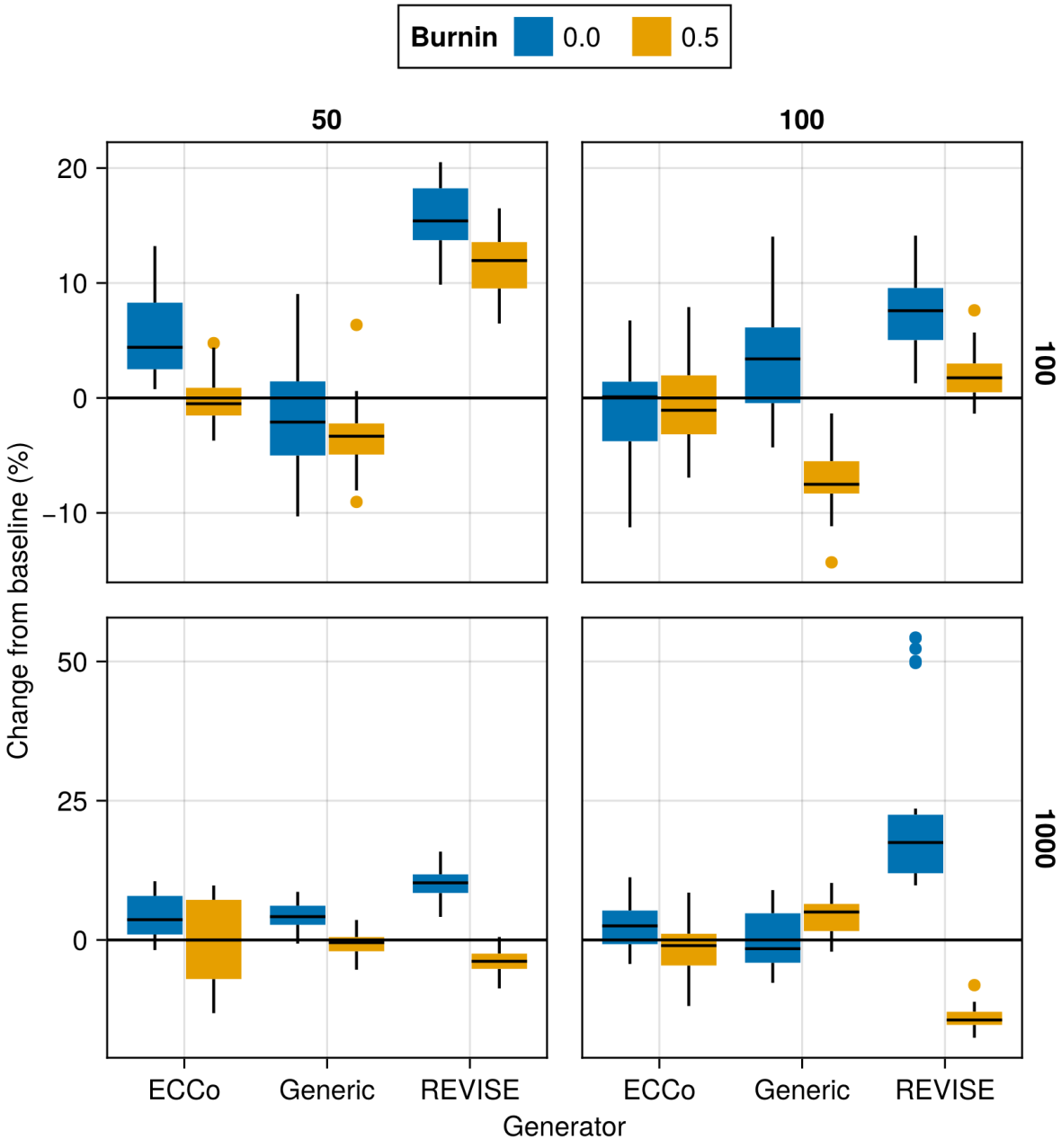

Figure 25: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

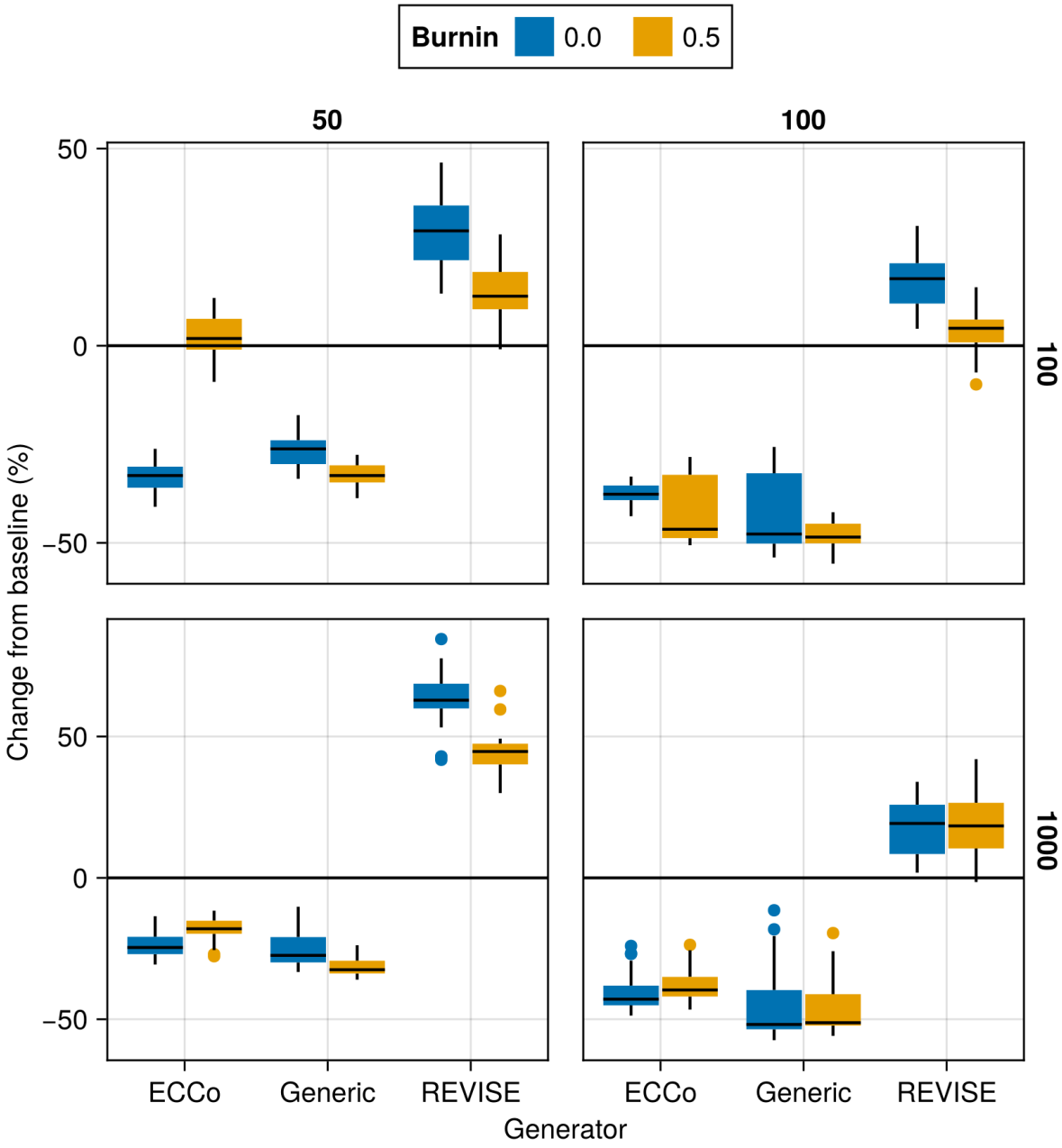

Figure 26: Average outcomes for the cost measure across hyperparameters. Data: Moons.

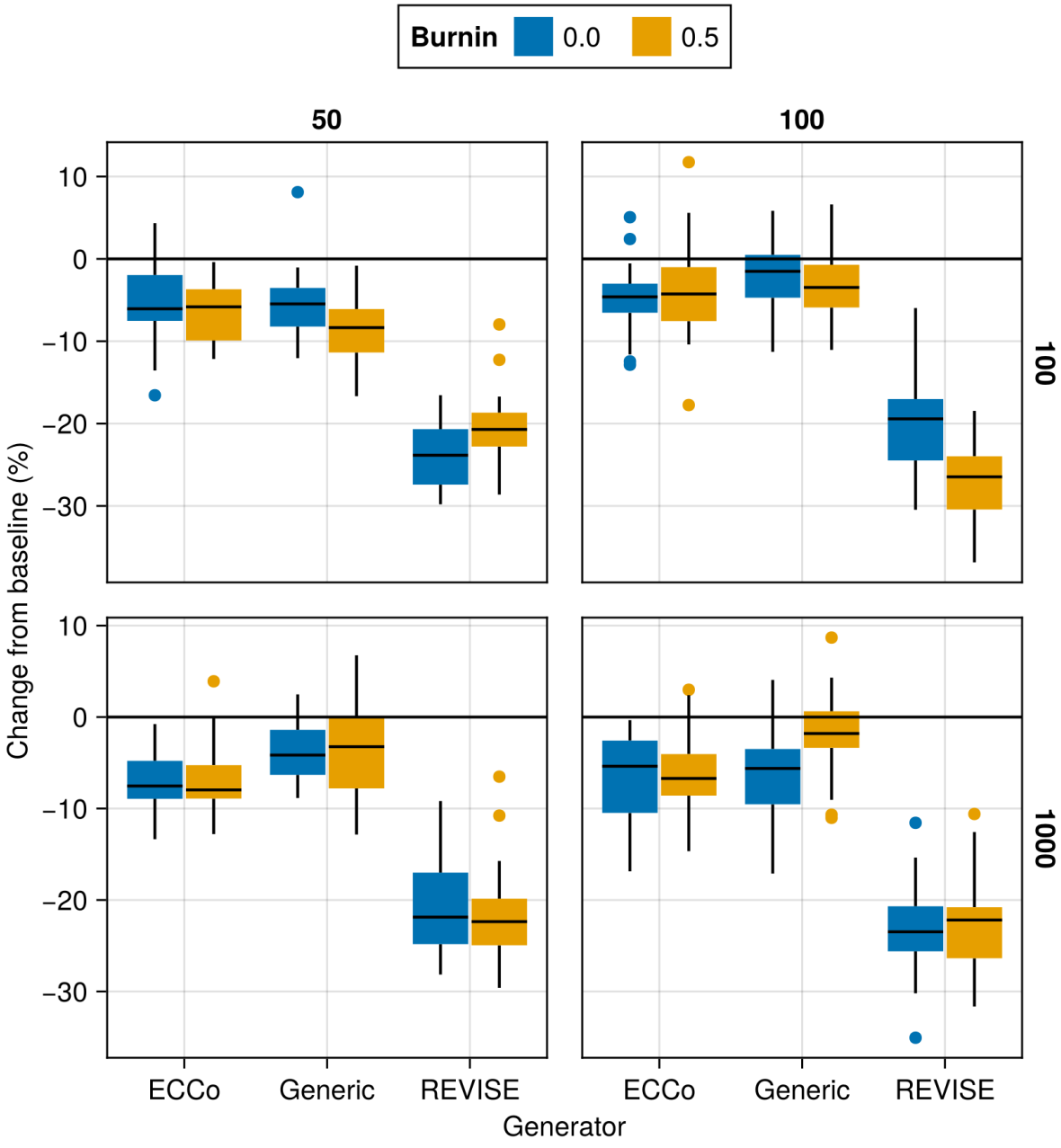

Figure 27: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.

Appendix E Tuning Key Parameters

Based on the findings from our initial large grid searches (Section D), we tune selected hyperparameters for all datasets: namely, the decision threshold τ and the strength of the energy regularization λ_{reg}. The final hyperparameter choices for each dataset are presented in Table 2 in Section C. Detailed results for each data set are shown in Figure 28 to Figure 45. From Table 2, we notice that the same decision threshold of $\tau = 0.5$ is optimal for all but on dataset. We attribute this to the fact that a low decision threshold results in a higher share of mature counterfactuals and hence more opportunities for the model to learn from examples (Figure 37 to Figure 45). This has played a role in particular for our real-world tabular datasets and MNIST, which suffered from low levels of maturity for higher decision thresholds. In cases where maturity is not an issue, as for *Moons*, higher decision thresholds lead to better outcomes, which may have to do with the fact that the resulting counterfactuals are more faithful to the model. Concerning the regularization strength, we find somewhat high variation across datasets. Most notably, we find that relatively low levels of regularization are optimal for MNIST. We hypothesize that this finding may be attributed to the uniform scaling of all input features (digits).

Finally, to increase the proportion of mature counterfactuals for some datasets, we have also investigated the effect on the learning rate η for the counterfactual search and even smaller regularization strengths for a fixed decision threshold of 0.5 (Figure 46 to Figure 51). For the given low decision threshold, we find that the learning rate has no discernable impact on the proportion of mature counterfactuals (Figure 52 to Figure 57). We do notice, however, that the results for MNIST are much improved when using a low value λ_{reg}, the strength for the engery regularization: plausibility is increased by up to ~10% (Figure 50) and the proportion of mature counterfactuals reaches 100%.

One consideration worth exploring is to combine high decision thresholds with high learning rates, which we have not investigated here.

> Package Version (Reproducibility)
>
> Tuning was run using `v1.1.3` of `TaijaData`. The follow-up version `v1.1.4` introduced an option to split real-world tabular datasets into train and test set, ensuring that pre-processing steps like standardization is fit on the training set only. If you are rerunning the tuning experiments with a version of `TaijaData` that is higher than `v1.1.3`, than for the default parameters specified in the configuration files, you may end up with slightly different results, although we would not expect any changes in terms of qualitative findings. For exact reproducibility, please use `v1.1.3`.

E.1 Key Parameters

The hyperparameter grid for tuning key parameters is shown in Note 7. The corresponding evaluation grid used for these experiments is shown in Note 8.

> Note 7: Training Phase
>
> - Generator Parameters:
>   - Decision Threshold: `0.5, 0.75, 0.9`
> - Model: `mlp`
> - Training Parameters:
>   - λ_{reg}: `0.1, 0.25, 0.5`
>   - Objective: `full, vanilla`

> Note 8: Evaluation Phase
>
> - Generator Parameters:
>   - λ_{egy}: `0.1, 0.5, 1.0, 5.0, 10.0`

E.1.1 Plausibility

The results with respect to the plausibility measure are shown in Figure 28 to Figure 36.

E.1.2 Proportion of Mature CE

The results with respect to the proportion of mature counterfactuals in each epoch are shown in Figure 37 to Figure 45.

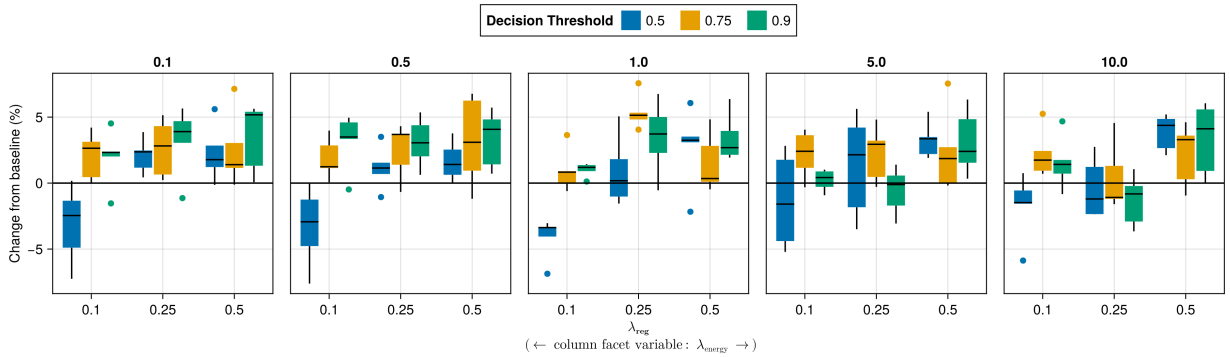

Figure 28: Average outcomes for the plausibility measure across key hyperparameters. Data: Adult.

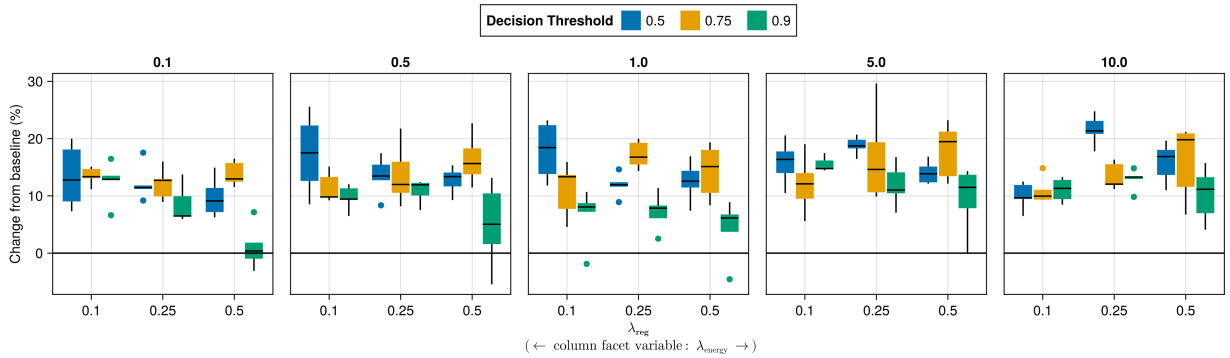

Figure 29: Average outcomes for the plausibility measure across key hyperparameters. Data: California Housing.

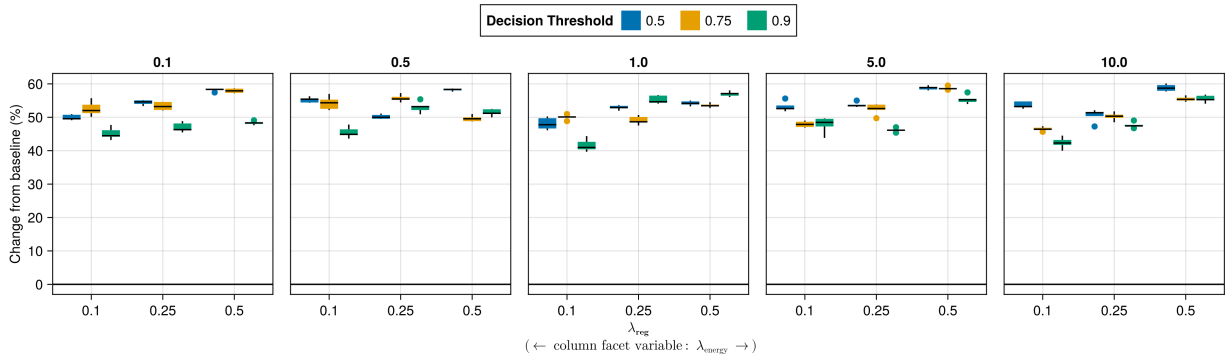

Figure 30: Average outcomes for the plausibility measure across key hyperparameters. Data: Circles.

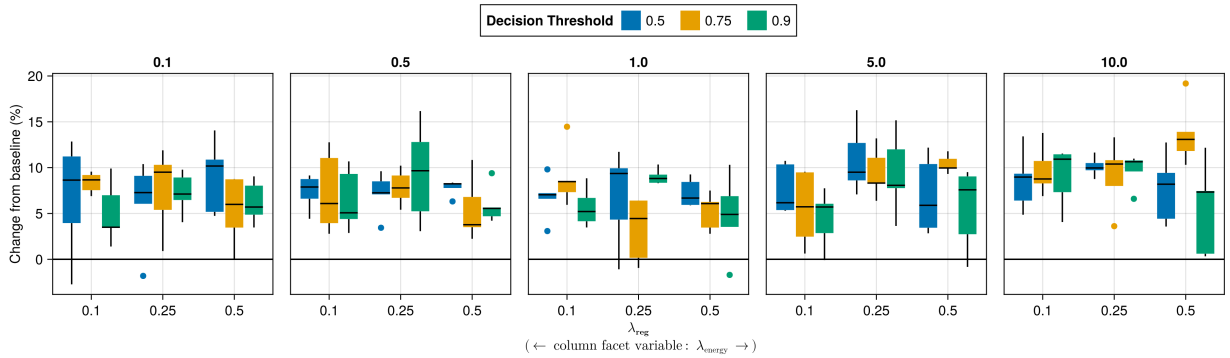

Figure 31: Average outcomes for the plausibility measure across key hyperparameters. Data: Credit.

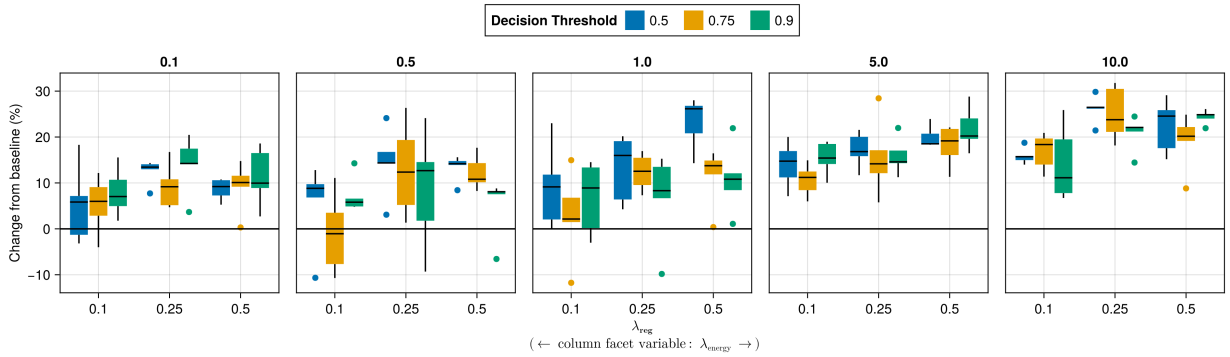

Figure 32: Average outcomes for the plausibility measure across key hyperparameters. Data: GMSC.

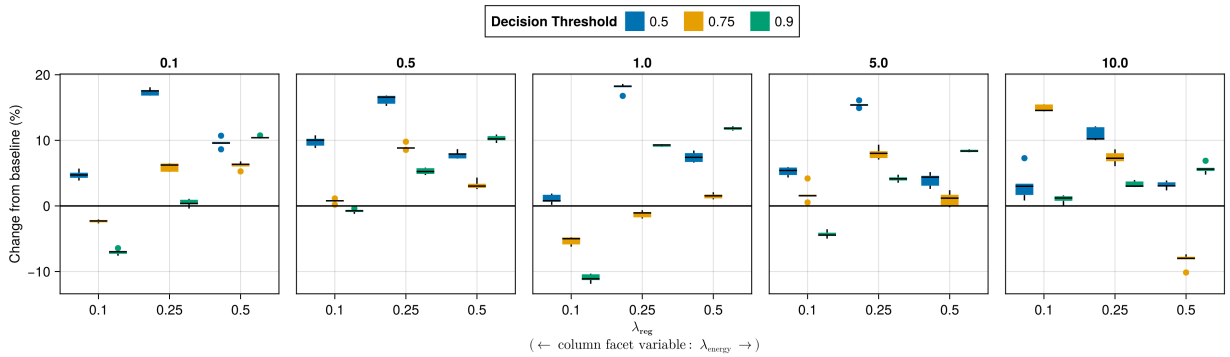

Figure 33: Average outcomes for the plausibility measure across key hyperparameters. Data: Linearly Separable.

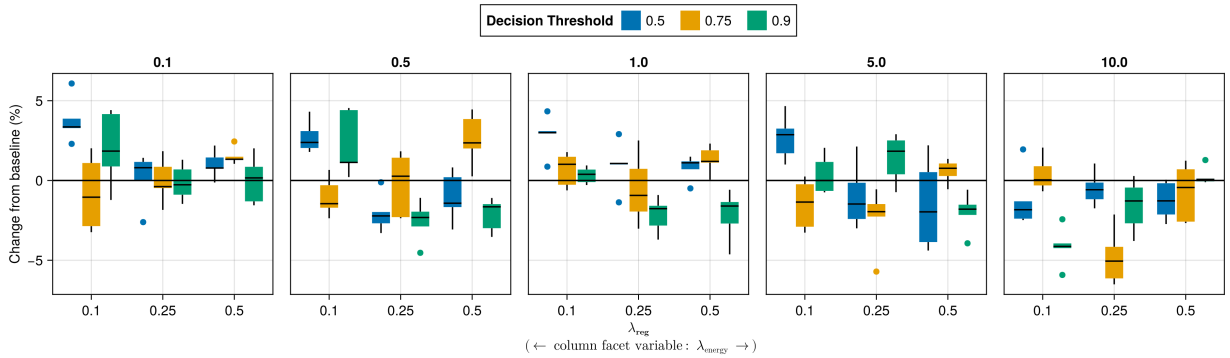

Figure 34: Average outcomes for the plausibility measure across key hyperparameters. Data: MNIST.

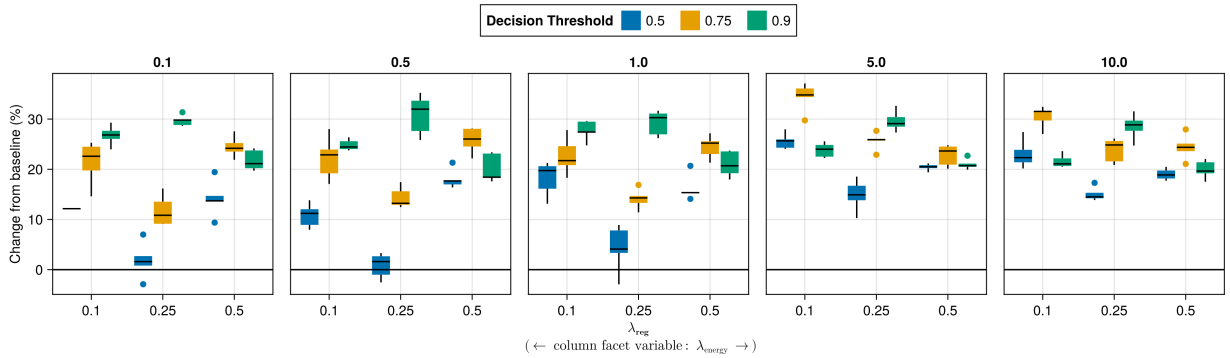

Figure 35: Average outcomes for the plausibility measure across key hyperparameters. Data: Moons.

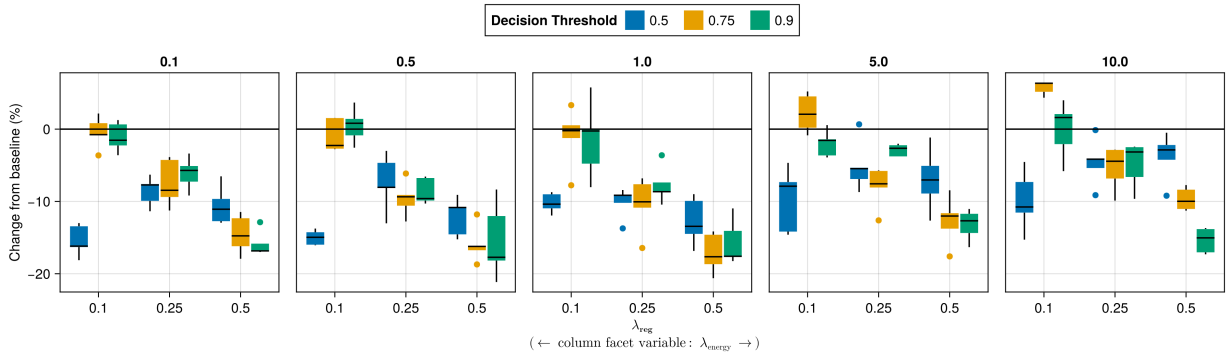

Figure 36: Average outcomes for the plausibility measure across key hyperparameters. Data: Overlapping.

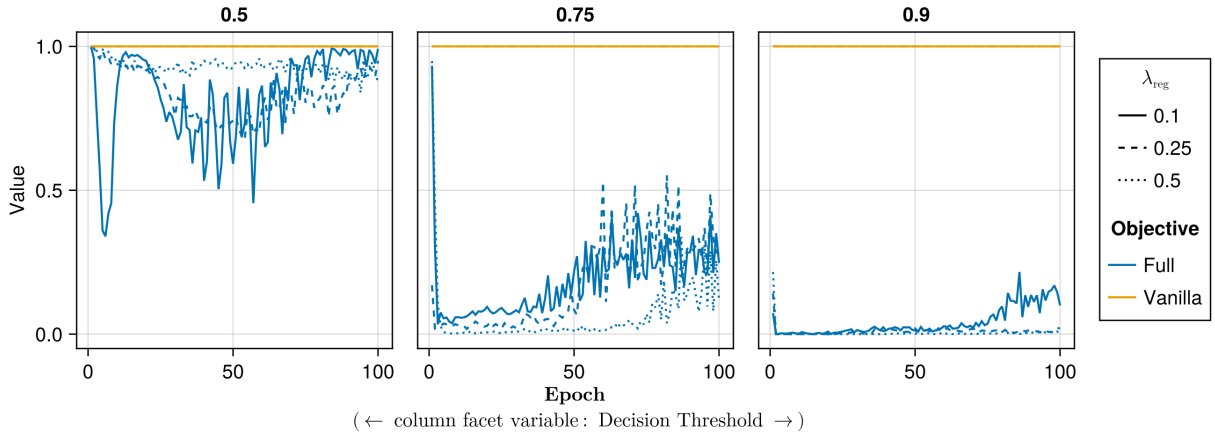

Figure 37: Proportion of mature counterfactuals in each epoch. Data: Adult.

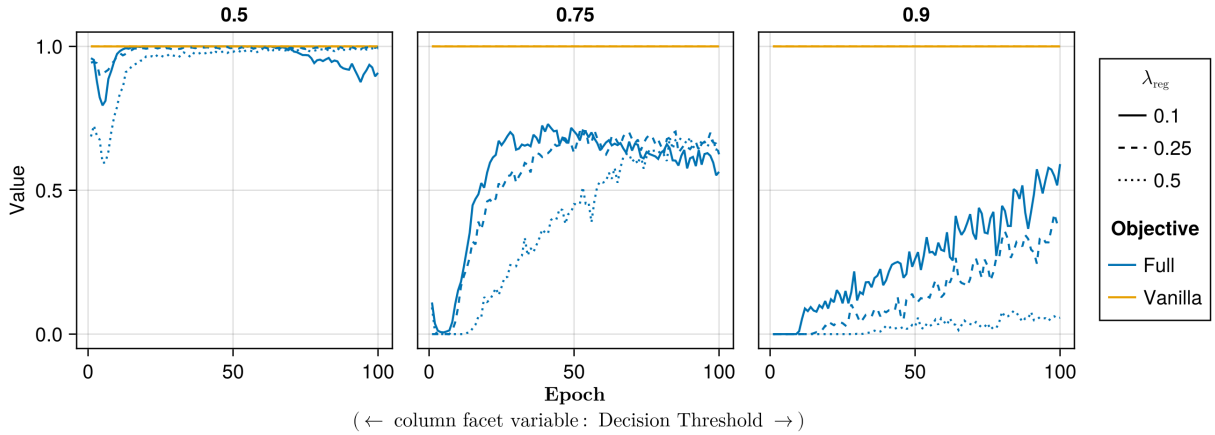

Figure 38: Proportion of mature counterfactuals in each epoch. Data: California Housing.

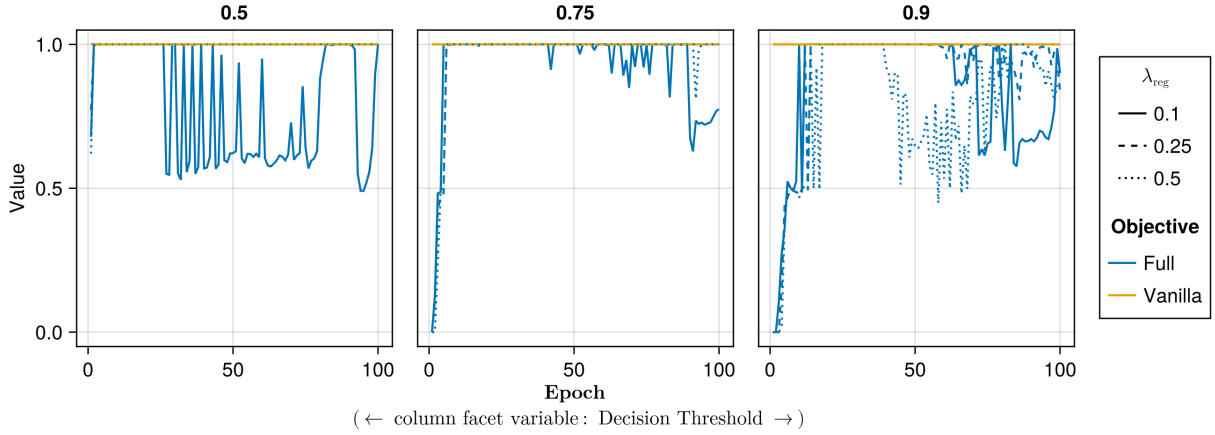

Figure 39: Proportion of mature counterfactuals in each epoch. Data: Circles.

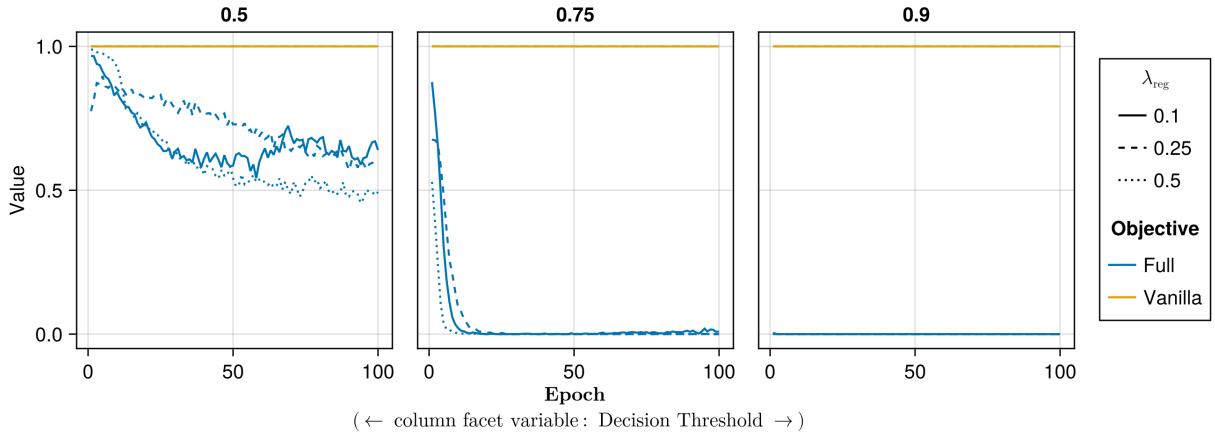

Figure 40: Proportion of mature counterfactuals in each epoch. Data: Credit.

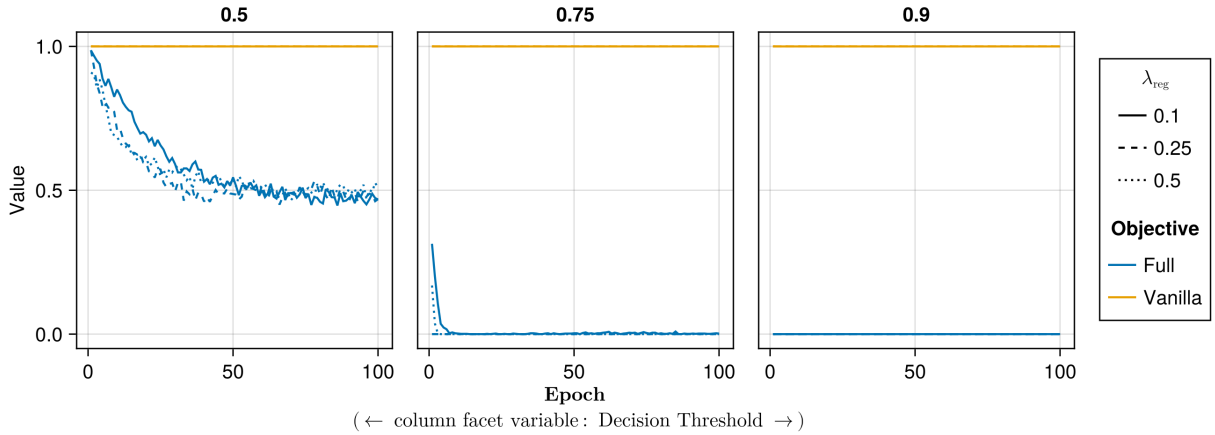

Figure 41: Proportion of mature counterfactuals in each epoch. Data: GMSC.

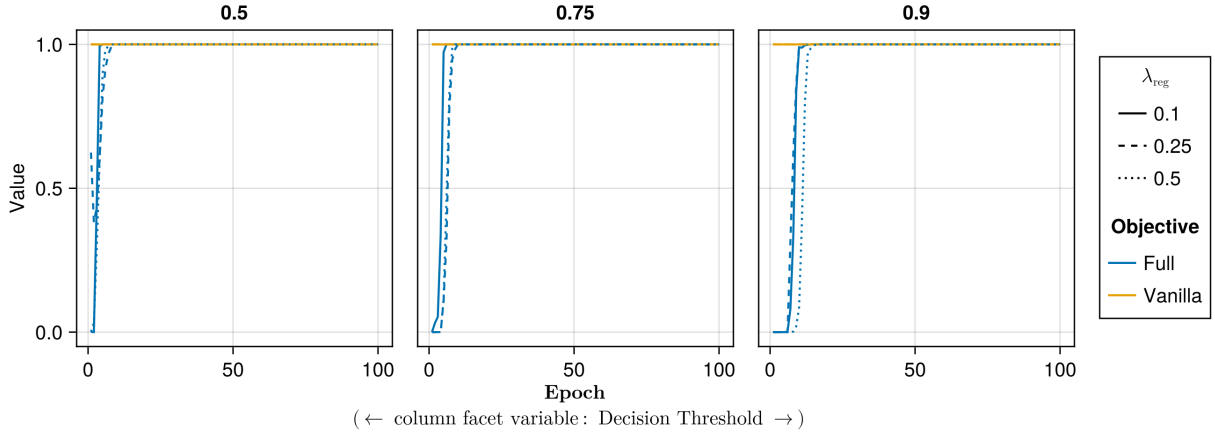

Figure 42: Proportion of mature counterfactuals in each epoch. Data: Linearly Separable.

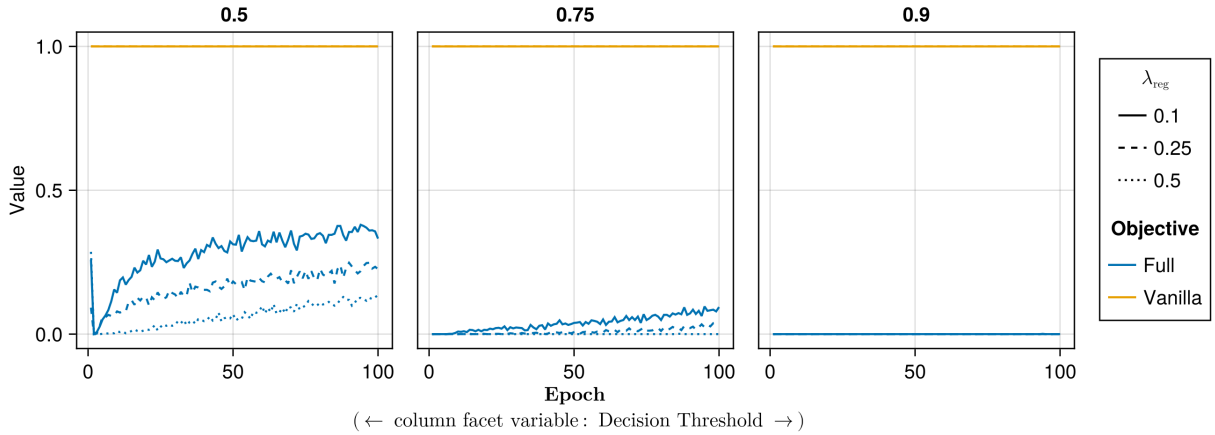

Figure 43: Proportion of mature counterfactuals in each epoch. Data: MNIST.

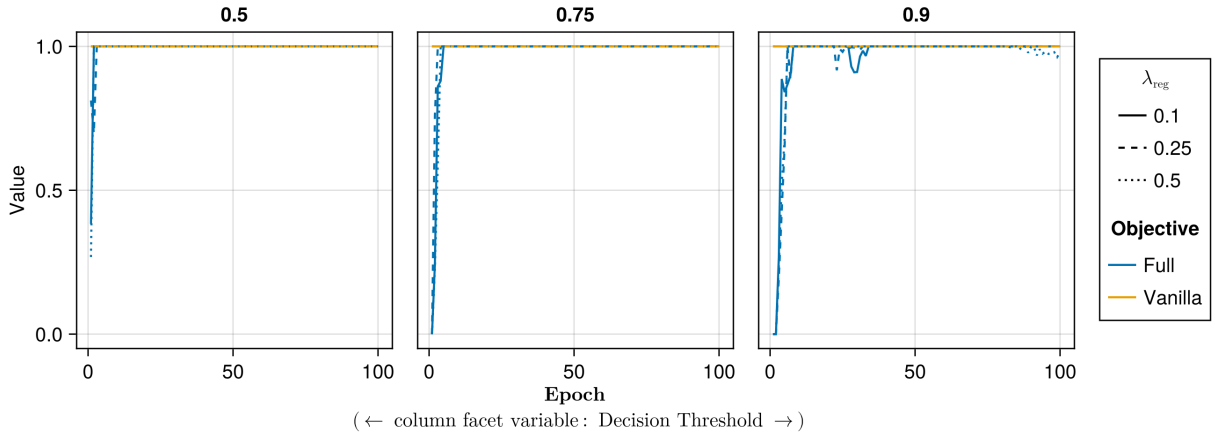

Figure 44: Proportion of mature counterfactuals in each epoch. Data: Moons.

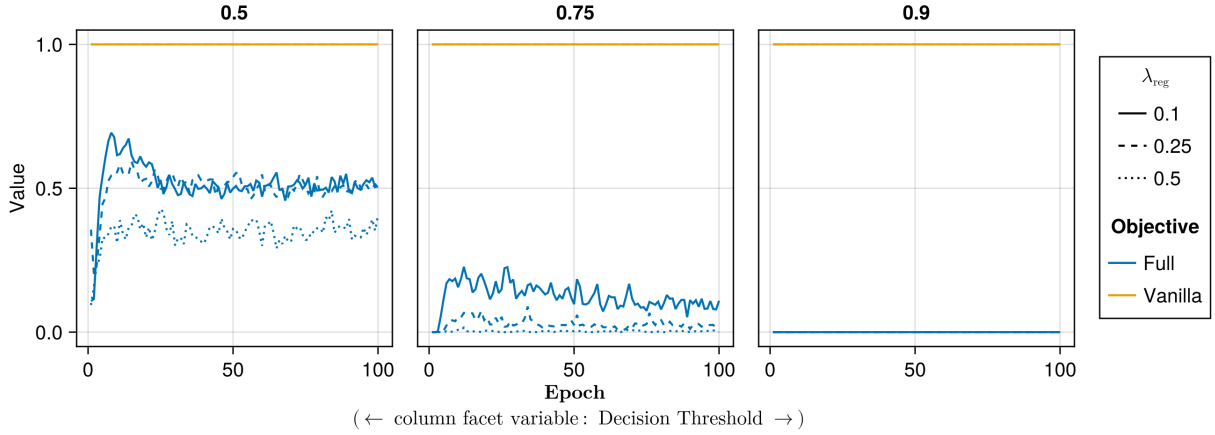

Figure 45: Proportion of mature counterfactuals in each epoch. Data: Overlapping.

E.2 Learning Rate

The hyperparameter grid for tuning the learning rate is shown in Note 9. The corresponding evaluation grid used for these experiments is shown in Note 10.

Note 9: Training Phase

- Generator Parameters:
  - Learning Rate: `0.1, 0.5, 1.0`
- Model: `mlp`
- Training Parameters:
  - λ_{reg}: `0.01, 0.1, 0.5`
  - Objective: `full, vanilla`

Note 10: Evaluation Phase

- Generator Parameters:
  - λ_{egy}: `0.1, 0.5, 1.0, 5.0, 10.0`

E.2.1 Plausibility

The results with respect to the plausibility measure are shown in Figure 46 to Figure 51.

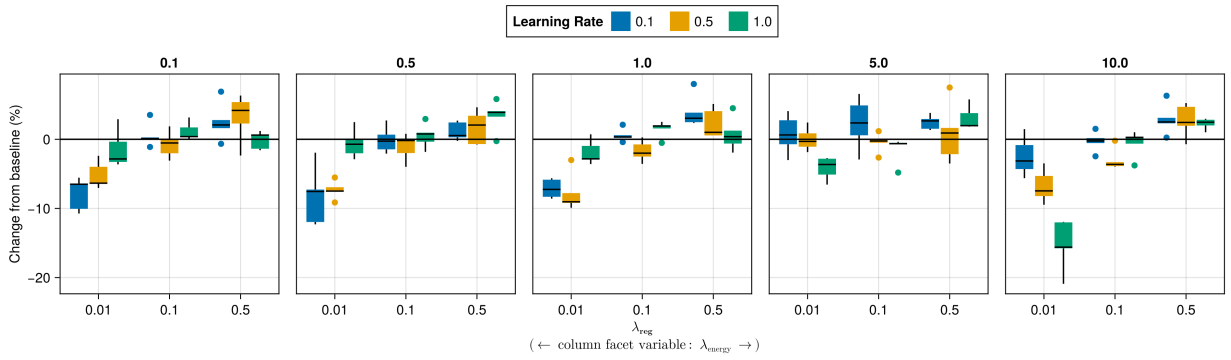

Figure 46: Average outcomes for the plausibility measure across key hyperparameters. Data: Adult.

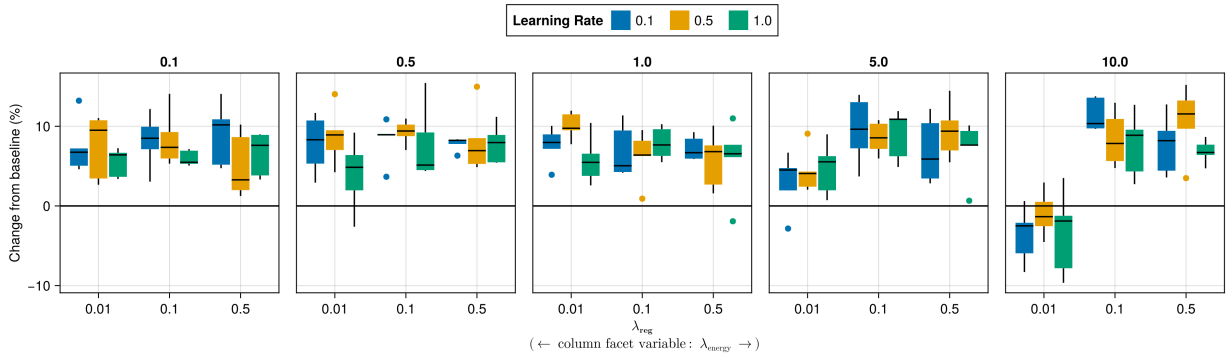

Figure 47: Average outcomes for the plausibility measure across key hyperparameters. Data: Credit.

E.2.2 Proportion of Mature CE

The results with respect to the proportion of mature counterfactuals in each epoch are shown in Figure 52 to Figure 57.

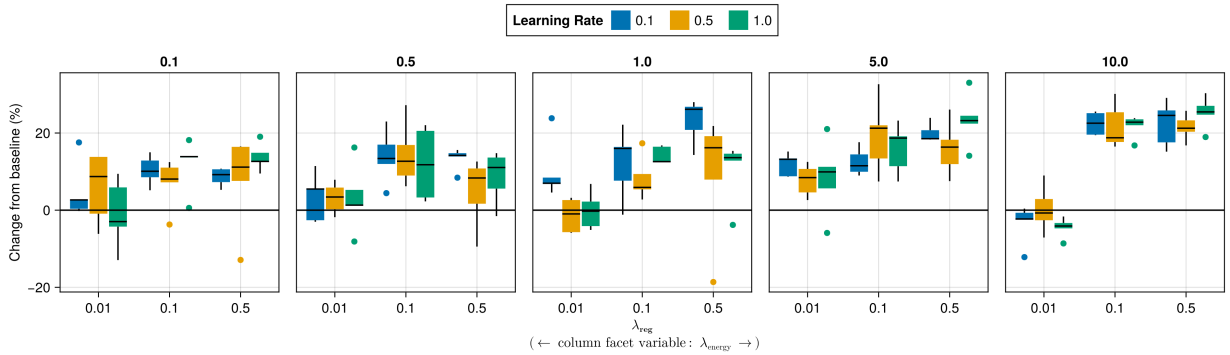

Figure 48: Average outcomes for the plausibility measure across key hyperparameters. Data: GMSC.

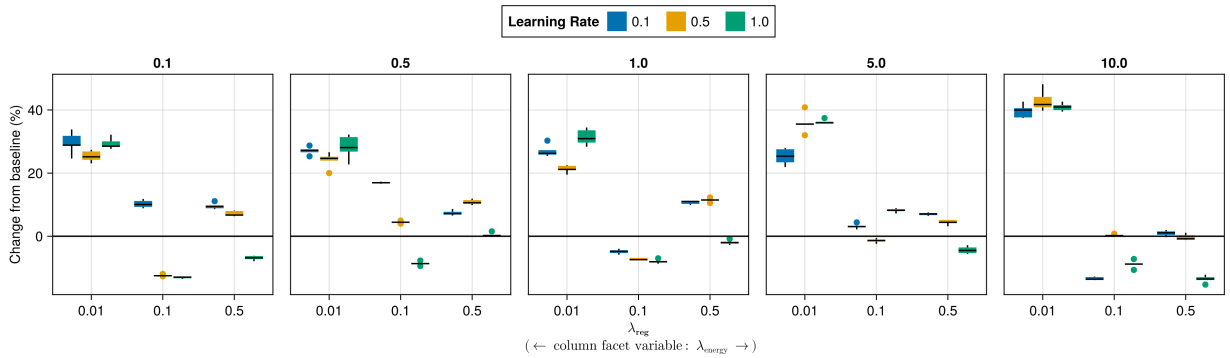

Figure 49: Average outcomes for the plausibility measure across key hyperparameters. Data: Linearly Separable.

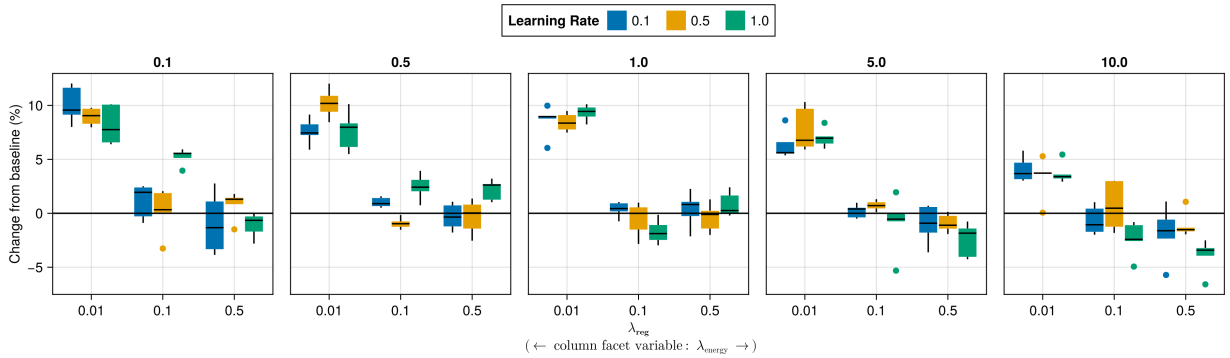

Figure 50: Average outcomes for the plausibility measure across key hyperparameters. Data: MNIST.

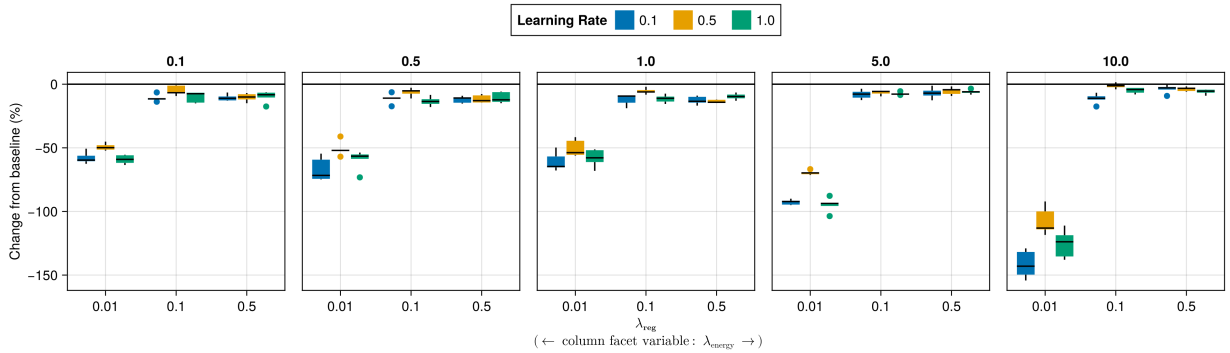

Figure 51: Average outcomes for the plausibility measure across key hyperparameters. Data: Overlapping.

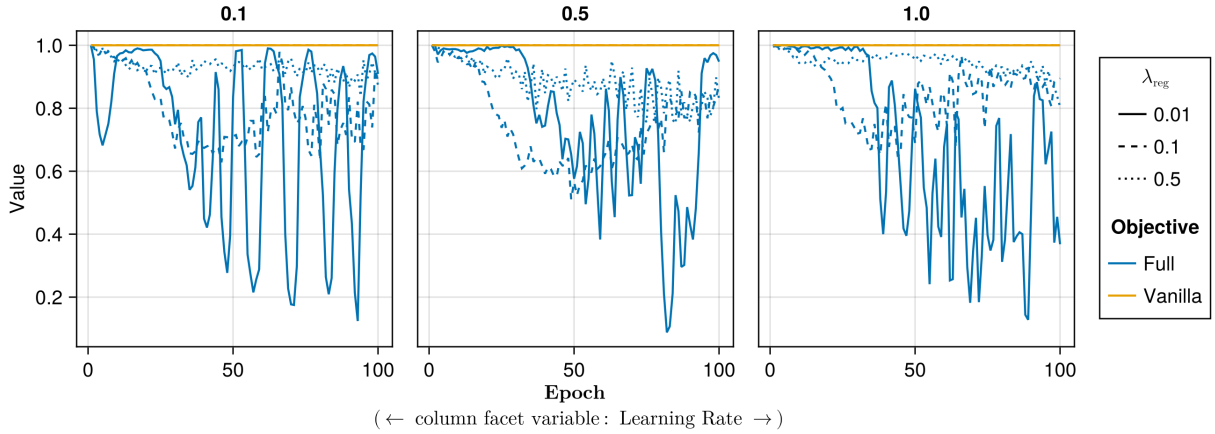

Figure 52: Proportion of mature counterfactuals in each epoch. Data: Adult.

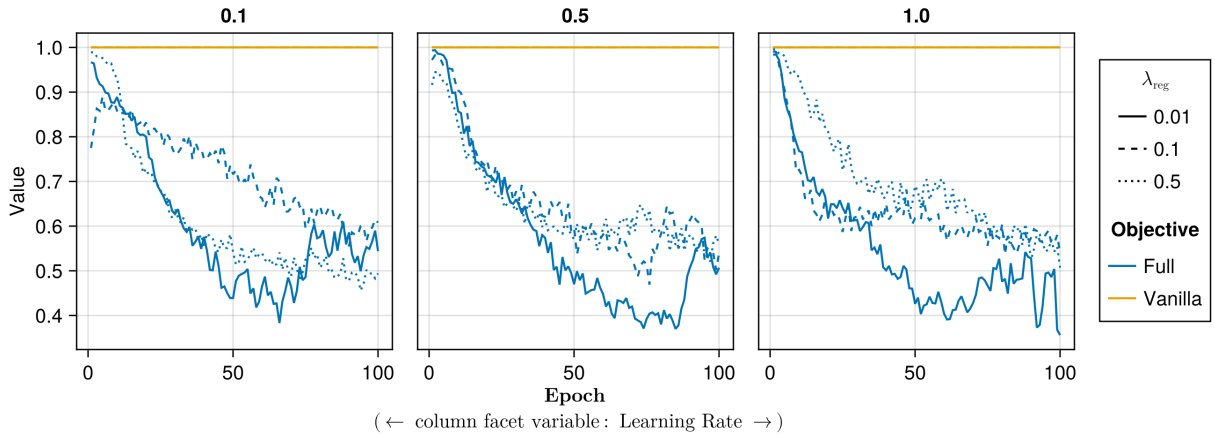

Figure 53: Proportion of mature counterfactuals in each epoch. Data: Credit.

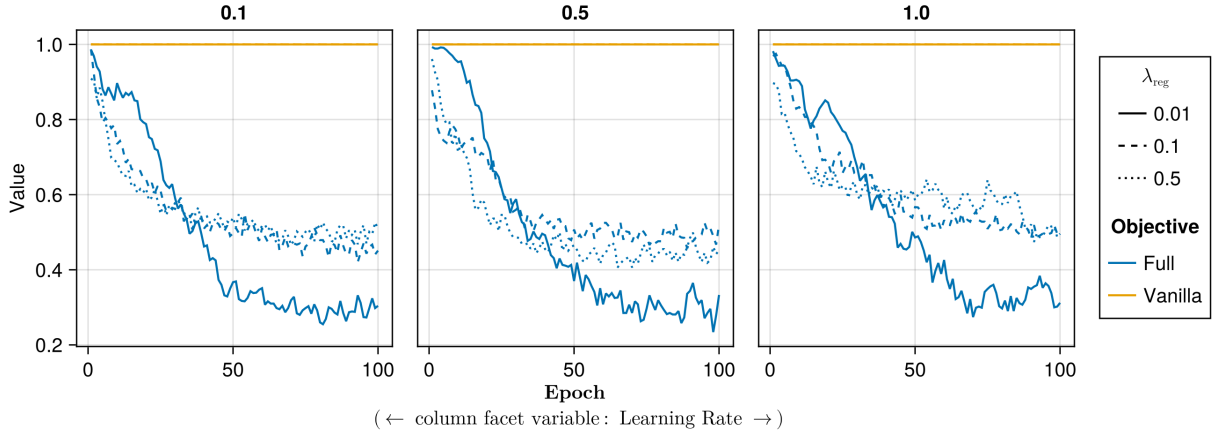

Figure 54: Proportion of mature counterfactuals in each epoch. Data: GMSC.

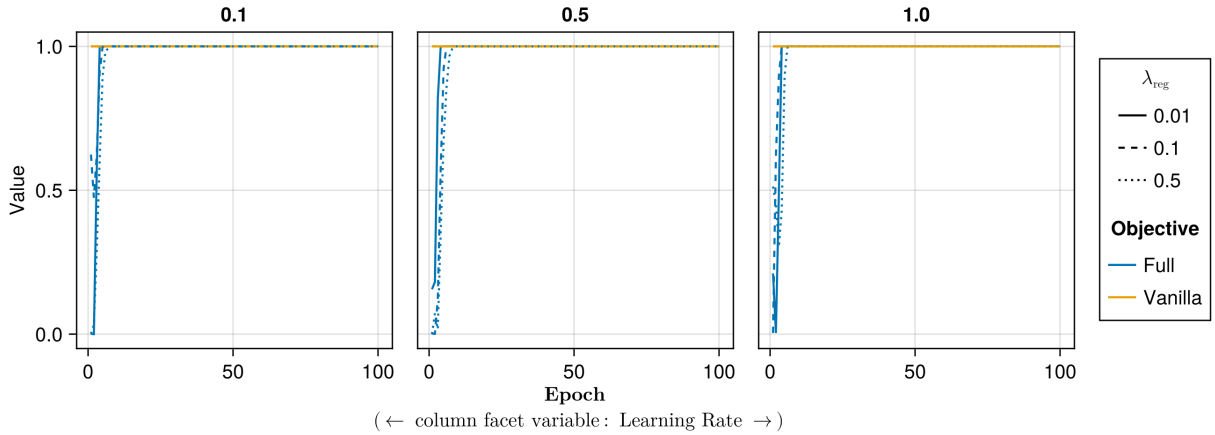

Figure 55: Proportion of mature counterfactuals in each epoch. Data: Linearly Separable.

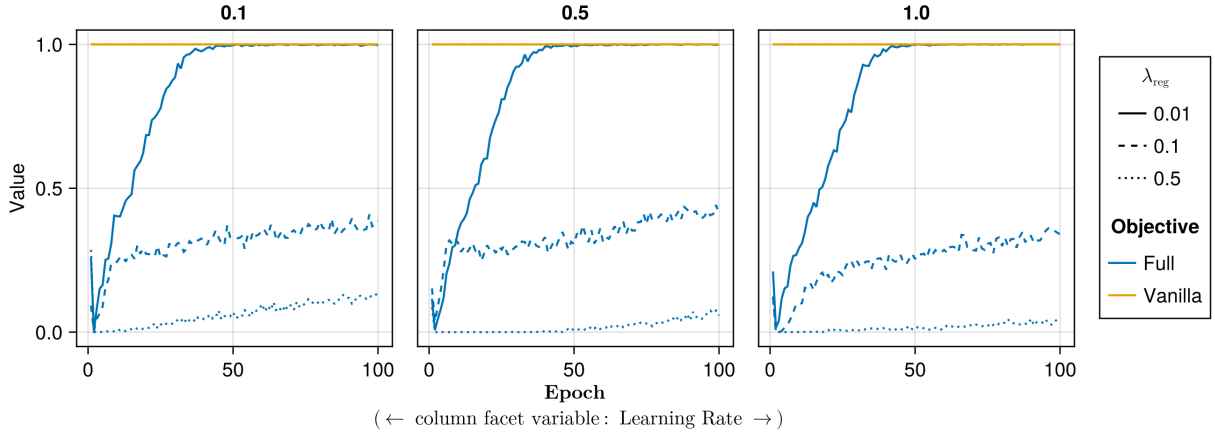

Figure 56: Proportion of mature counterfactuals in each epoch. Data: MNIST.

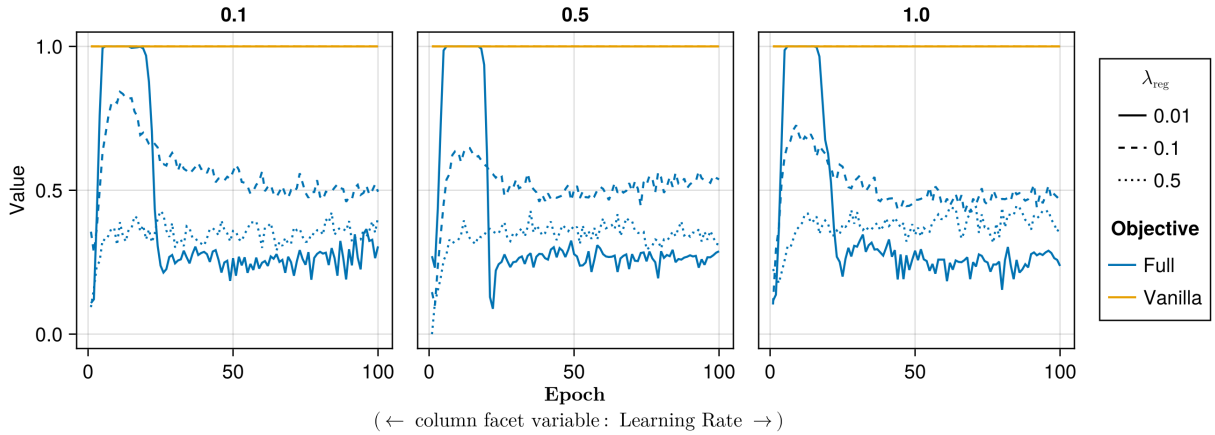

Figure 57: Proportion of mature counterfactuals in each epoch. Data: Overlapping.

Appendix F Computation Details

F.1 Hardware

We performed our experiments on a high-performance cluster. Details about the cluster will be disclosed upon publication to avoid revealing information that might interfere with the double-blind review process. Since our experiments involve highly parallel tasks and rather small models by today's standard, we have relied on distributed computing across multiple central processing units (CPU). Graphical processing units (GPU) were not required.

F.1.1 Grid Searches

Model training for the largest grid searches with 270 unique parameter combinations was parallelized across 34 CPUs with 2GB memory each. The time to completion varied by dataset for reasons discussed in Section 5: 0h49m (*Moons*), 1h4m (*Linearly Separable*), 1h49m (*Circles*), 3h52m (*Overlapping*). Model evaluations for large grid searches were parallelized across 20 CPUs with 3GB memory each. Evaluations for all data sets took less than one hour (<1h) to complete.

F.1.2 Tuning

For tuning of selected hyperparameters, we distributed the task of generating counterfactuals during training across 40 CPUs with 2GB memory each for all tabular datasets. Except for the *Adult* dataset, all training runs were completed in less that half an hour (<0h30m). The *Adult* dataset took around 0h35m to complete. Evaluations across 20 CPUs with 3GB memory each generally took less than 0h30m to complete. For *MNIST*, we relied on 100 CPUs with 2GB memory each. For the *MLP*, training of all models could be completed in 1h30m, while the evaluation across 20 CPUs (6GB memory) took 4h12m. For the *CNN*, training of all models took ~8h, with conventionally trained models taking ~0h15m each and model with CT taking ~0h30m-0h45m each.

F.2 Software

All computations were performed in the Julia Programming Language (Bezanson et al. 2017). We have developed a package for counterfactual training that leverages and extends the functionality provided by several existing packages, most notably CounterfactualExplanations.jl (Altmeyer, Deursen, and Liem 2023) and the Flux.jl library for deep learning (Michael Innes et al. 2018; Mike Innes 2018). For data-wrangling and presentation-ready tables we relied on DataFrames.jl (Bouchet-Valat and Kamiski 2023) and PrettyTables.jl (Chagas et al. 2024), respectively. For plots and visualizations we used both Plots.jl (Christ et al. 2023) and Makie.jl (Danisch and Krumbiegel 2021), in particular AlgebraOfGraphics.jl. To distribute computational tasks across multiple processors, we have relied on MPI.jl (Byrne, Wilcox, and Churavy 2021).