COUNTERFACTUAL TRAINING: TEACHING MODELS PLAUSIBLE AND ACTIONABLE EXPLANATIONS

SUPPLEMENTARY APPENDIX

January 21, 2026

ABSTRACT

This is the supplementary appendix to our IEEE SaTML 2026 paper titled *Counterfactual Training: Teaching Models Plausible and Actionable Explanations*. It provides helpful details on mathematical notations and formulas, our proposed training regime, extended empirical findings, hyperparameter tuning and grid searches as well as software and computations.

***K*eywords** Counterfactual Training • Counterfactual Explanations • Algorithmic Recourse • Explainable AI • Representation Learning

Table of contents

Appendix A Notation

A.1 Variables and Parameters

Below we provide an overview of some notation used frequently throughout the paper:

- $\mathcal{Y}$: The output domain.
- 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.
- $\mathcal{X}$: The input domain.
- $\mathbf{x}$: a single training sample.
- $\mathbf{x}'$: a counterfactual.
- $t = 1, ..., T$: Step indicator for counterfactual search iterations.
- $\mathbf{x}'_{\text{AE}}$: a nascent counterfactual, defined as a counterfactual that has not yet converged.
- $\mathbf{x}'_{\text{CE}}$: a mature counterfactual, defined as a counterfactual that has either passed a threshold probability or exhausted all T steps.
- $\mathbf{x}^+$: a training sample in the target class (ground-truth).
- $\mathbf{y}^+$: The one-hot encoded output vector for the target class.
- θ: Model parameters (unspecified).
- Θ: Matrix of parameters.
- $\mathbf{M}(\cdot)$: linear predictions (logits) of the classifier.

A.2 Formulas

A.2.1 Maximum Mean Discrepancy

Maximum mean discrepancy is defined as follows,

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

where $k(\cdot, \cdot)$ is a kernel function (Gretton et al. 2012). We make use of a Gaussian kernel with a constant length-scale parameter of 0.5. In our implementation, Equation 1 is by default applied to the entire subset of the training data for which $y = y^+$.

A.3 Other Conventions

In some place of this appendix, we use the terms *full/Full* (i.e. the full CT objective) and *vanilla/Vanilla* (i.e. vanilla training objective) to refer to models trained with counterfactual training (*CT*) and the baseline (*BL*), respectively.

Appendix B Technical Details of Our Approach

B.1 Generating Counterfactuals through Gradient Descent

In this section, we provide some additional 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). A more general version of the counterfactual search objective presented in the paper is as follows,

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

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 $\text{reg}(\cdot)$ 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 yloss$(\cdot)$. All generators also penalize the distance (ℓ_1-norm) of counterfactuals from their original factual state. For *Generic* and *ECCCo*, 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, *ECCCo* uses a penalty that regularizes the energy associated with the counterfactual, $\mathbf{x}'$ (Altmeyer et al. 2024). We omit the conformal set size penalty proposed in the original paper, since a) the authors found faithfulness to primarily depend on the energy penalty and hence this alleviates us from one additional hyperparameter.

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 the main paper, 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 that this approach induces models to be relatively less sensitive to immutable features, propose a theoretical result supporting this and provide empirical evidence that strengthens our argument (both in the main paper and additional findings in this appendix). Below we derive the analytical results in Proposition in the main paper.

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):

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \mathbf{x}^\top \Sigma^{-1}(\mu_k - \mu_m) + \text{const} \tag{2}$$

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

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^{D} \frac{\mu_{k,d} - \mu_{m,d}}{\sigma_d^2} x_d + \text{const} \tag{3}$$

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

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^{D} \left(\theta_{k,d} - \theta_{m,d}\right) x_d + \text{const} \tag{4}$$

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

Based on Equation 3 and Equation 4 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 with respect to coefficient $\theta_{y^+,d}$ is equal to

$$\frac{\partial}{\partial \theta_{y^+,d}} \left(\text{div}(\mathbf{x}^+, \mathbf{x}', \mathbf{y}; \theta)\right) = \frac{\partial}{\partial \theta_{y^+,d}} \left((-\mathbf{M}_\theta(\mathbf{x}^+)[y^+]) - (-\mathbf{M}_\theta(\mathbf{x}')[y^+])\right) = x'_d - x^+_d \tag{5}$$

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 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,

$$\begin{aligned} x_d^{\text{LB}} &= \arg\min_{x_d}\{\mu_d - n_{\sigma_d}\sigma_d, \arg\min_{x_d} x_d\} \\ x_d^{\text{UB}} &= \arg\max_{x_d}\{\mu_d + n_{\sigma_d}\sigma_d, \arg\max_{x_d} x_d\} \end{aligned} \tag{6}$$

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 Hyperparameters

Note 1 presents the default hyperparameters used during training.

Note 1: Training Phase

- Meta Parameters:
  - Generator: `ecco`
  - Model: `mlp`
- Model:
  - Activation: `relu`
  - No. Hidden: `32`
  - No. Layers: `1`
- Training Parameters:
  - Burnin: `0.0`
  - Class Loss: `logitcrossentropy`
  - Convergence: `threshold`
  - Generator Parameters:
    - Decision Threshold: `0.75`
    - λ_{cst}: `0.001`
    - λ_{egy}: `5.0`
    - Learning Rate: `0.25`
    - Maximum Iterations: `30`

    * Optimizer: `sgd`
    * Type: `ECCo`
  - λ_{adv}: `0.25`
  - λ_{clf}: `1.0`
  - λ_{div}: `0.5`
  - λ_{reg}: `0.1`
  - Learning Rate: `0.001`
  - No. Counterfactuals: `1000`
  - No. Epochs: `100`
  - Objective: `full`
  - Optimizer: `adam`

B.5 Evaluation Details

B.5.1 Counterfactual Outcomes

For all of our counterfactual evaluations, we proceed as follows: for each dataset we run J bootstrap rounds ("No. Runs") to account for stochasticity (Note 2); for each bootstrap round, we randomly draw factual and target pairs; then, for each model, we draw samples from the test set (with replacement) for which the model predicts the randomly chosen factual class; finally, we generate multiple counterfactuals ("No. Counterfactuals") and evaluate the outcomes (Note 2). This is in line with standard practice in the related literature on CE (see e.g. Schut et al. (2021)). For our final results presented in the main paper, we rely on held-out test sets for evaluation. For tuning purposes we rely on training and/or validation sets.

Note 2 presents the default hyperparameters used during evaluation for tuning purposes. For the main results presented in the paper, we use larger evaluations, specifically:

- "No. Runs": We set the number of bootstrap rounds to $J = 100$ for all datasets.
- "No. Individuals": In each round we draw 1,250, 500 and 125 samples for synthetic datasets, real-world tabular datasets and *MNIST*, respectively, across five different values for the strength of the energy penalty of *ECCCo* at test time, $\lambda_{\text{egy}} \in \{0.1, 0.5, 1.0, 5.0, 10.0\}$.

Note 2: Evaluation Phase

- Convergence: `threshold`
- Decision Threshold: `0.95`
- Maximum Iterations: `50`
- No. Individuals: `100`
- No. Runs: `5`

B.5.2 Predictive Performance

To assess (robust) predictive performance, we evaluate model accuracy on (adversarially perturbed) test data. To generate adversarial examples we use the Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2015). For the main results in the paper, we choose a range of values $\epsilon = [0.0, 0.1]$. In some places of this appendix, you will also find predictive performance evaluations in terms of the F1-score.

Appendix C Details on Main Experiments

C.1 Final Hyperparameters

As discussed the main paper, CT is sensitive to certain hyperparameter choices. We study the effect of many hyperparameters extensively in Section D of this appendix. 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 1 along with training, test and batch sizes.

C.1.1 Confidence Intervals

Table 2 present the exact confidence intervals (99%) for the difference in mean outcomes on which we base our assessment of statistical significance in the main paper. Grouped by evaluation metrics (Variable) and dataset (Data), the table presents the mean outcomes for CT and BT and finally the lower bound (LB) and upper bound (UB) of the confidence interval. To compute the intervals, we used the percentile method for bootstrapped confidence intervals:

Table 1: Final hyperparameters used for the main results presented in the main paper. Any hyperparameter not shown here is set to its default value (Note 1).

Data	No. Train	No. Test	Batchsize	Domain	Decision Threshold	No. Counterfactuals	λ_{reg}
LS	3600	600	30	none	0.5	1000	0.01
Circ	3600	600	30	none	0.5	1000	0.5
Moon	3600	600	30	none	0.9	1000	0.25
OL	3600	600	30	none	0.5	1000	0.25
Adult	26049	5010	1000	none	0.75	5000	0.25
CH	16504	3101	1000	none	0.5	5000	0.25
Cred	10617	1923	1000	none	0.5	5000	0.25
GMSC	13371	2474	1000	none	0.5	5000	0.5
MNIST	11000	2000	1000	(-1.0, 1.0)	0.5	5000	0.01

Table 2: Mean outcomes for CT and BL along with bootstrapped confidence intervals (99%) for difference in mean outcomes grouped by dataset and evaluation metric. Column LB and UB show the lower and upper bound of the intervals, respectively, and computed using the percentile method (for significance, interval should not include zero). The underlying counterfactual evaluations are the same as the ones used to produce the main table in the paper.

Variable	Data	CT	BL	LB	UB
Cost	Adult	2.19	2.28	-0.32	0.11
Cost	CH	1.37	2.46	-1.18	-1.0
Cost	Circ	0.7	1.22	-0.55	-0.49
Cost	Cred	2.7	2.29	0.16	0.6
Cost	GMSC	1.03	3.04	-2.37	-1.86
Cost	LS	3.75	4.48	-0.8	-0.67
Cost	MNIST	72.08	53.42	11.15	26.68
Cost	Moon	1.52	1.6	-0.12	-0.05
Cost	OL	1.55	2.62	-1.25	-0.9
IP*	Adult	0.07	0.11	-0.06	-0.02
IP*	CH	0.02	0.06	-0.05	-0.03
IP*	Circ	0.0	0.0	-0.01	-0.0
IP*	Cred	0.03	0.06	-0.05	-0.01
IP*	GMSC	0.05	0.07	-0.02	-0.01
IP*	LS	0.11	0.23	-0.13	-0.11
IP*	MNIST	0.02	0.02	-0.07	0.07
IP*	Moon	0.02	0.02	-0.01	0.0
IP*	OL	0.12	0.09	-0.01	0.05
IP	Adult	15.13	15.16	-0.42	0.39
IP	CH	6.72	7.52	-1.05	-0.6
IP	Circ	0.97	2.36	-1.44	-1.35
IP	Cred	19.79	22.02	-3.17	-1.42
IP	GMSC	7.24	8.1	-1.26	-0.38
IP	LS	2.51	3.4	-0.95	-0.84
IP	MNIST	261.05	278.84	-27.38	-7.51
IP	Moon	1.37	1.71	-0.36	-0.3
IP	OL	4.52	4.44	-0.03	0.19

the lower and upper bound represent the $\alpha/2$- and $(1-\alpha/2)-$quantile of the bootstrap distribution, respectively, for $\alpha = 0.01$.

C.1.2 Qualitative Findings for Image Data

Figure 1 shows much more plausible (faithful) counterfactuals for a model with CT than the model with conventional training (Figure 2).

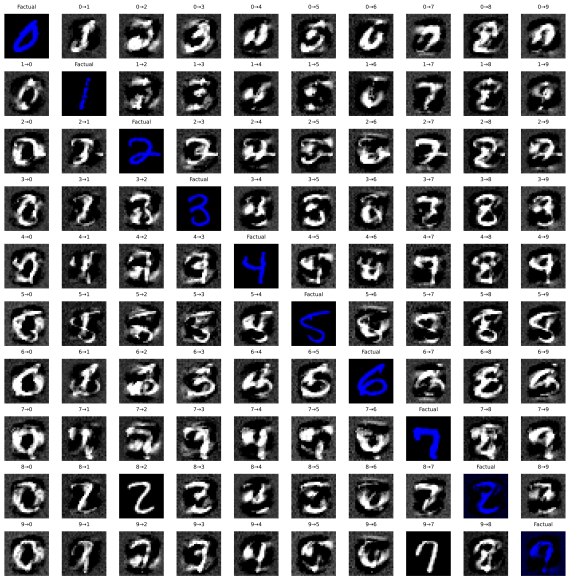

Figure 1: Counterfactual images for *MLP* with counterfactual training. Factual images are shown on the diagonal, with the corresponding counterfactual for each target class (columns) in that same row. The underlying generator, *ECCCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

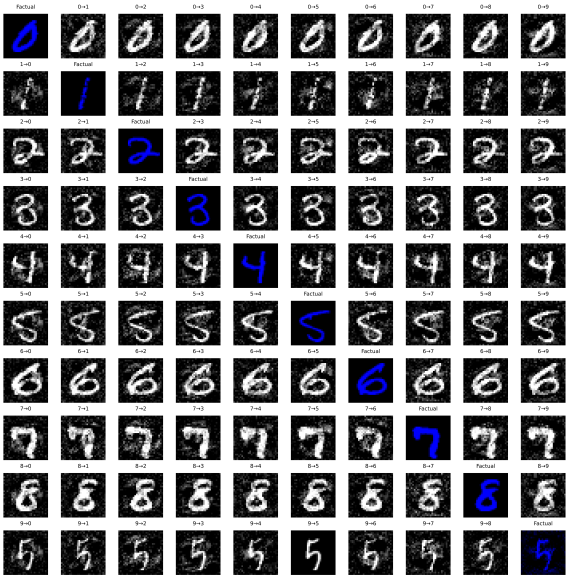

Figure 2: The same setup, factuals, model architecture and generator as in Figure 1, but the model was trained conventionally.

C.1.3 Integrated Gradients

We make use of integrated gradients (IG) proposed by Sundararajan, Taly, and Yan (2017) to empirically evaluate the feature protection mechanism in CT. We choose this approach because it produces theoretically sound results, works well for non-linear models, and remains relatively inexpensive.

IG calculates the contribution of each input feature towards a specific prediction by approximating the integral of the model output with respect to its input, using a set of samples that linearly interpolate between a test instance and some baseline instance (Sundararajan, Taly, and Yan 2017). This process produces a vector of real numbers, one per input feature, which informs about the contribution of each feature to the prediction. For example:

- a large positive value indicates that a feature has strong positive influence on the classification (i.e., increases the score for a class);
- a small negative value indicates that a feature has weak negative influence on the classification (i.e., decreases the score for a class).

To calculate the contributions, IG compares the output to a baseline. The selection of an appropriate baseline is an important design decision — it should produce a "neutral" prediction to avoid capturing effects that cannot be directly attributed to the model (Sundararajan, Taly, and Yan 2017; Sturmfels, Lundberg, and Lee 2020). To remain consistent in our evaluations, we use a baseline drawn at random from a uniform distribution, $\mathcal{U}(-1, 1)$, for all datasets. This aligns with standard evaluation practices for IG.

We run IG on models trained on all datasets to compare their sensitivity to features that were protected using CT:

- for synthetic datasets, this is always the first feature
- for real-world tabular datasets, this is always *age*
- for MNIST, this is first five and last five rows of pixels

As IG outputs are not bounded (i.e., they are arbitrary real numbers), it becomes a challenge to meaningfully compare IG outputs of different models — ones that are trained conventionally, and ones that underwent counterfactual training. For our purposes, we observe with reference to our Proposition, that we are interested estimating changes in the relative

contribution of protected features compared to mutable ones. Thus, to meaningfully compare integrated gradients for different models and to accommodate for variable ranges of outputs in absolute terms, we standardize the integrated gradients across features.

Let $\mathbf{g}_d$ denote the estimated IG for feature d. Then in the case of 2D synthetic datasets we find that taking the absolute value of the outputs, $|\mathbf{g}_d|$, and then dividing them by a $\max(\mathbf{g}) - \min(\mathbf{g})$ term allows us to make the most meaningful comparison. In the case of real-world datasets we choose to normalize the values to a $[0, 1]$ range instead. We compare the (average) sensitivity to the features that were protected for CT models. Once again we use bootstrapping (100 rounds, 2500 samples per round) to establish the significance of our results.

C.1.4 Costs and Validity

In Table 3, we present additional outcomes for common evaluation metrics: Table 3a presents the average reduction in costs of counterfactuals for CT vs. BL with no mutability constraints, i.e. corresponding to the first two columns in the main table of the paper; Table 3b shows the corresponding average validities; finally, Table 3c shows average validities for the case with mutability constraints, i.e. corresponding to the third columns in the main table of the paper.

As noted in the discussion section of the main paper, we observe mixed results results here. Average costs in terms of distances from factual values decrease for most datasets, which is positively surprising since improved plausibility requires counterfactuals to travel further into the target domain than minimum distance counterfactuals. It appears that in these cases faithful counterfactuals for the baseline model still end up far away from their initial starting points, but not close enough for samples in the target domain to be plausible. In that sense, CT can be seen to improve both plausibility and costs for faithful CE. In some cases though (*LS*, *CH*, *MNIST*), we do seem to observe the tradeoff between plausibility and costs play out, as we would expect (compare panels (a) and (b) of Figure 1 in the main paper for reference).

Concerning validity, we find that can lead to substantial reductions and only increases average validity compared to the baseline in one case (*Circ*). As noted in the discussion section of the main paper, this result does not surprise us: by design, CT shrinks the solution space for valid counterfactual explanations, thus making it "harder" to reach validity compared to the baseline model. Note that for a number of reasons this should not be seen as problematic:

1. Validity of gradient-based CE is a function on the number of steps and the step size which we both kept fixed during evaluation: simply adjusting $T = 50$ to higher values or choosing a larger step size will lead to higher rates of validity.
2. Even though reaching validity is sometimes "harder" in terms of the necessary number of steps for a given step size, we have already shown that the average distances that counterfactuals need to travel decrease for most datasets. Users care about costs in terms of feature distances, not search iteration steps.
3. From a philosophical perspective on algorithmic recourse, validity in and off itself is not a sufficient desideratum for counterfactuals. In fact, Venkatasubramanian and Alfano (2020) propose introducing an upper bound on costs of the flipset (i.e. the set of valid CE), arguing that valid but highly costly counterfactuals are not useful to individuals in practice. In a similar fashion, it could be argued that there should be an upper bound on the implausibility of counterfactuals in the flipset.

Table 3: Costs and validity.

(a) Reduction in average costs for CT vs. the baseline. Results correspond to the case with no mutability constraints in the main table of the paper.

Data	**Cost** $(-\%)$
LS	$-27.11 \pm 0.75^*$
Circ	$40.17 \pm 0.85^*$
Moon	$32.54 \pm 1.23^*$
OL	$12.08 \pm 1.58^*$
Adult	-4.59 ± 2.54
CH	$-33.04 \pm 1.96^*$
Cred	$27.43 \pm 1.05^*$
GMSC	$-22.40 \pm 3.64^*$
MNIST	$-40.71 \pm 7.02^*$
Avg.	-1.74

(b) Average validities of counterfactuals for CT and BL. Unconstrained case.

Data	**CT**	**BL**
LS	1.0	1.0
Circ	1.0	0.51
Moon	1.0	1.0
OL	0.86	0.98
Adult	0.68	0.99
CH	1.0	1.0
Cred	0.72	1.0
GMSC	0.94	1.0
MNIST	1.0	1.0

(c) Average validities of counterfactuals for CT and BL. Mutability constrained case.

Data	**CT**	**BL**
LS	1.0	1.0
Circ	0.71	0.48
Moon	1.0	0.98
OL	0.34	0.56
Adult	0.7	0.99
CH	1.0	1.0
Cred	0.74	1.0
GMSC	0.97	1.0
MNIST	1.0	1.0

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 our proposed training objective.
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 4, Table 5, Table 6). 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-based adaption (IP*) 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 4, Table 5, Table 6). 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 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 3), 25-30% for the *Moons* data (e.g. Figure 5) and 10-20% for the *Linearly Separable* data (e.g. Figure 4). At the same time, the average costs of faithful counterfactuals are reduced in many cases by around 20-25% for *Circles* (e.g. Figure 7) and up to 50% for *Moons* (e.g. Figure 9). For the *Linearly Separable* data, costs are generally increased although typically by less than 10% (e.g. Figure 8), 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 *ECCCo* 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 *ECCCo*. 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 6): 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 11), *Linearly Separable* (Figure 12) and *Overlapping* (Figure 14). 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 19) and *Moons* (Figure 21). 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 3. The corresponding evaluation grid used for these experiments is shown in Note 4.

Note 3: 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 4: 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 4.

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	Se
Circ	Accuracy	Full	1.0	0.0
Circ	Accuracy	Vanilla	1.0	0.0
Circ	F1-score	Full	1.0	0.0
Circ	F1-score	Vanilla	1.0	0.0
LS	Accuracy	Full	1.0	0.0
LS	Accuracy	Vanilla	1.0	0.0
LS	F1-score	Full	1.0	0.0
LS	F1-score	Vanilla	1.0	0.0
Moon	Accuracy	Full	1.0	0.0
Moon	Accuracy	Vanilla	1.0	0.0
Moon	F1-score	Full	1.0	0.0
Moon	F1-score	Vanilla	1.0	0.0
OL	Accuracy	Full	0.91	0.0
OL	Accuracy	Vanilla	0.92	0.0
OL	F1-score	Full	0.91	0.0
OL	F1-score	Vanilla	0.92	0.0

D.2.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 3 to Figure 6.

D.2.3 Cost

The results with respect to the cost measure are shown in Figure 7 to Figure 10.

D.3 Penalty Strengths

The hyperparameter grid with varying penalty strengths during training 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:
  - 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 6: Evaluation Phase

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

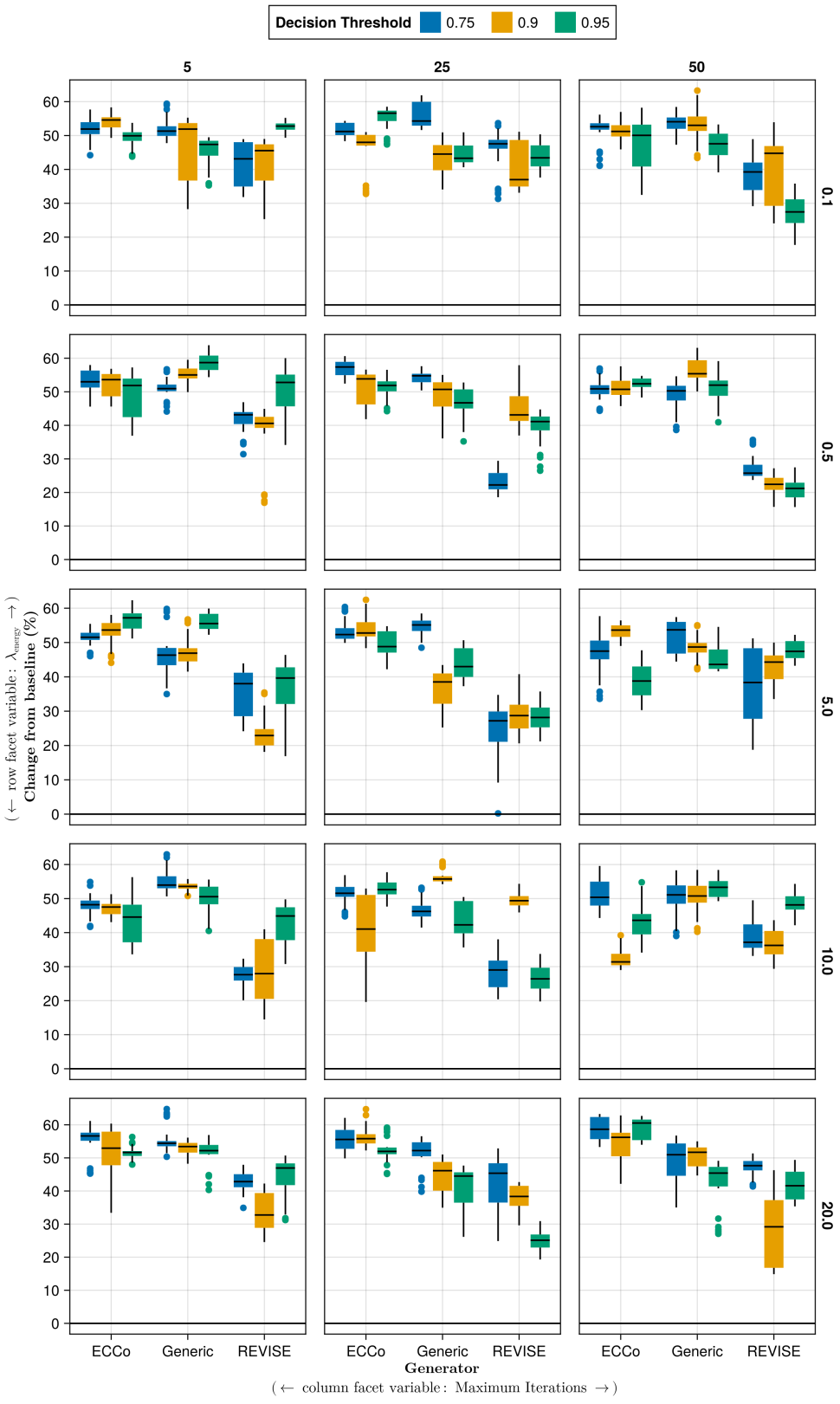

Figure 3: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

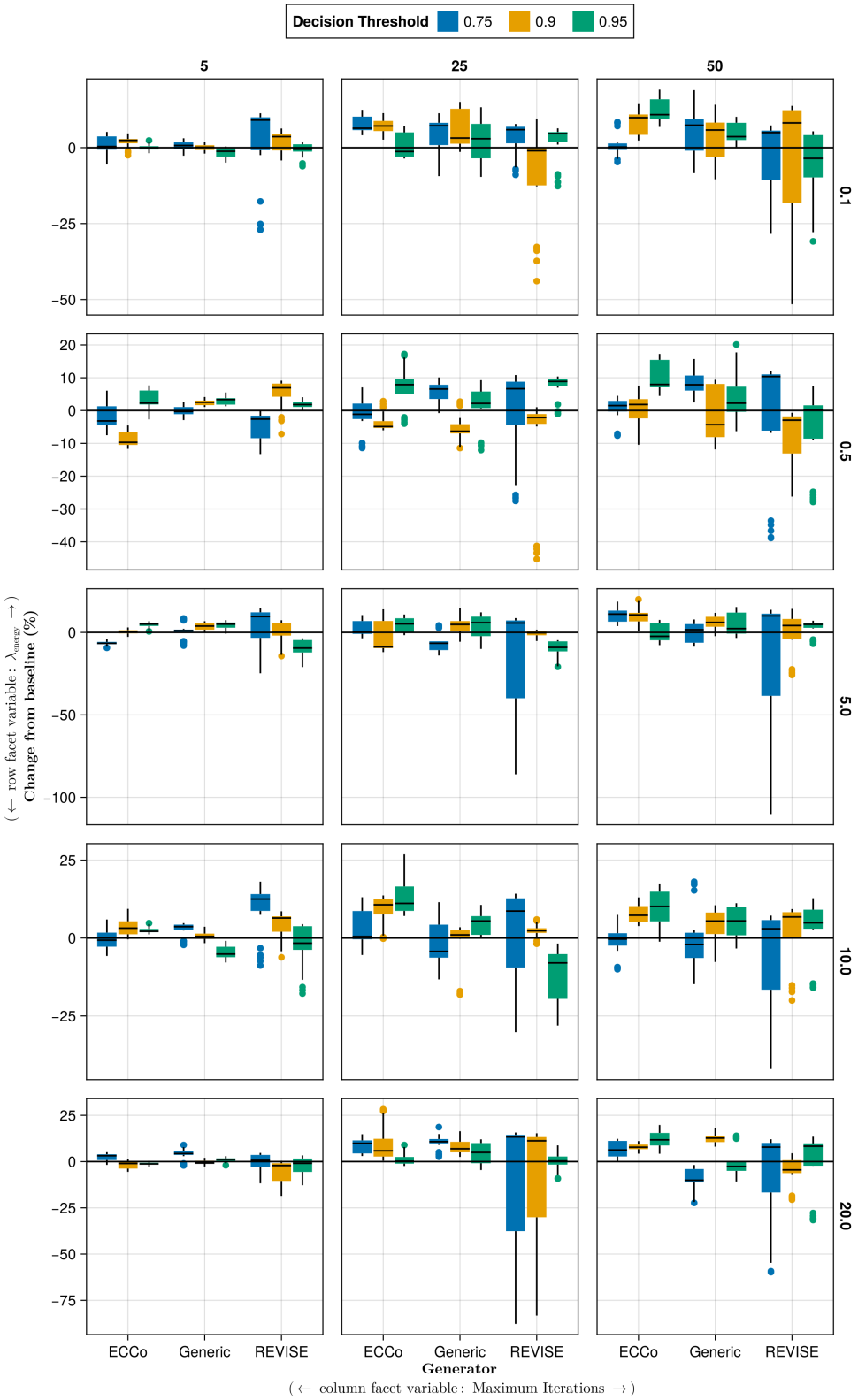

Figure 4: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

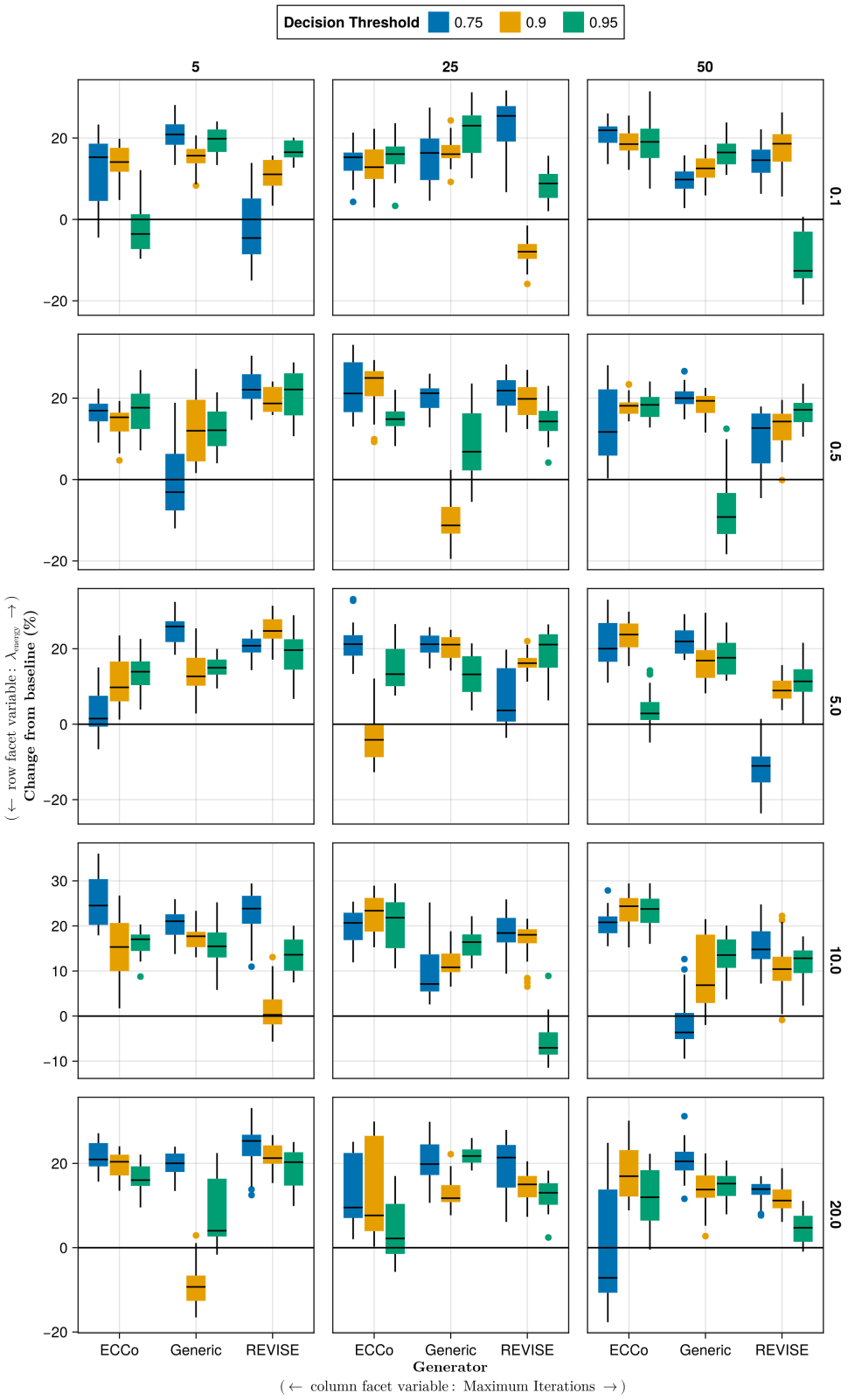

Figure 5: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

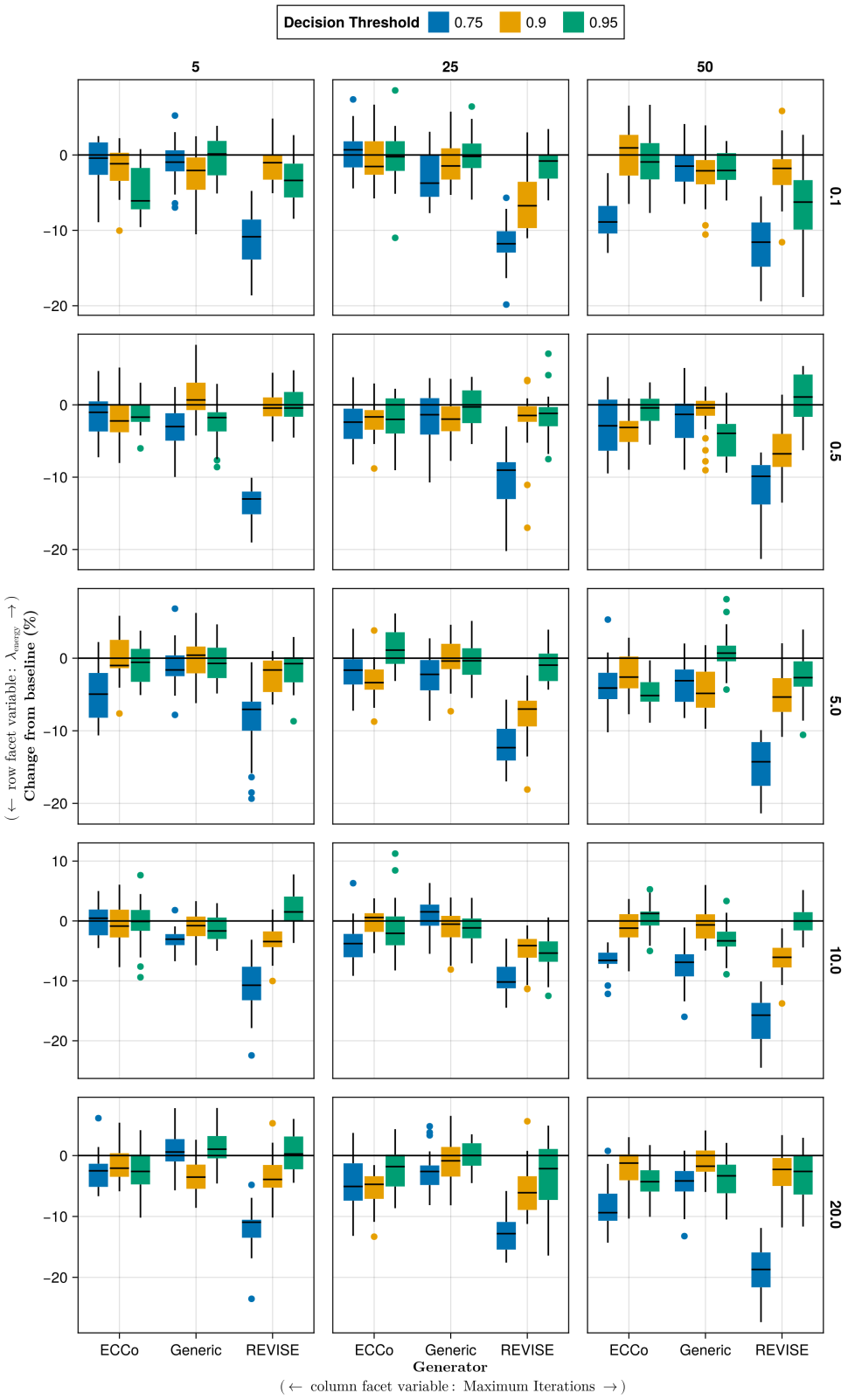

Figure 6: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

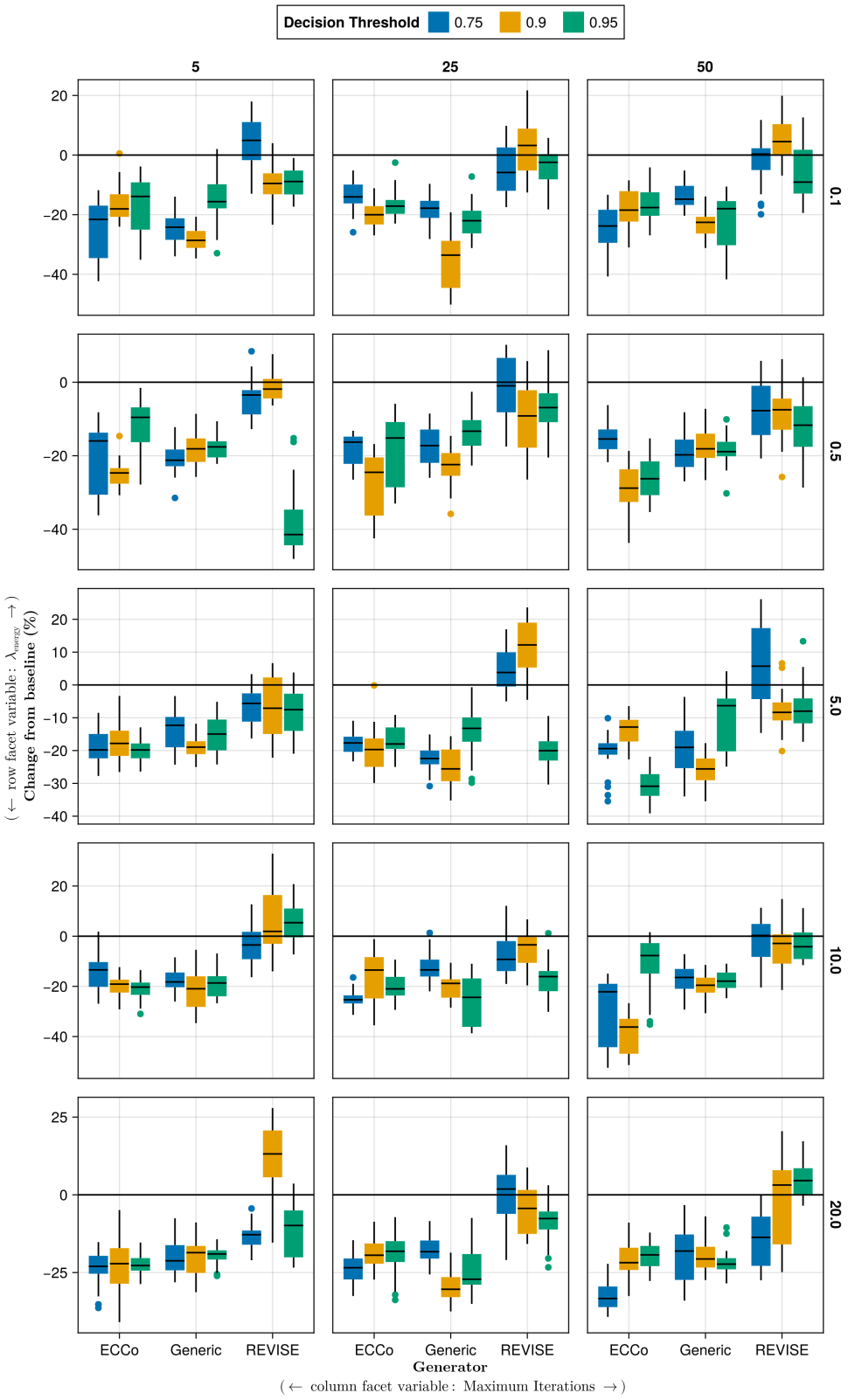

Figure 7: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

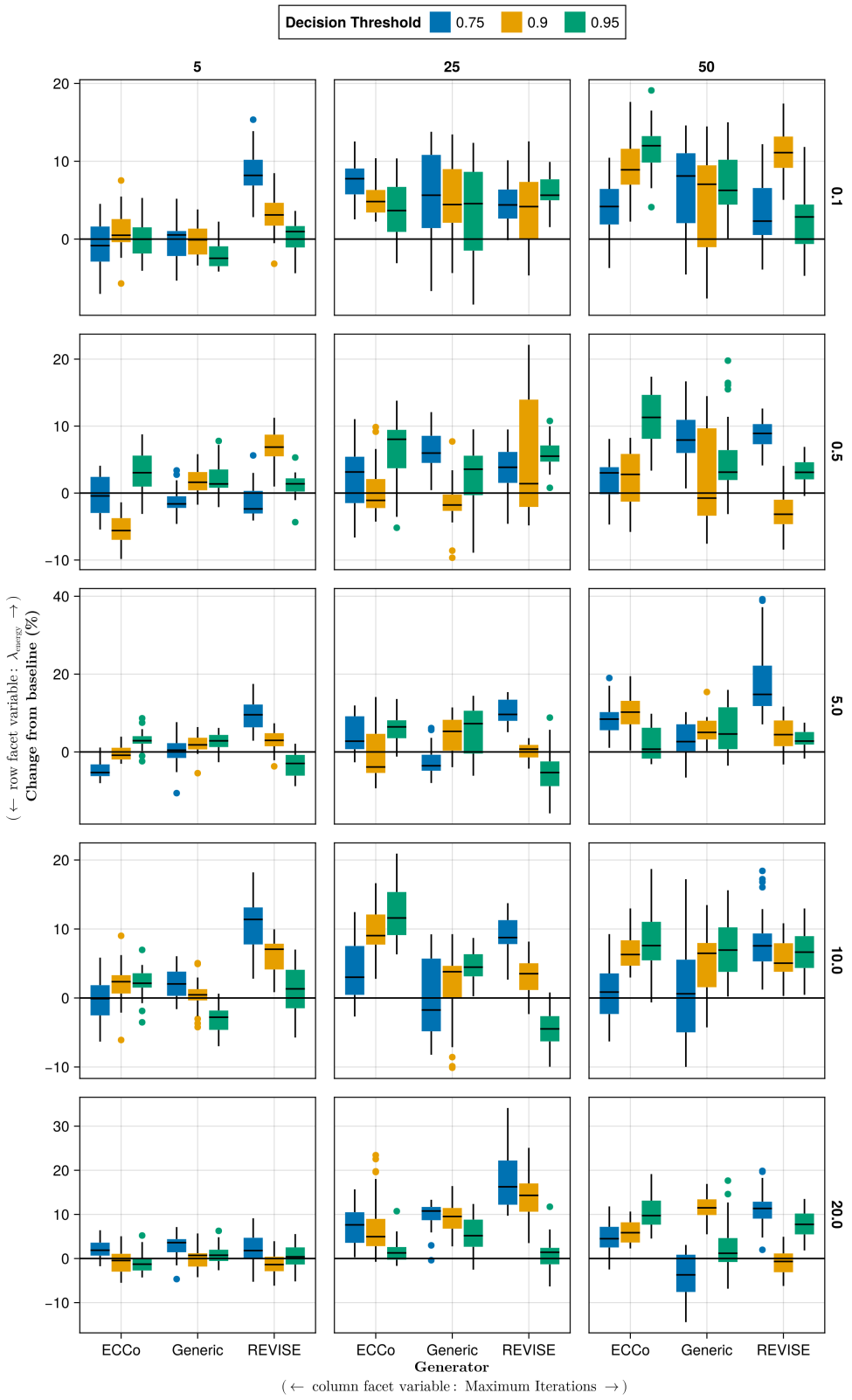

Figure 8: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

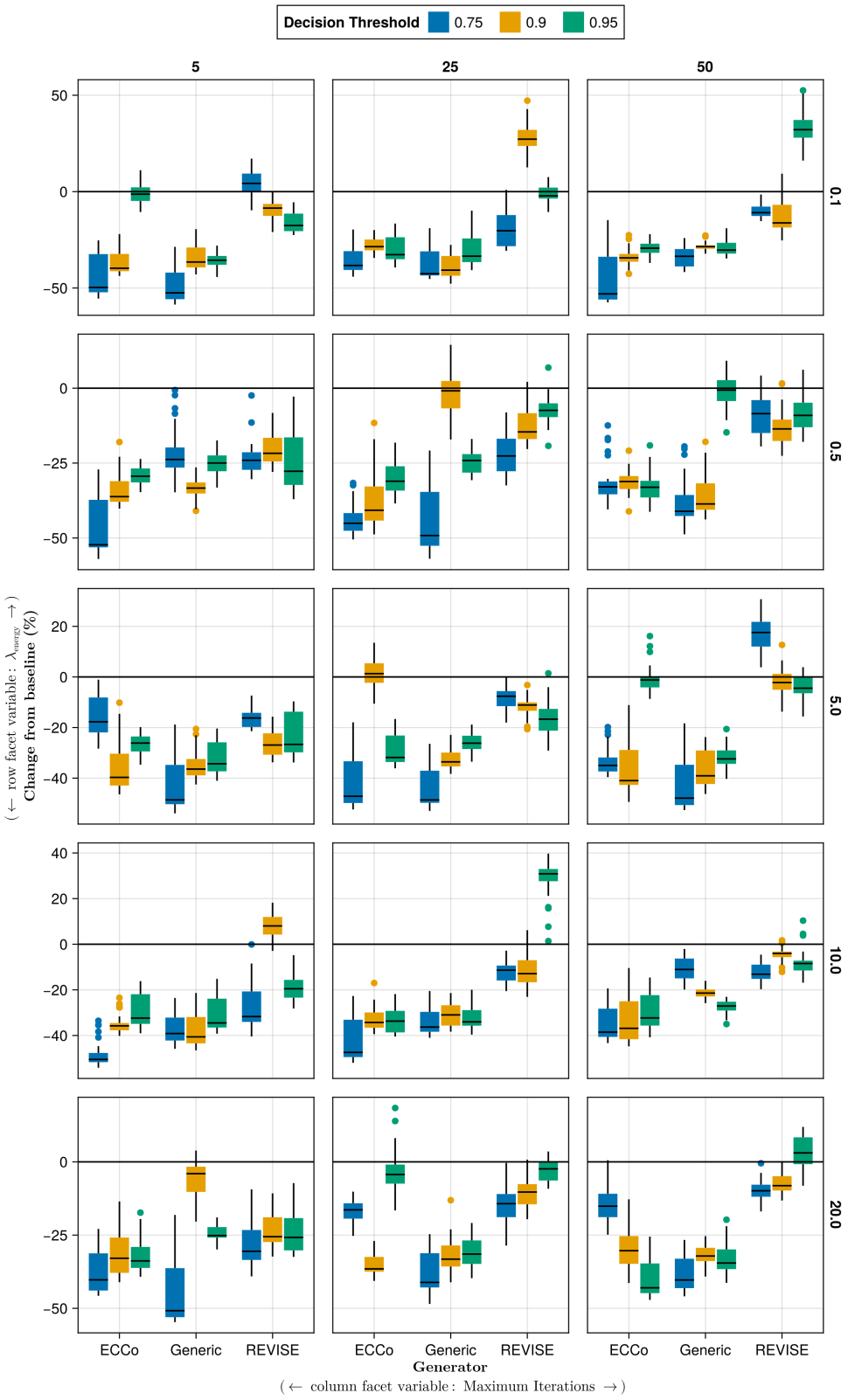

Figure 9: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

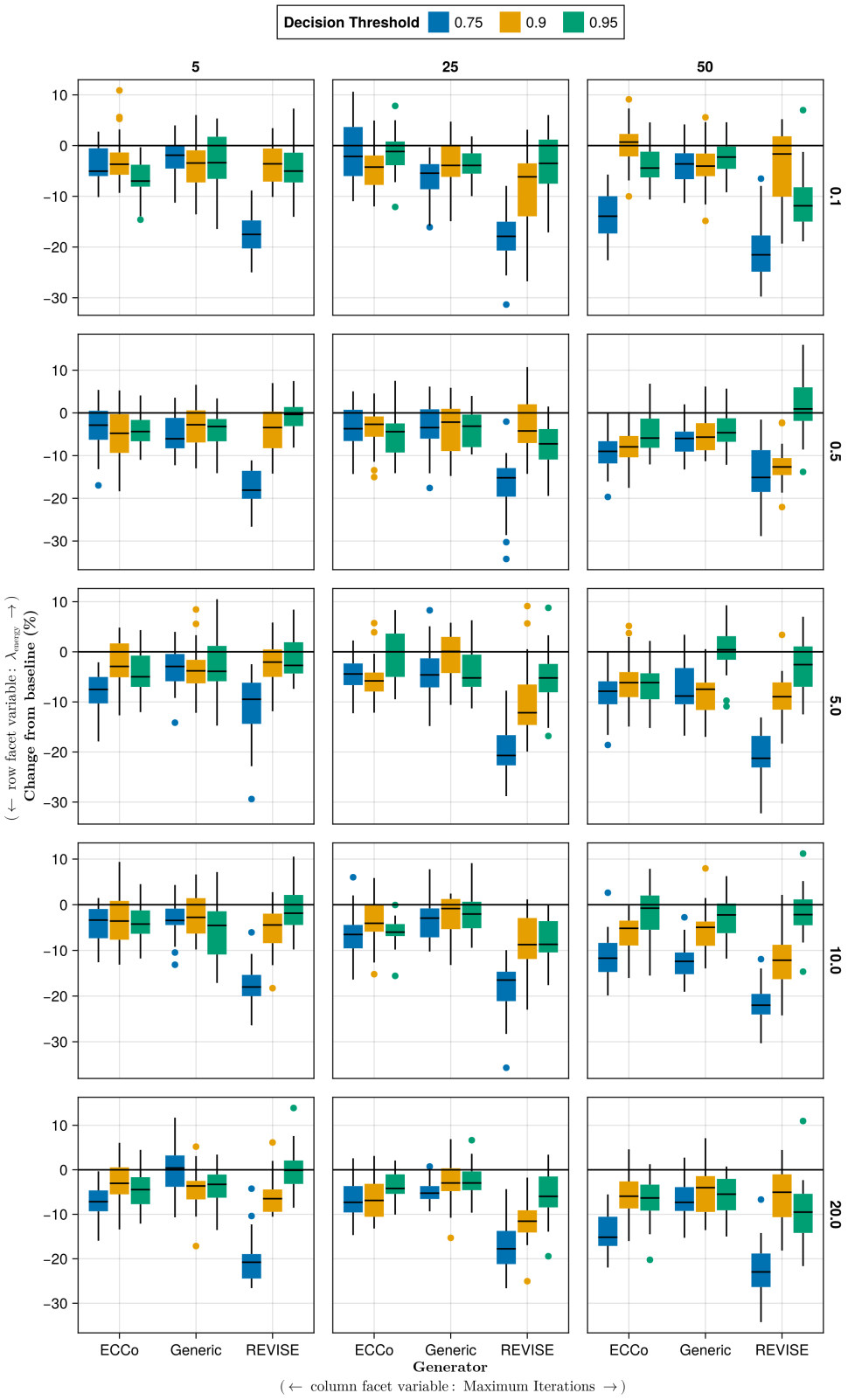

Figure 10: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

D.3.1 Predictive Performance

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

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	Se
Circ	Accuracy	Full	0.99	0.01
Circ	Accuracy	Vanilla	1.0	0.0
Circ	F1-score	Full	0.99	0.01
Circ	F1-score	Vanilla	1.0	0.0
LS	Accuracy	Full	1.0	0.01
LS	Accuracy	Vanilla	1.0	0.0
LS	F1-score	Full	1.0	0.01
LS	F1-score	Vanilla	1.0	0.0
Moon	Accuracy	Full	0.99	0.04
Moon	Accuracy	Vanilla	1.0	0.01
Moon	F1-score	Full	0.99	0.04
Moon	F1-score	Vanilla	1.0	0.01
OL	Accuracy	Full	0.91	0.02
OL	Accuracy	Vanilla	0.92	0.0
OL	F1-score	Full	0.91	0.02
OL	F1-score	Vanilla	0.92	0.0

D.3.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 11 to Figure 14.

D.3.3 Cost

The results with respect to the cost measure are shown in Figure 15 to Figure 18.

D.4 Other Parameters

The hyperparameter grid with other varying training parameters is shown in Note 7. The corresponding evaluation grid used for these experiments is shown in Note 8.

Note 7: 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 8: 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 6.

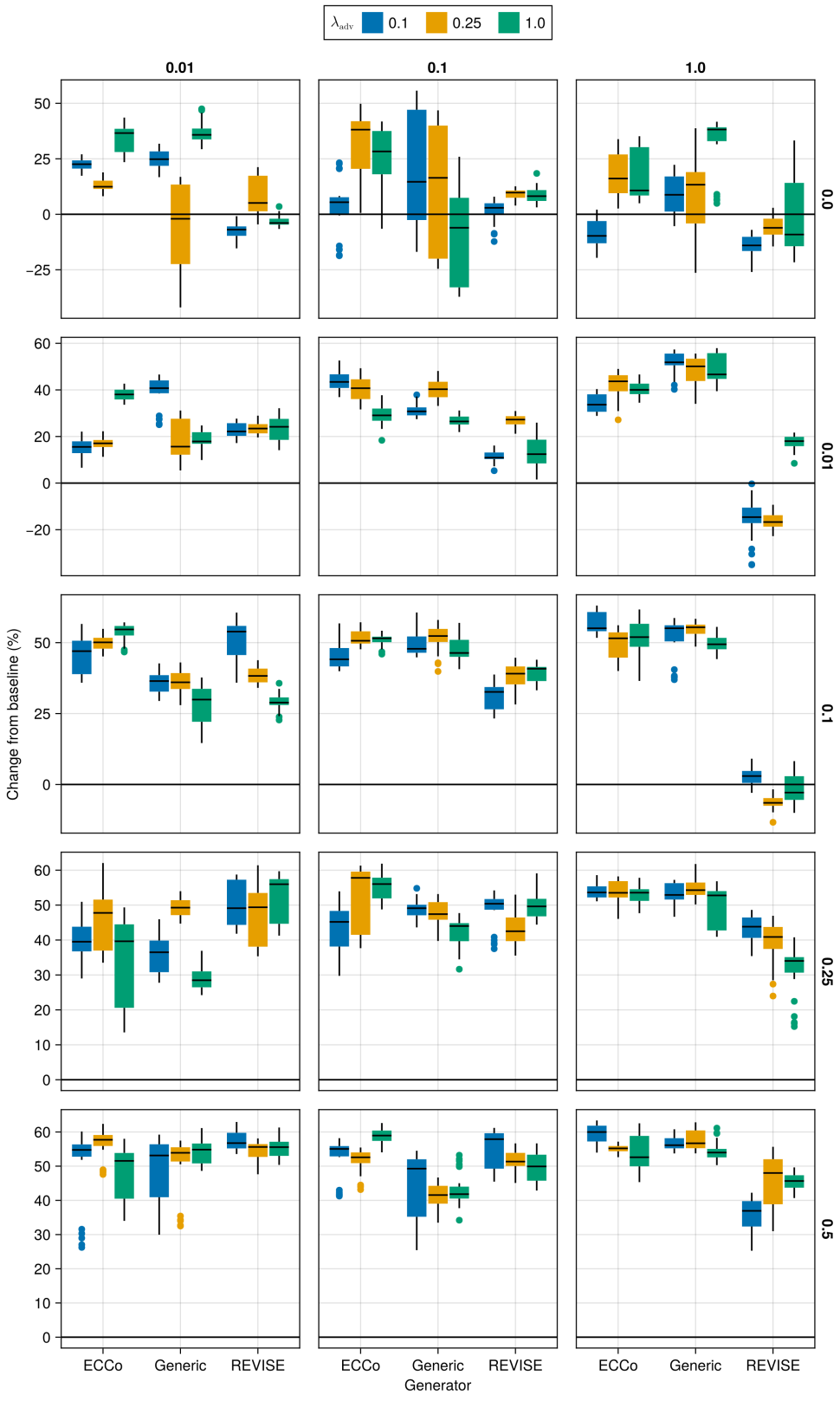

Figure 11: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

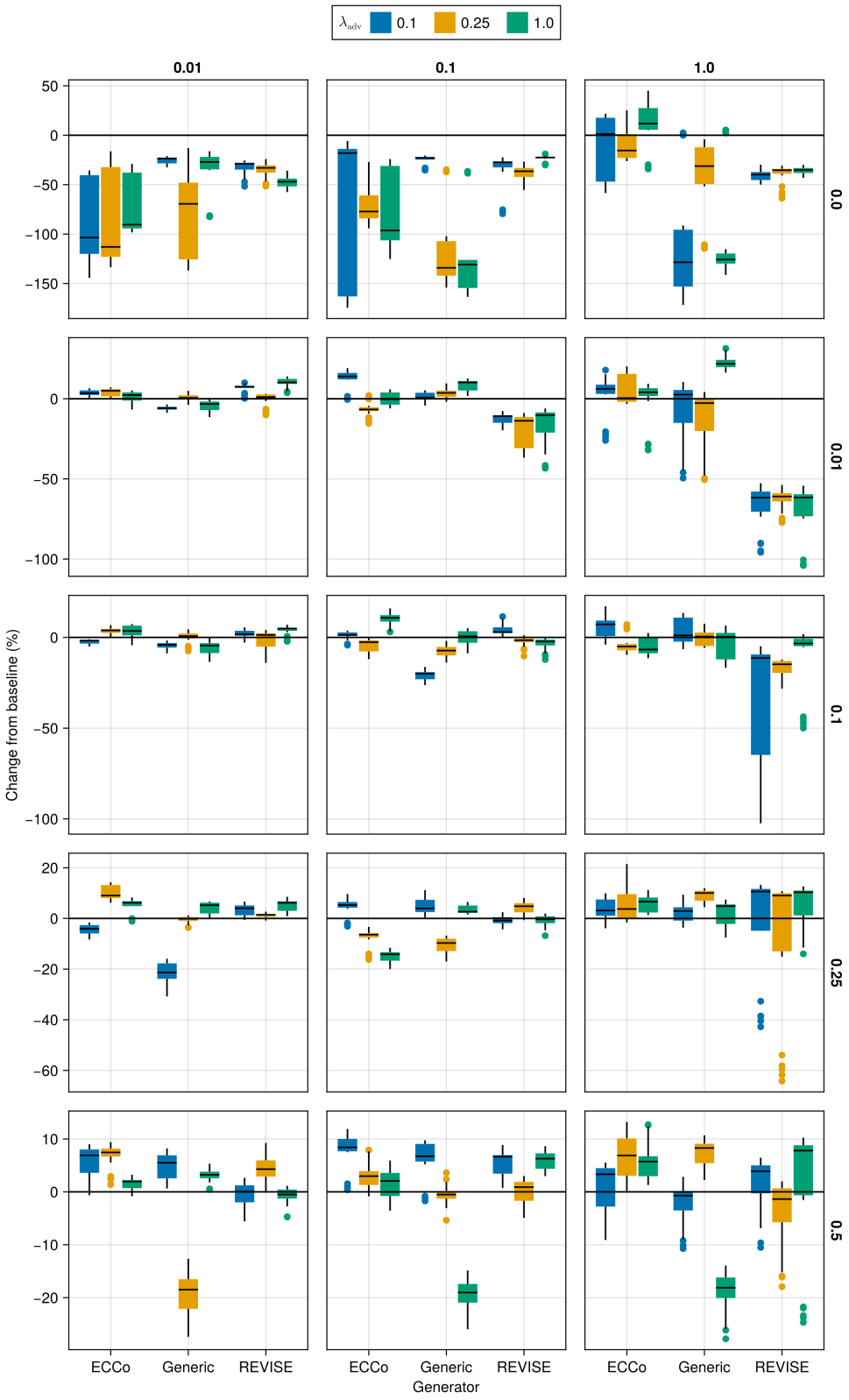

Figure 12: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

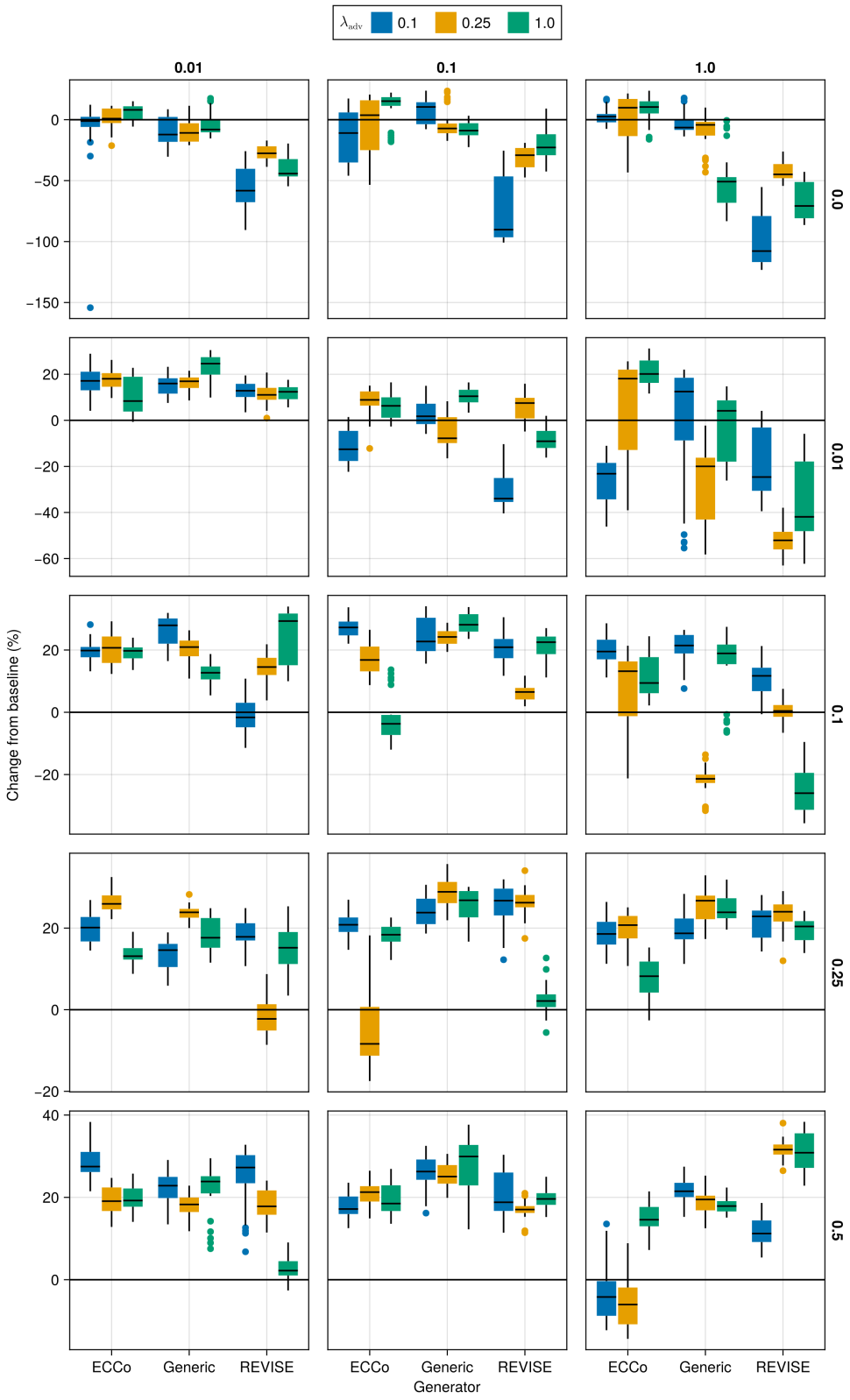

Figure 13: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

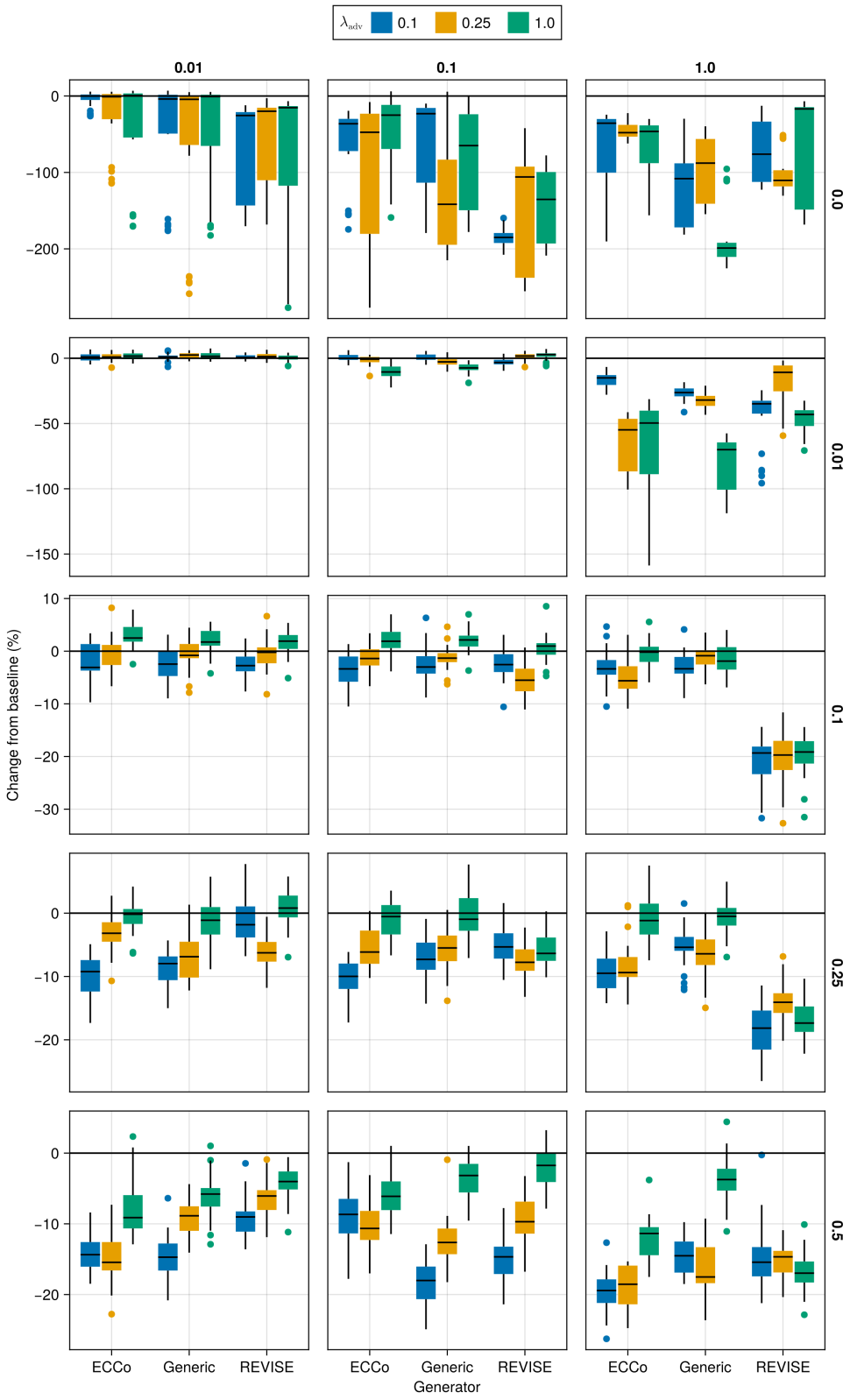

Figure 14: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

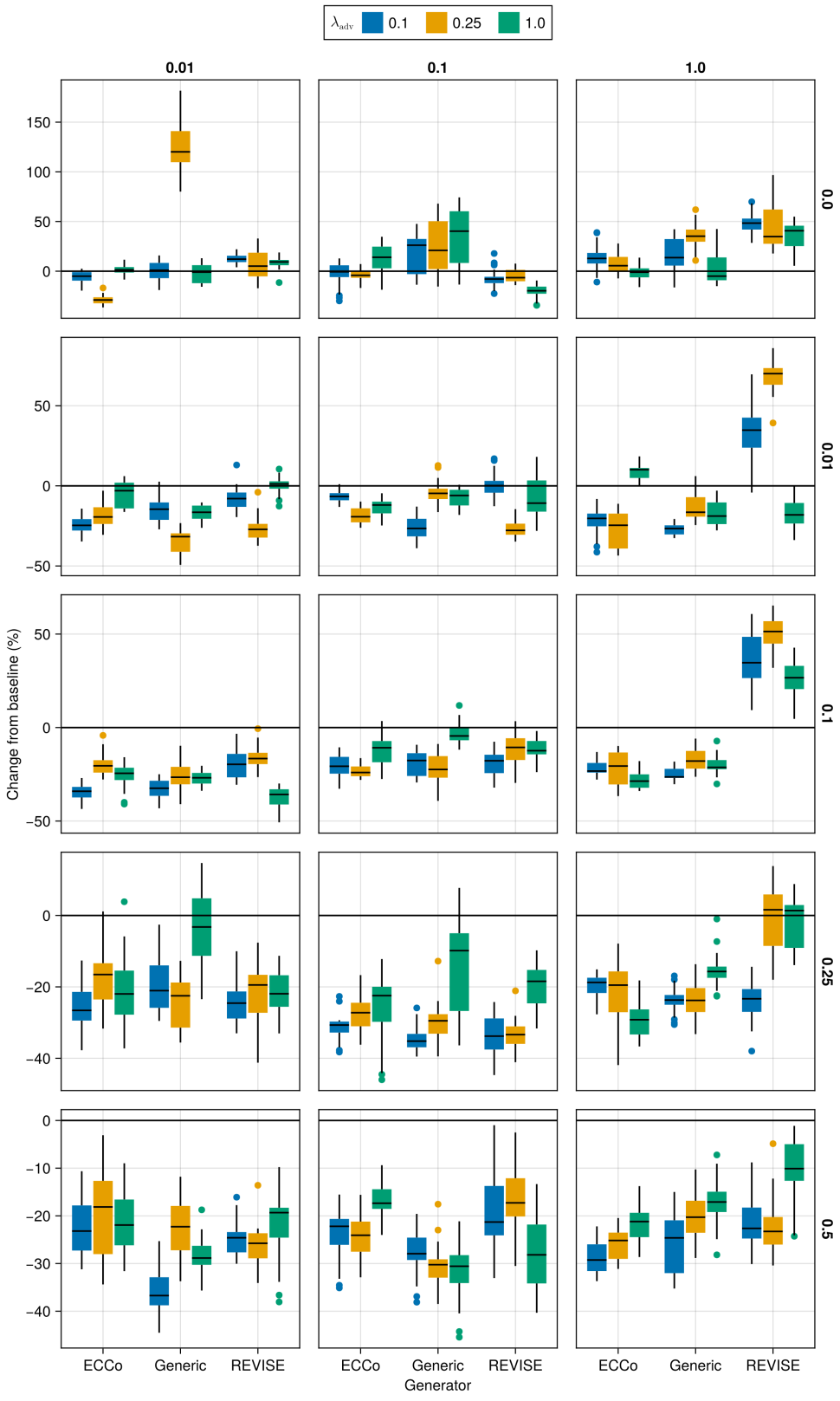

Figure 15: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

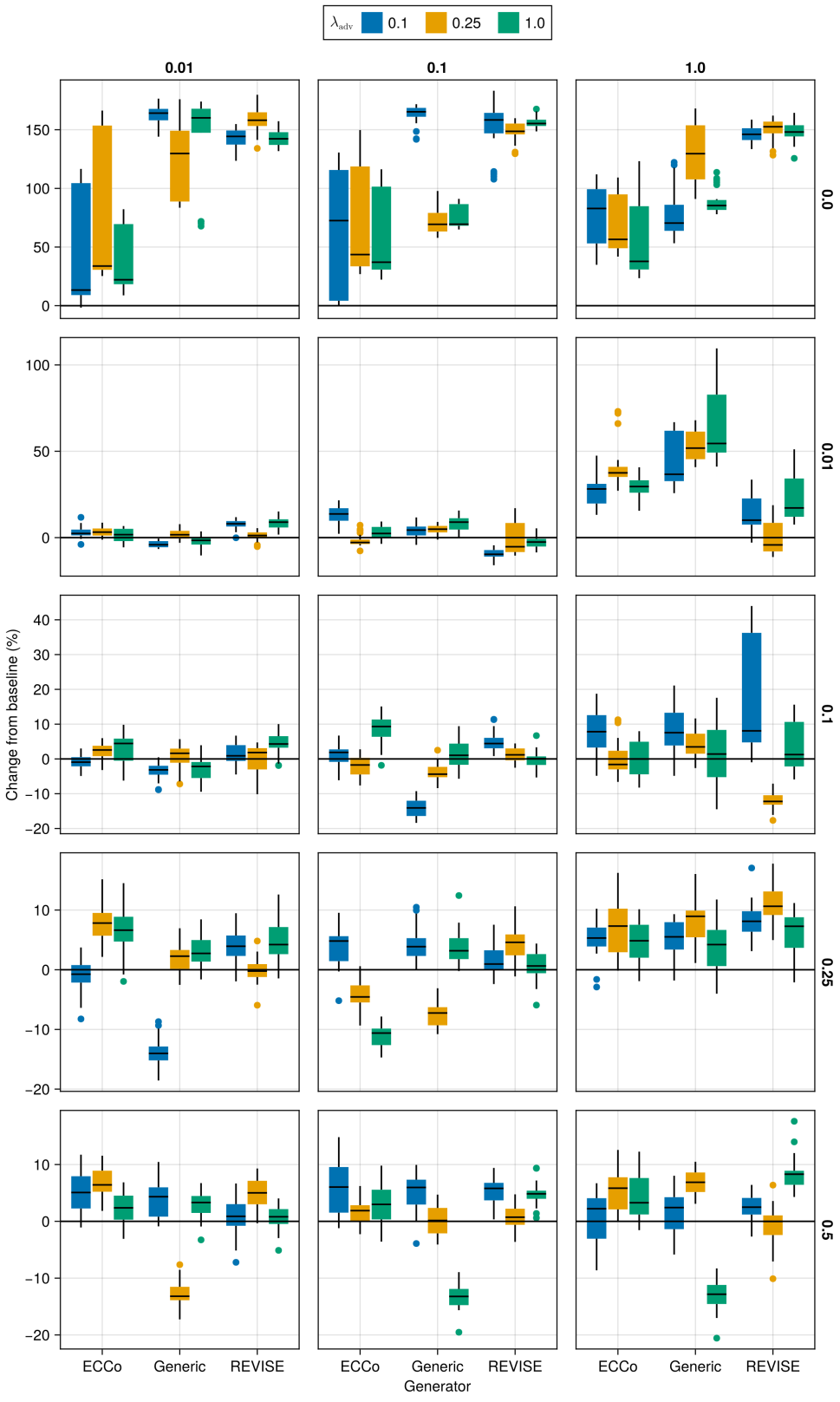

Figure 16: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

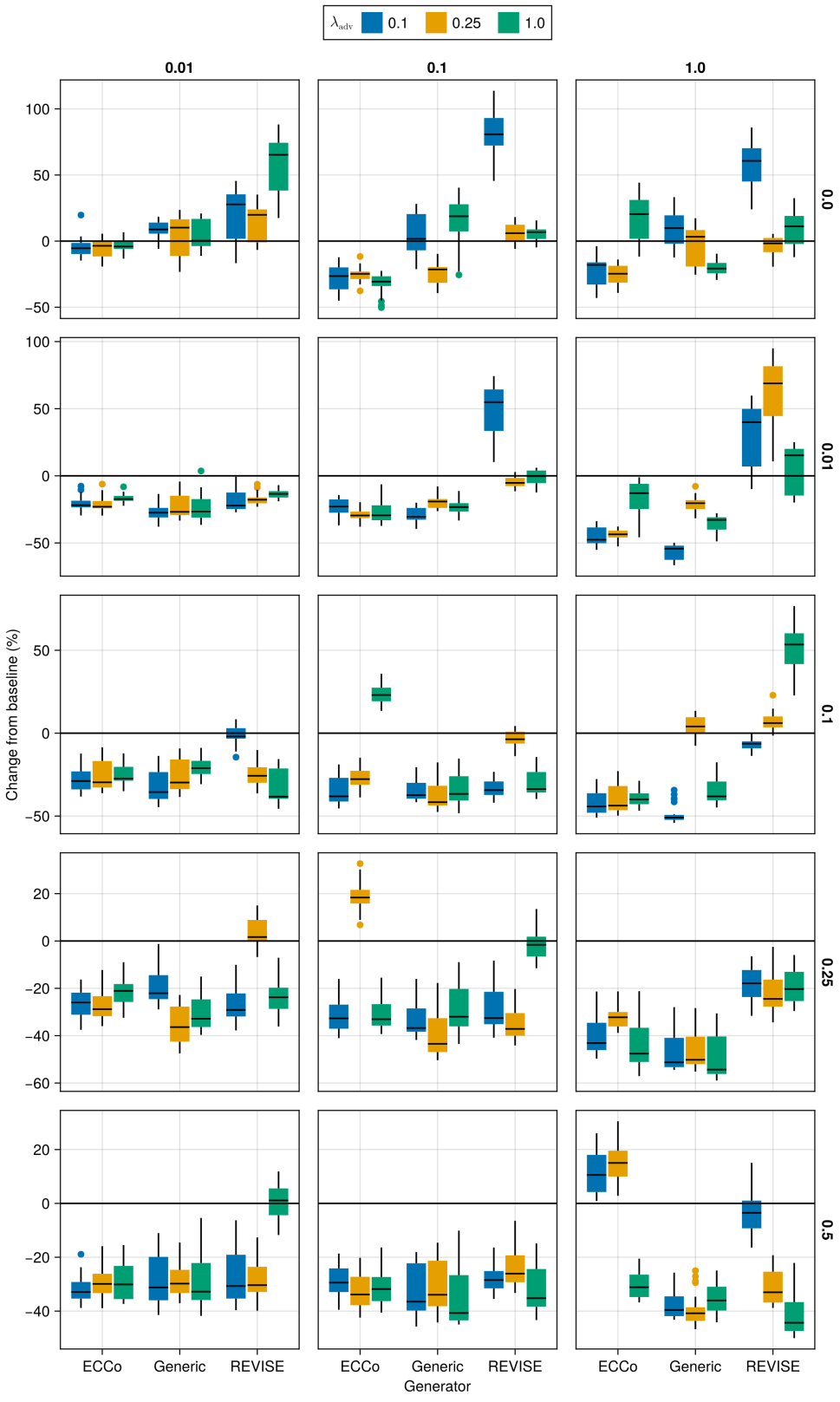

Figure 17: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

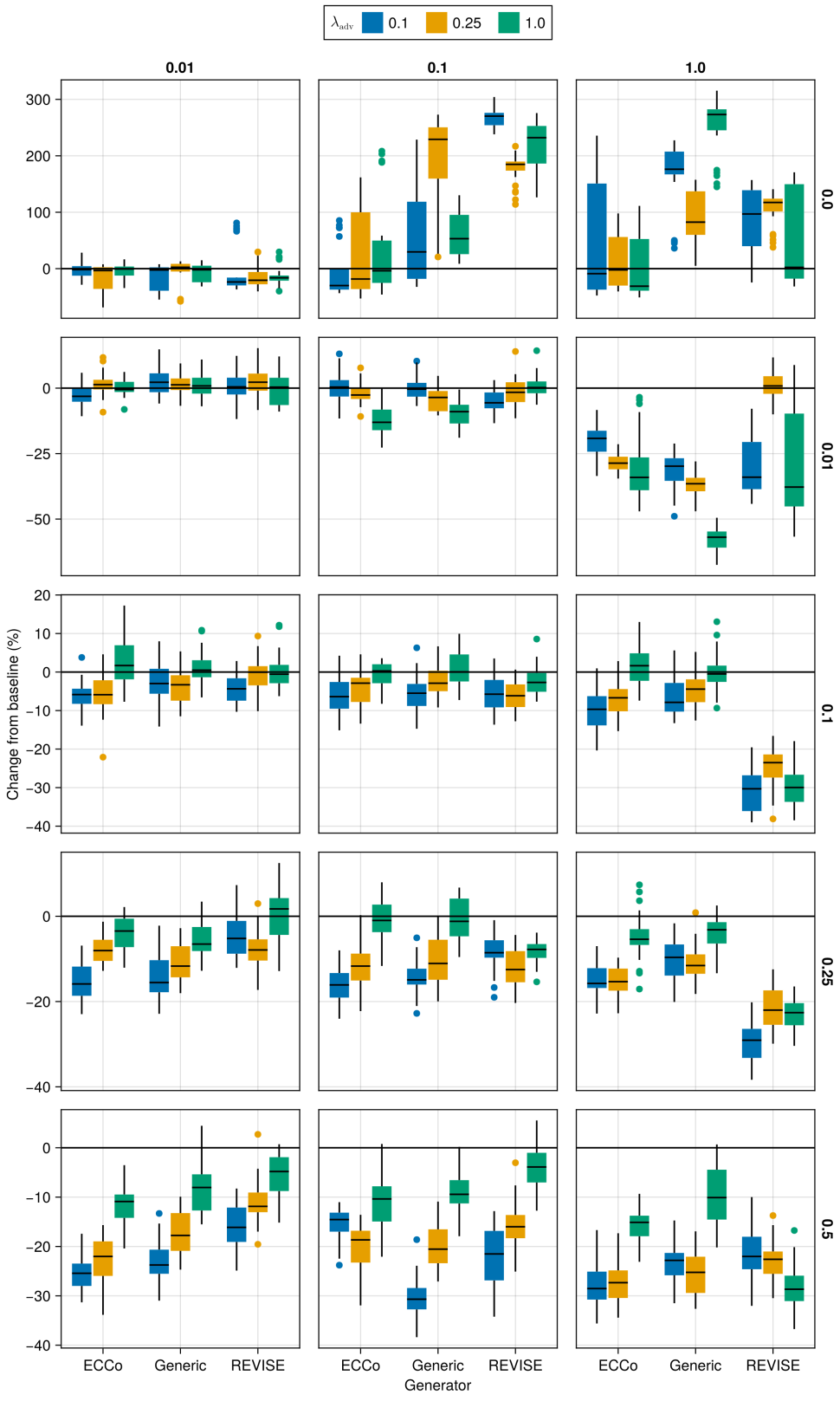

Figure 18: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

Table 6: Predictive performance measures by dataset and objective averaged across training-phase parameters (Note 7) and evaluation-phase parameters (Note 8).

Dataset	Variable	Objective	Mean	Se
Circ	Accuracy	Full	0.99	0.0
Circ	Accuracy	Vanilla	1.0	0.0
Circ	F1-score	Full	0.99	0.0
Circ	F1-score	Vanilla	1.0	0.0
LS	Accuracy	Full	1.0	0.0
LS	Accuracy	Vanilla	1.0	0.0
LS	F1-score	Full	1.0	0.0
LS	F1-score	Vanilla	1.0	0.0
Moon	Accuracy	Full	1.0	0.01
Moon	Accuracy	Vanilla	0.99	0.02
Moon	F1-score	Full	1.0	0.01
Moon	F1-score	Vanilla	0.99	0.02
OL	Accuracy	Full	0.91	0.01
OL	Accuracy	Vanilla	0.92	0.0
OL	F1-score	Full	0.91	0.01
OL	F1-score	Vanilla	0.92	0.0

D.4.2 Plausibility

The results with respect to the plausibility measure are shown in Figure 19 to Figure 22.

D.4.3 Cost

The results with respect to the cost measure are shown in Figure 23 to Figure 26.

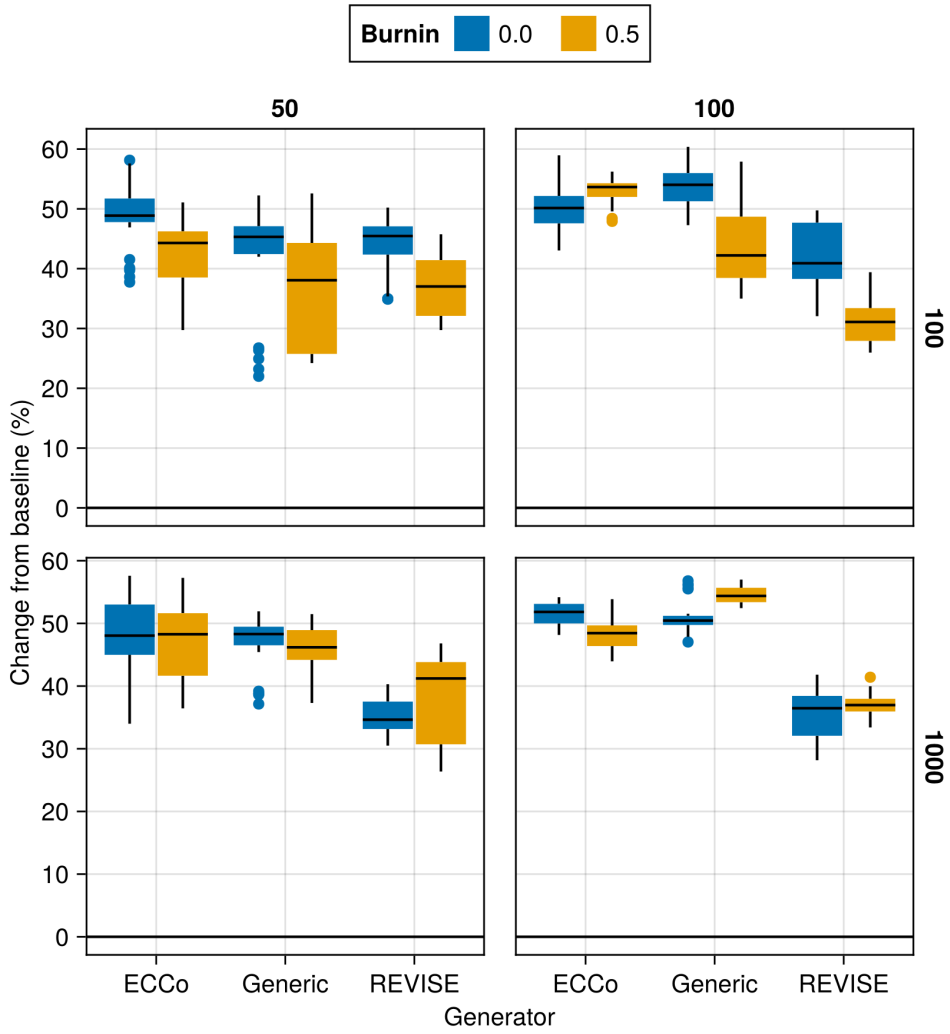

Figure 19: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

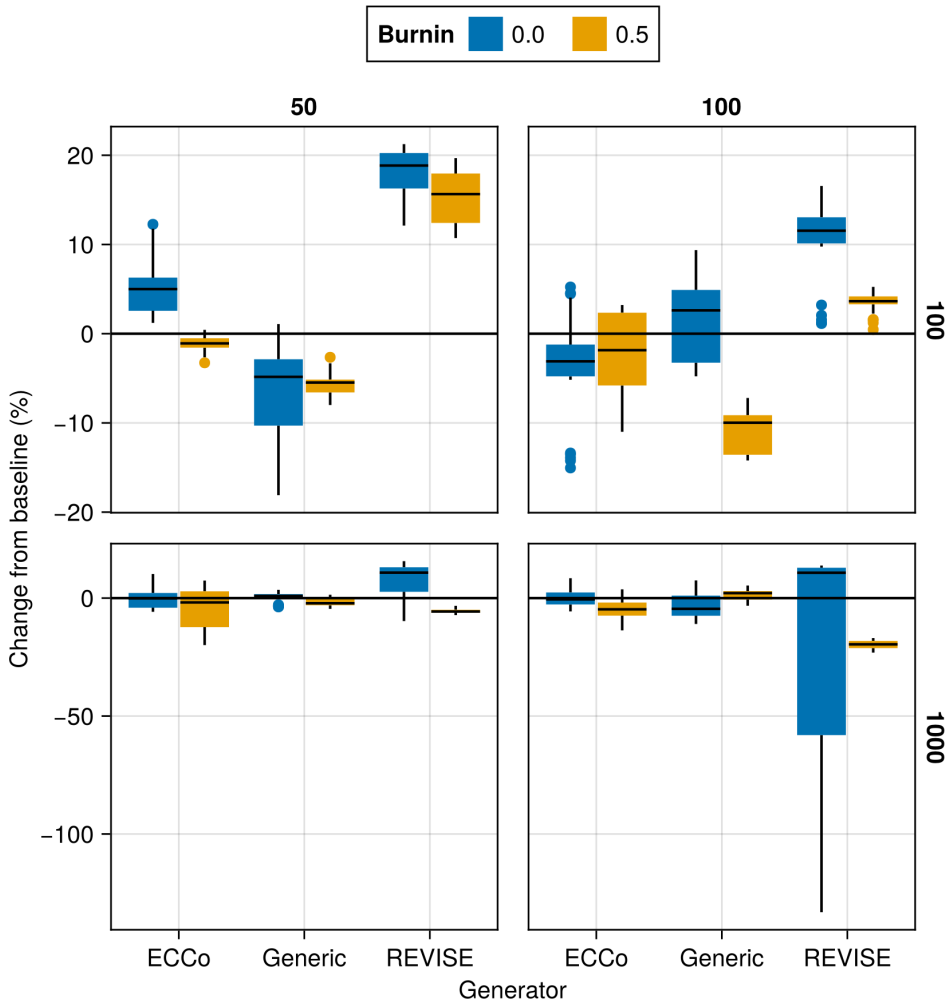

Figure 20: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

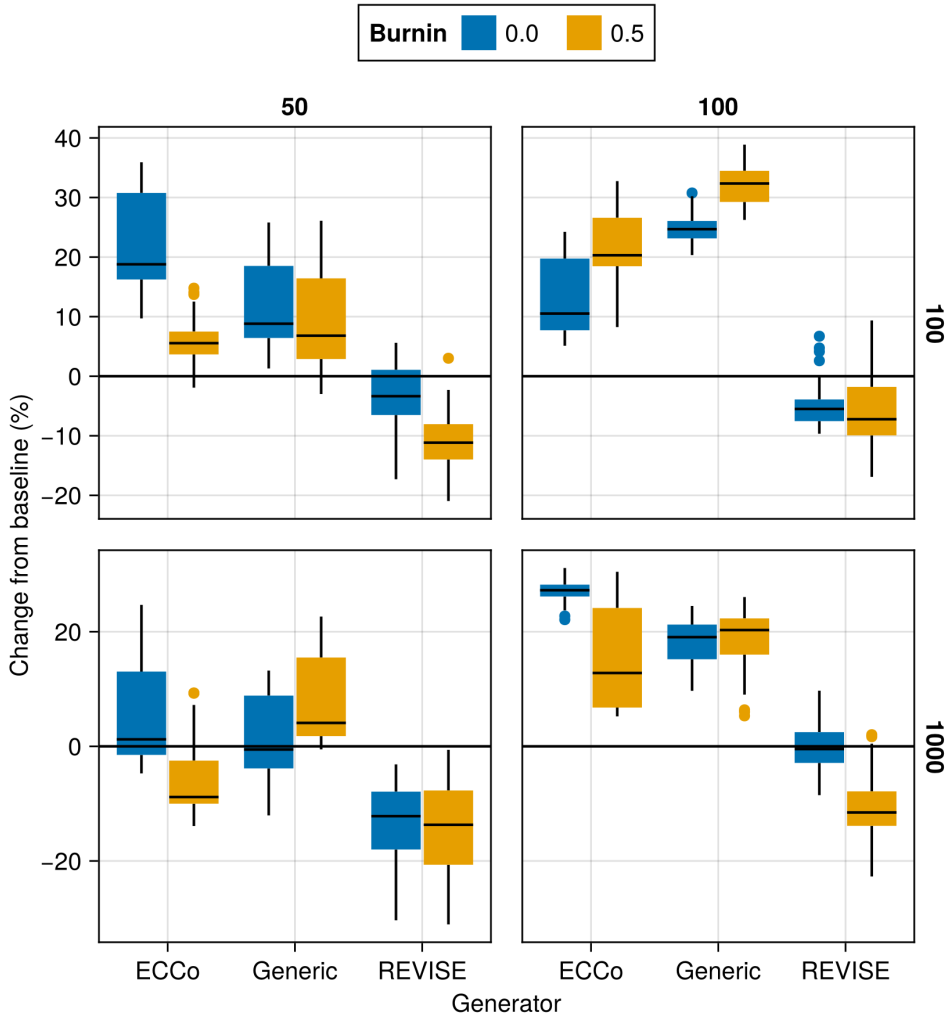

Figure 21: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

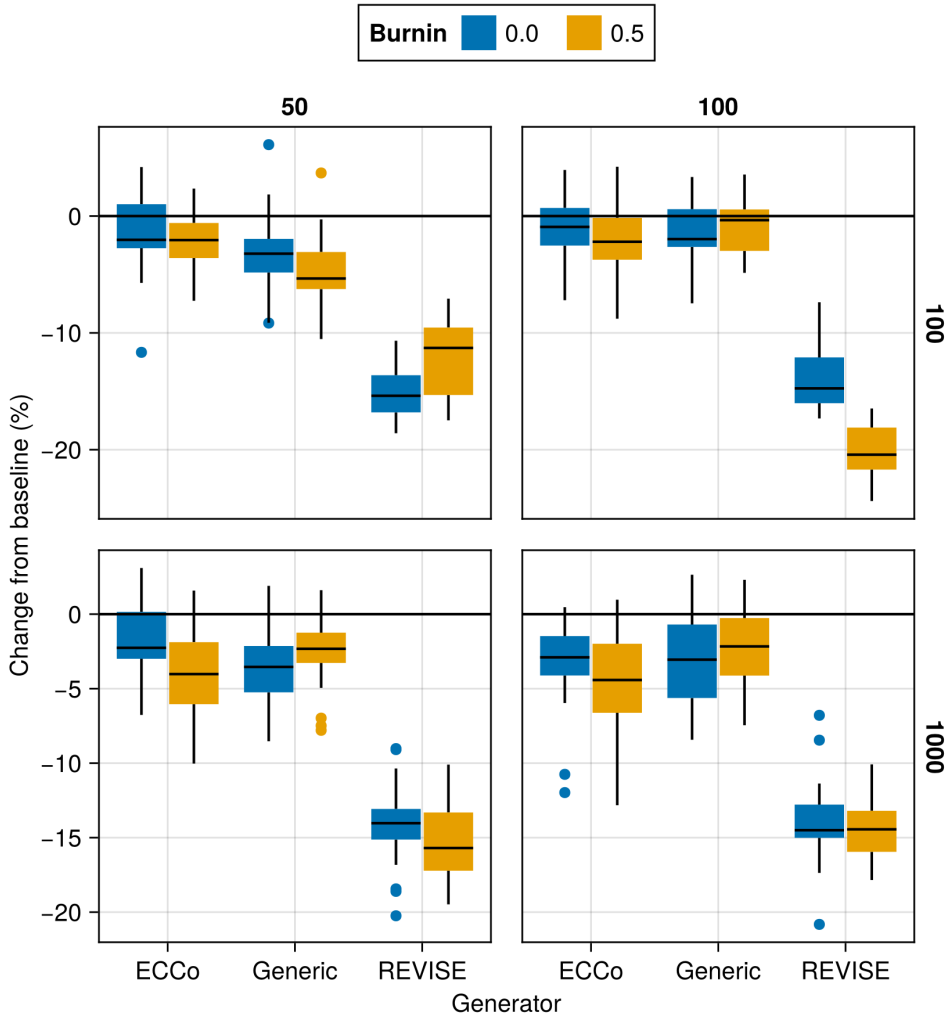

Figure 22: Average outcomes for the plausibility measure across hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

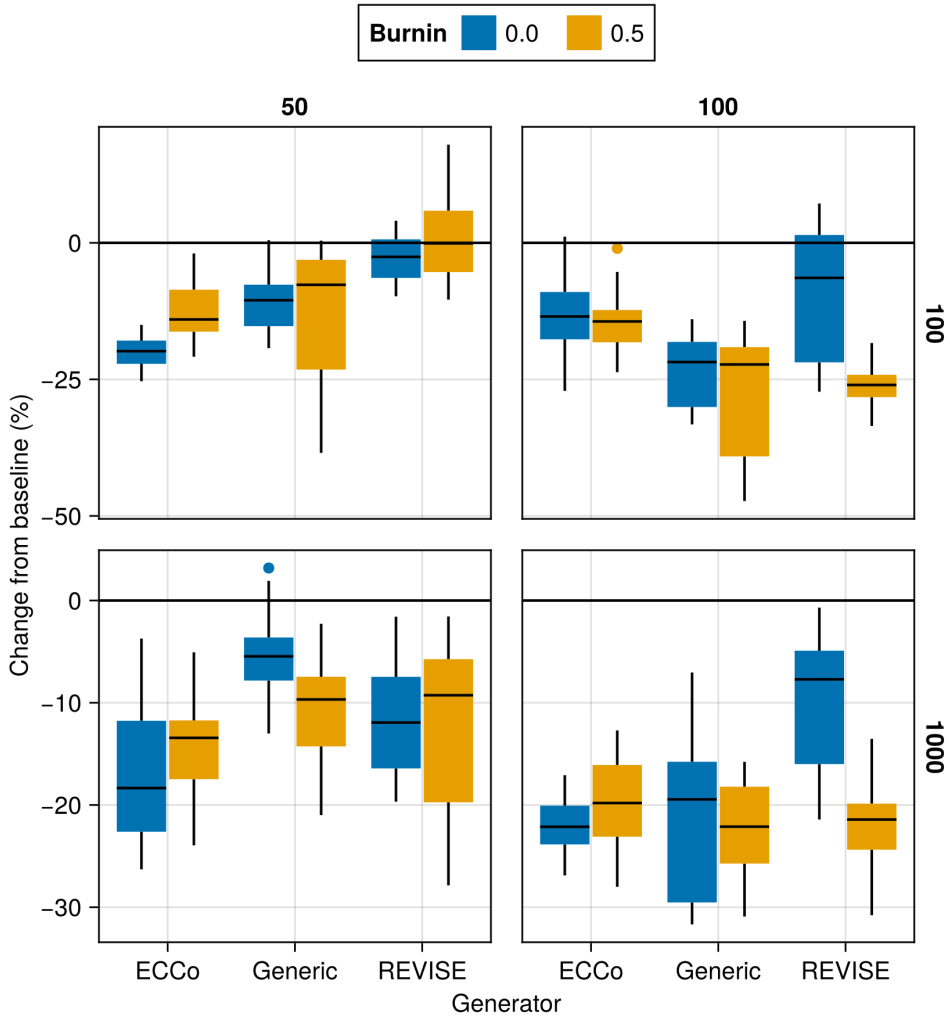

Figure 23: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

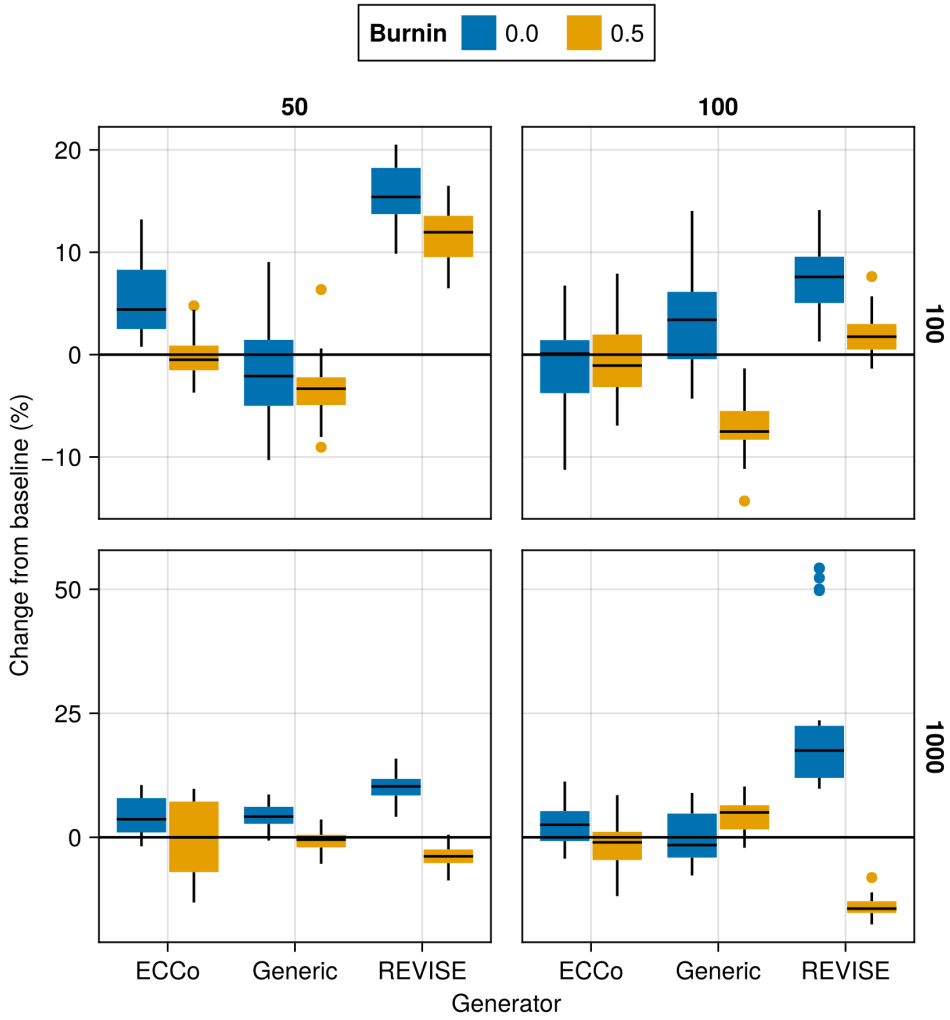

Figure 24: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

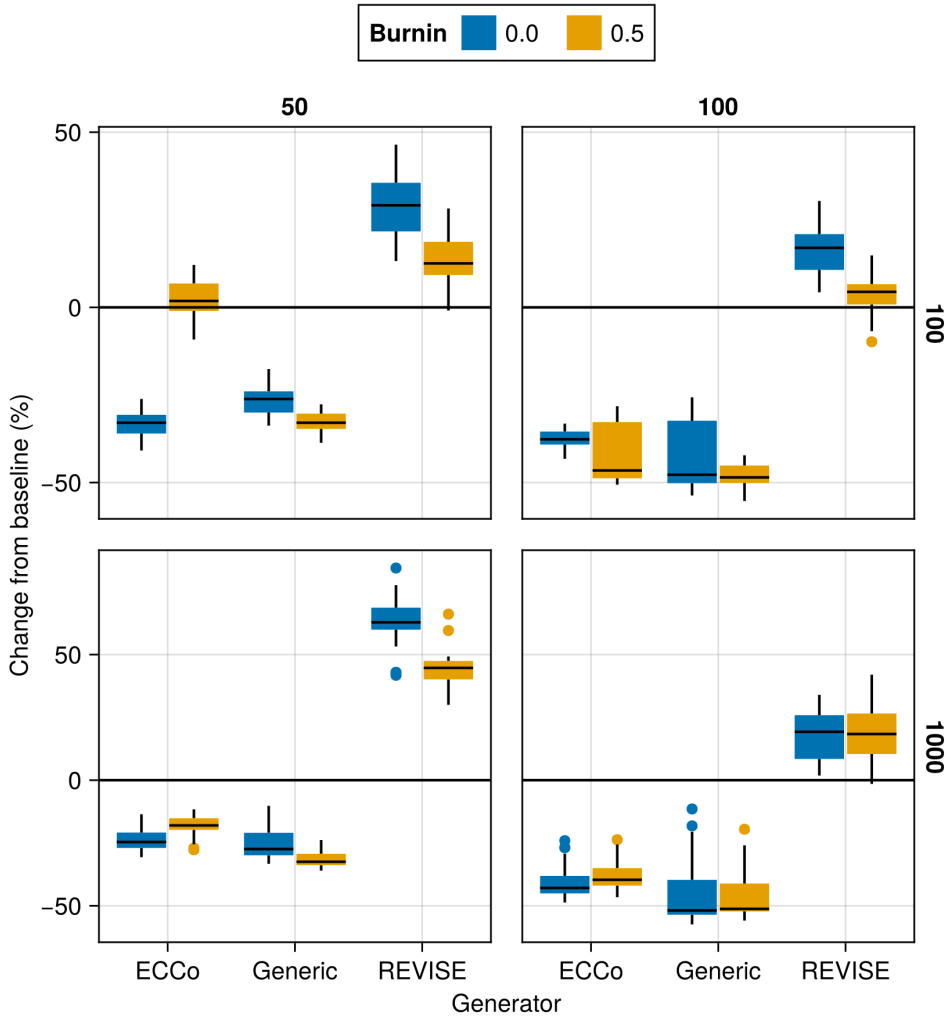

Figure 25: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

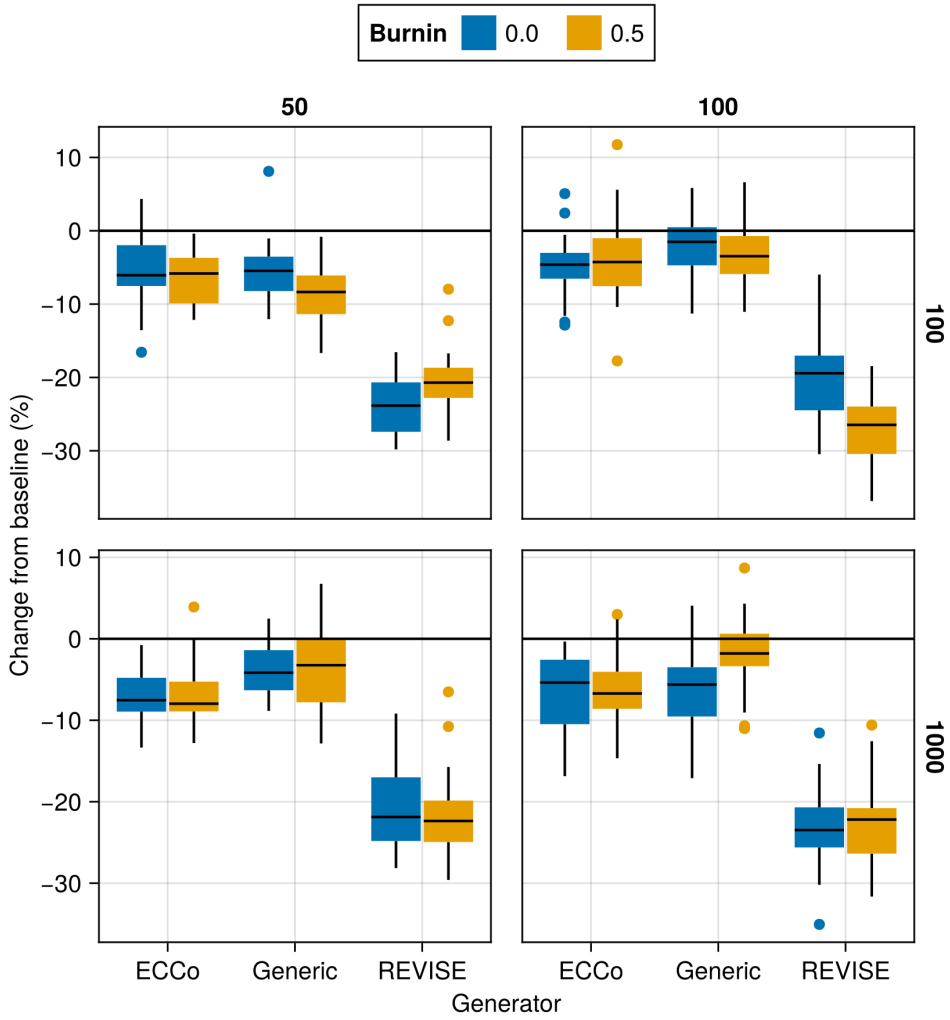

Figure 26: Average outcomes for the cost measure across hyperparameters. This shows the % change from the baseline model for the distance-based cost metric (Wachter, Mittelstadt, and Russell 2017). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). 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 1 in Section C. Detailed results for each data set are shown in Figure 27 to Figure 44. From Table 1, 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 36 to Figure 44). 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 45 to Figure 53). For the given low decision threshold, we find that the learning rate has no discernable impact on the proportion of mature counterfactuals (Figure 54 to Figure 62). 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 51) 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.

E.1 Key Parameters

The hyperparameter grid for tuning key parameters is shown in Note 9. The corresponding evaluation grid used for these experiments is shown in Note 10.

Note 9: 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 10: 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 27 to Figure 35.

E.1.2 Proportion of Mature CE

The results with respect to the proportion of mature counterfactuals in each epoch are shown in Figure 36 to Figure 44.

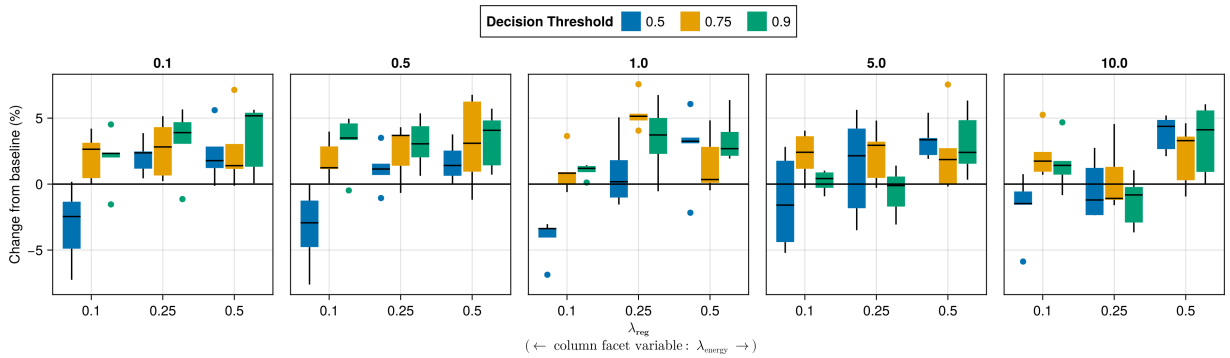

Figure 27: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Adult.

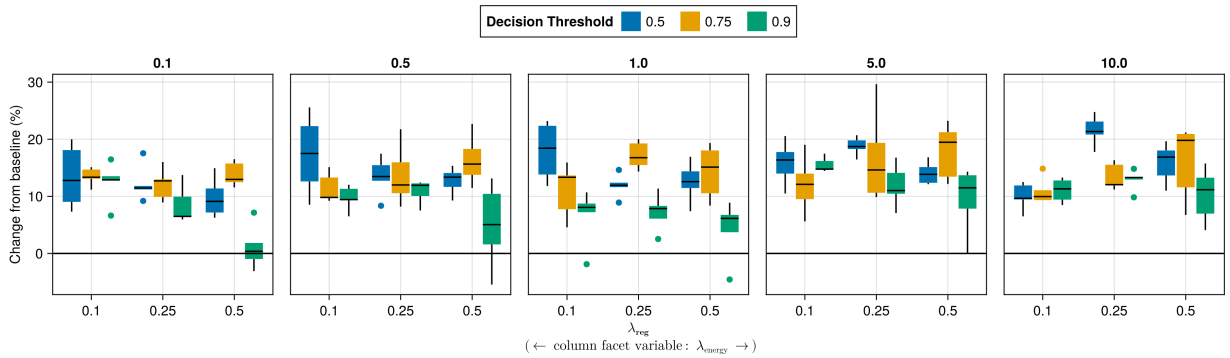

Figure 28: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: California Housing.

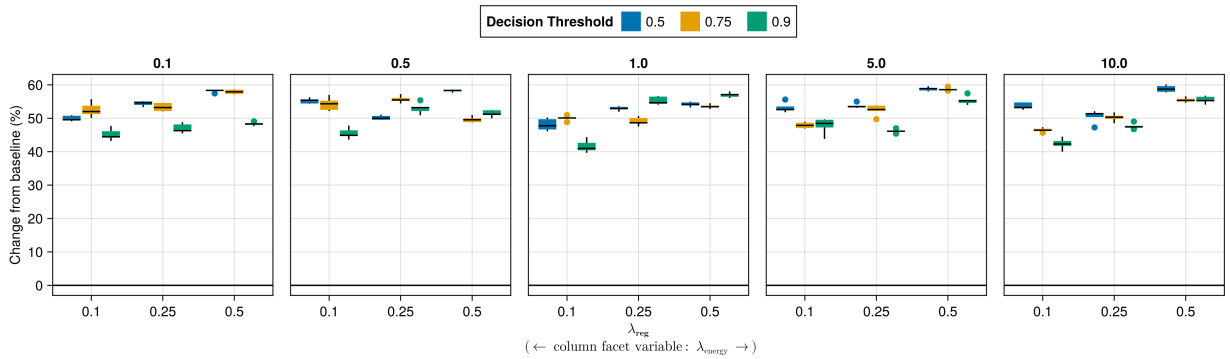

Figure 29: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

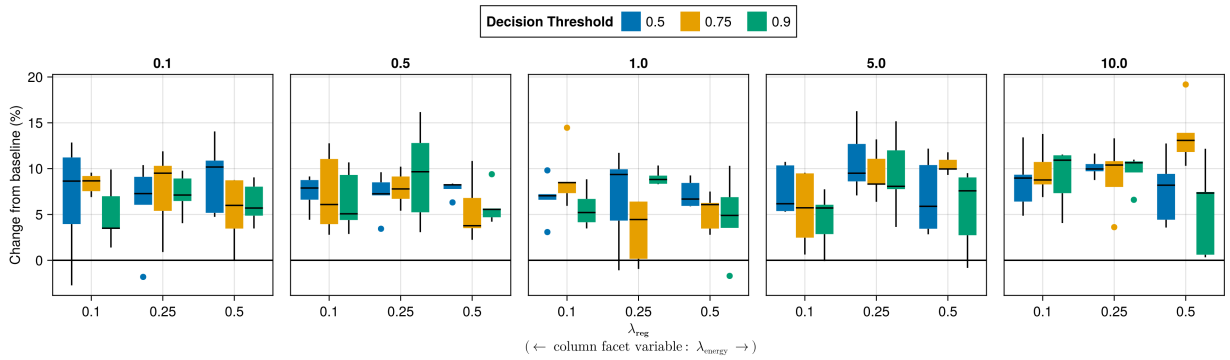

Figure 30: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Credit.

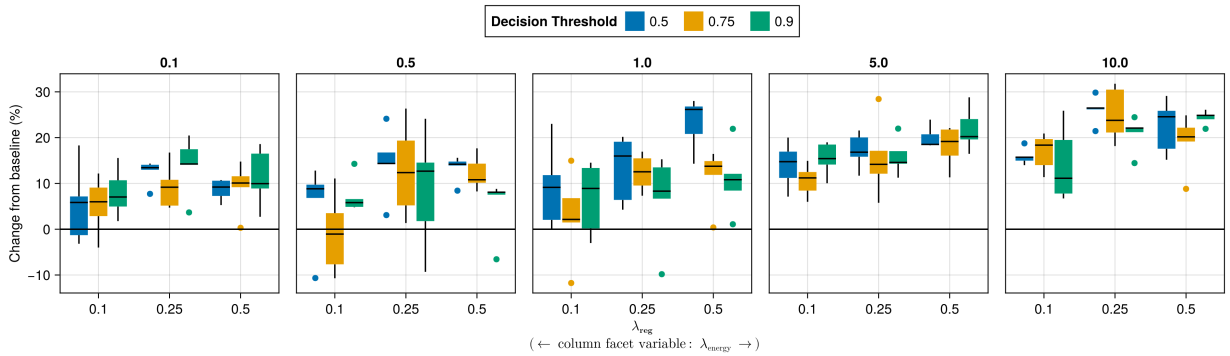

Figure 31: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: GMSC.

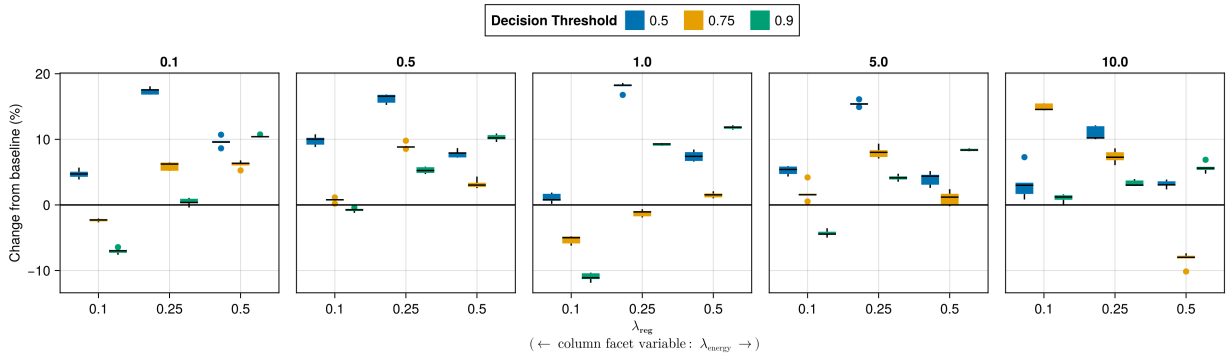

Figure 32: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

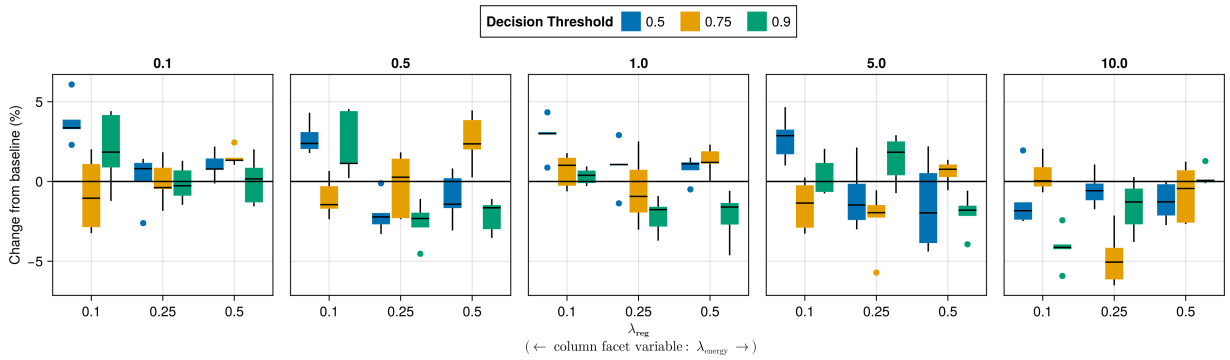

Figure 33: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: MNIST.

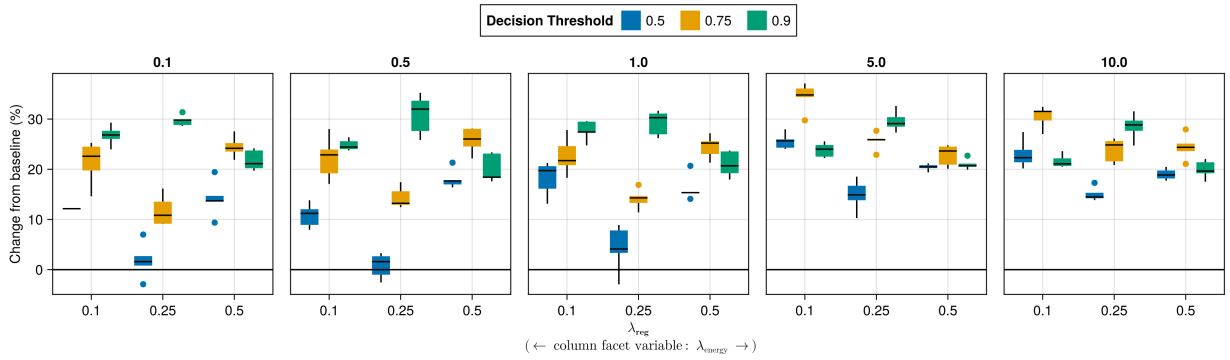

Figure 34: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

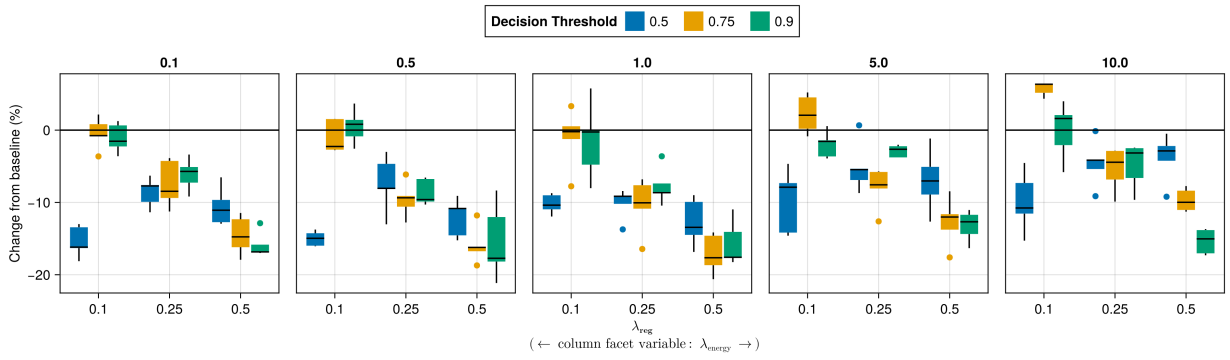

Figure 35: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

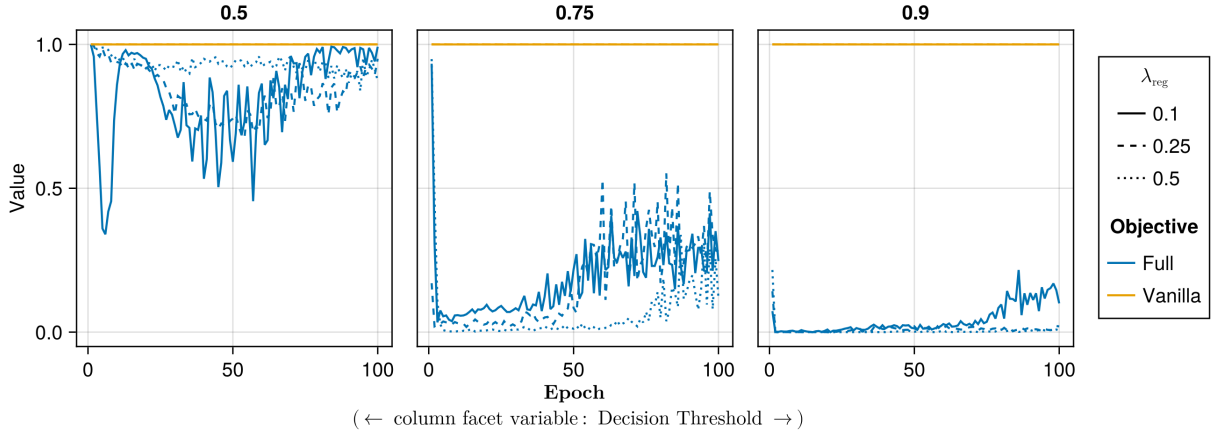

Figure 36: Proportion of mature counterfactuals in each epoch. Data: Adult.

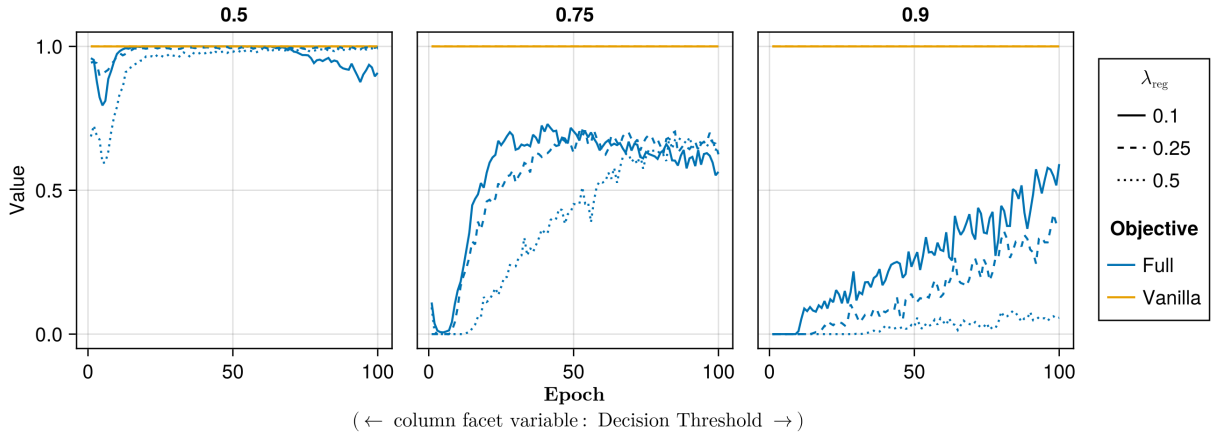

Figure 37: Proportion of mature counterfactuals in each epoch. Data: California Housing.

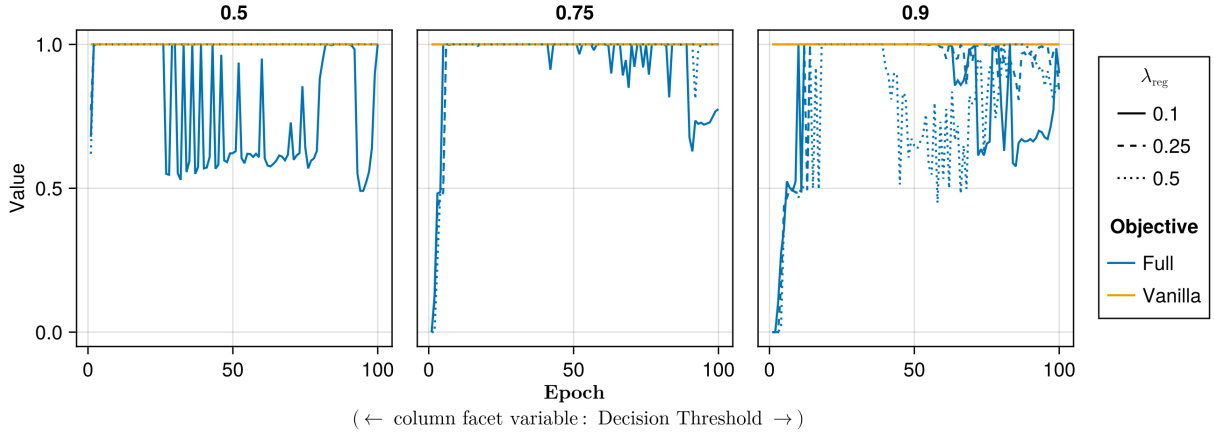

Figure 38: Proportion of mature counterfactuals in each epoch. Data: Circles.

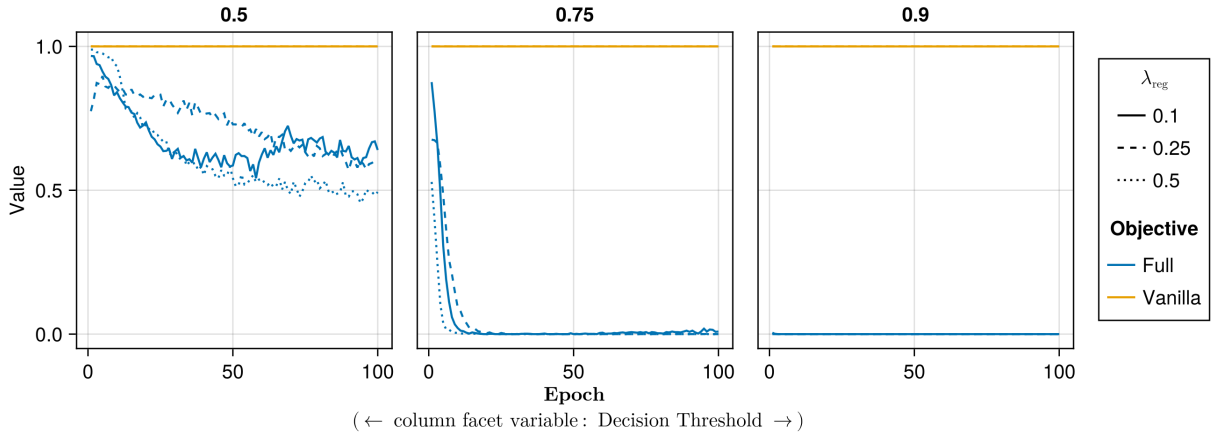

Figure 39: Proportion of mature counterfactuals in each epoch. Data: Credit.

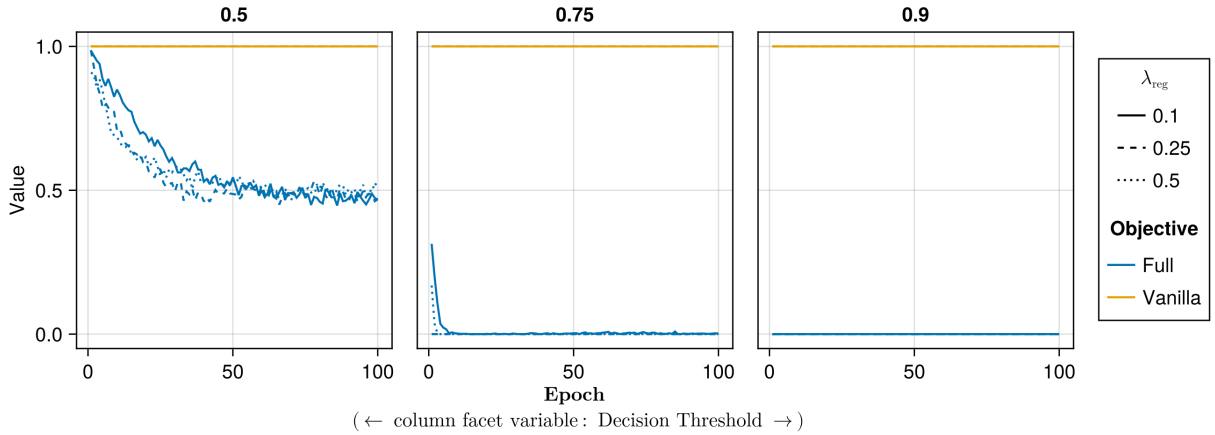

Figure 40: Proportion of mature counterfactuals in each epoch. Data: GMSC.

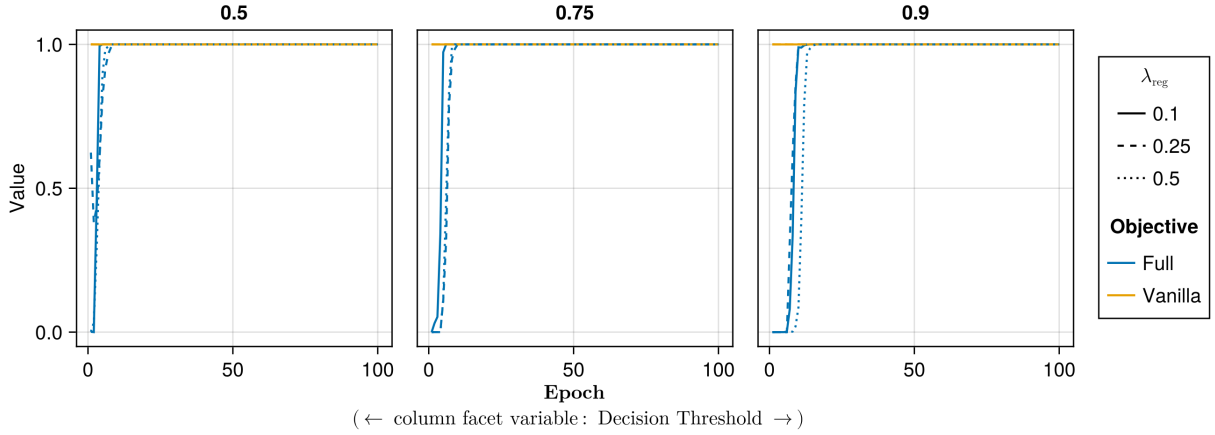

Figure 41: Proportion of mature counterfactuals in each epoch. Data: Linearly Separable.

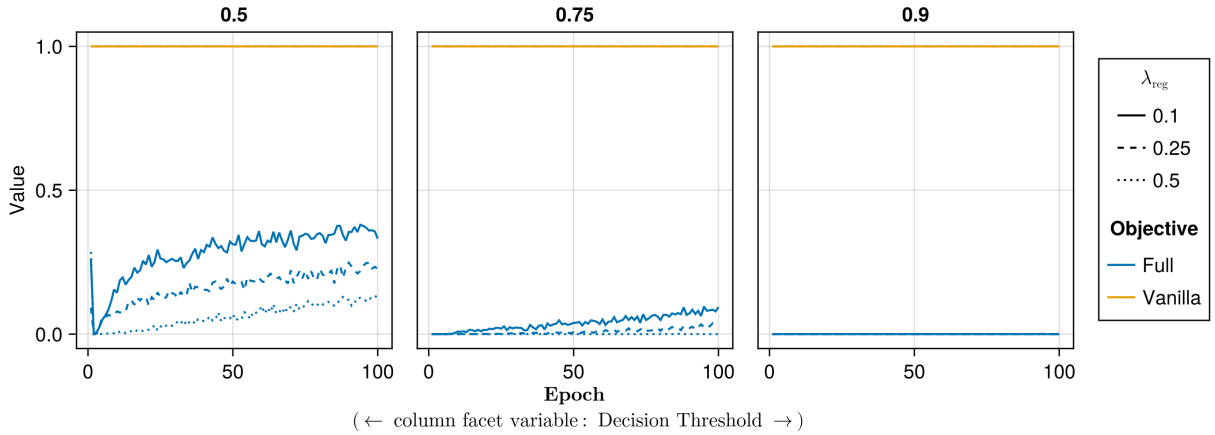

Figure 42: Proportion of mature counterfactuals in each epoch. Data: MNIST.

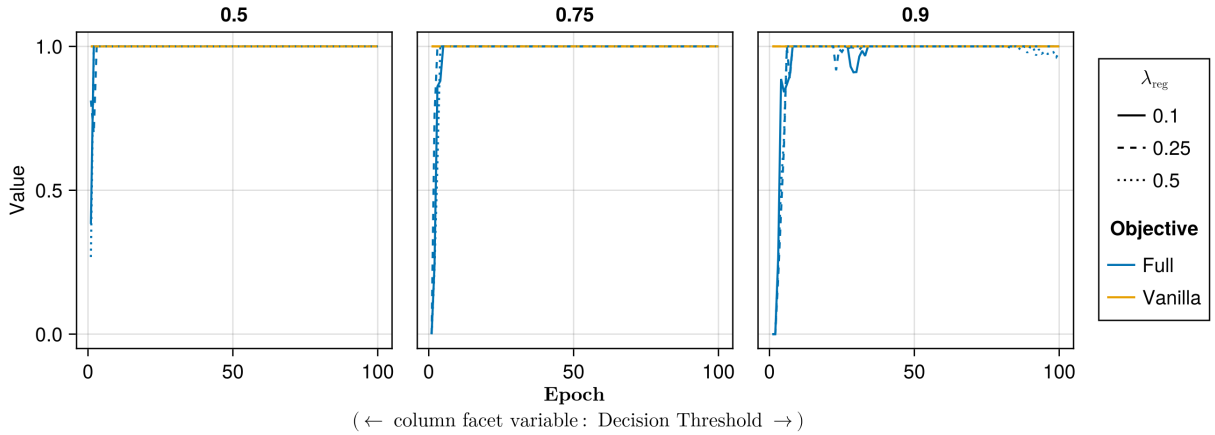

Figure 43: Proportion of mature counterfactuals in each epoch. Data: Moons.

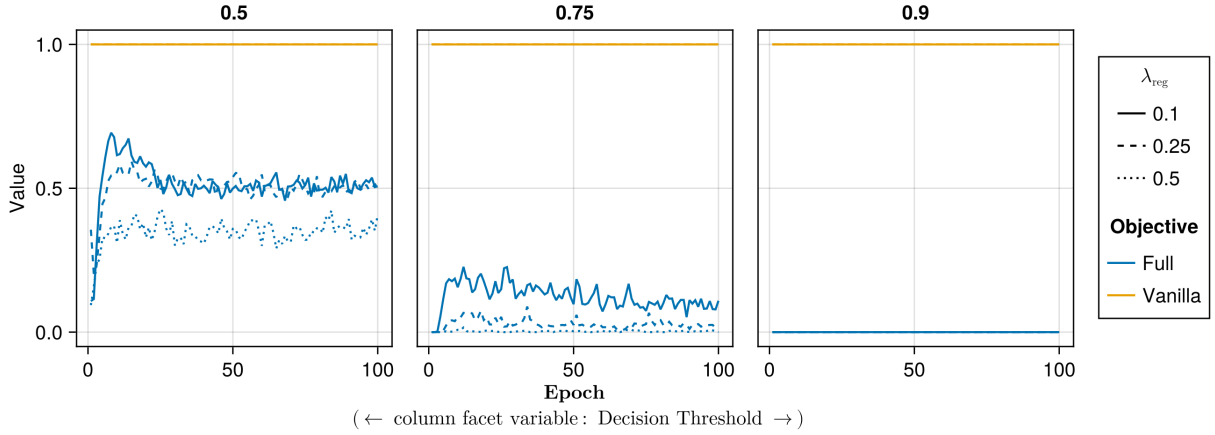

Figure 44: Proportion of mature counterfactuals in each epoch. Data: Overlapping.

E.2 Learning Rate

The hyperparameter grid for tuning the learning rate is shown in Note 11. The corresponding evaluation grid used for these experiments is shown in Note 12.

Note 11: 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 12: 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 45 to Figure 53.

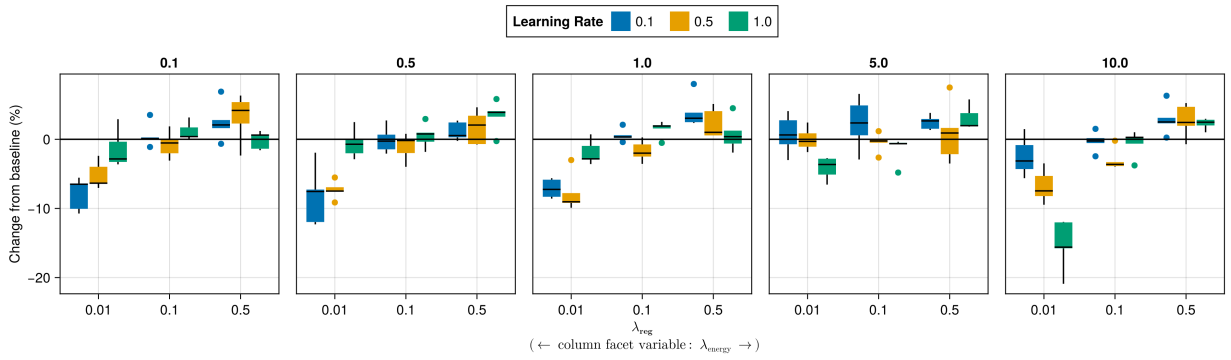

Figure 45: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Adult.

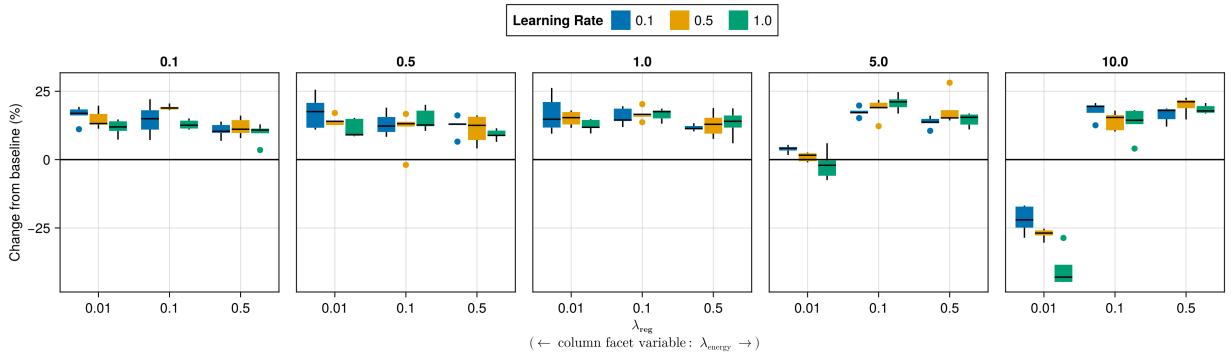

Figure 46: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: California Housing.

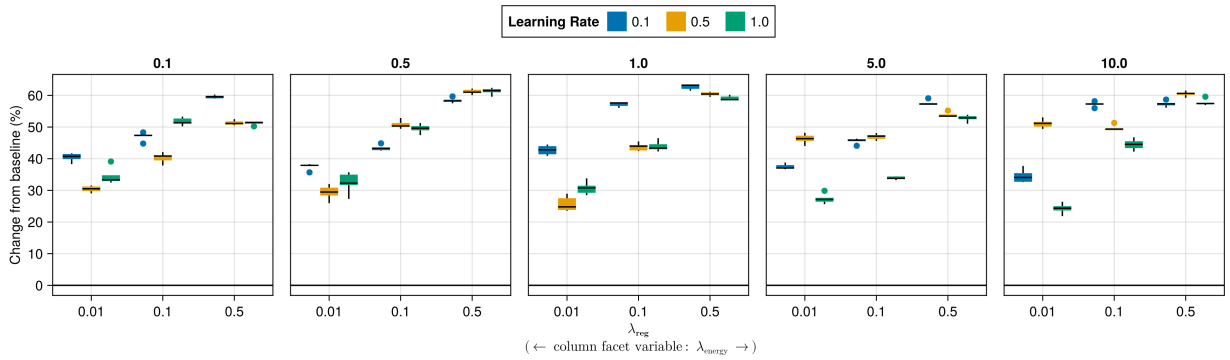

Figure 47: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Circles.

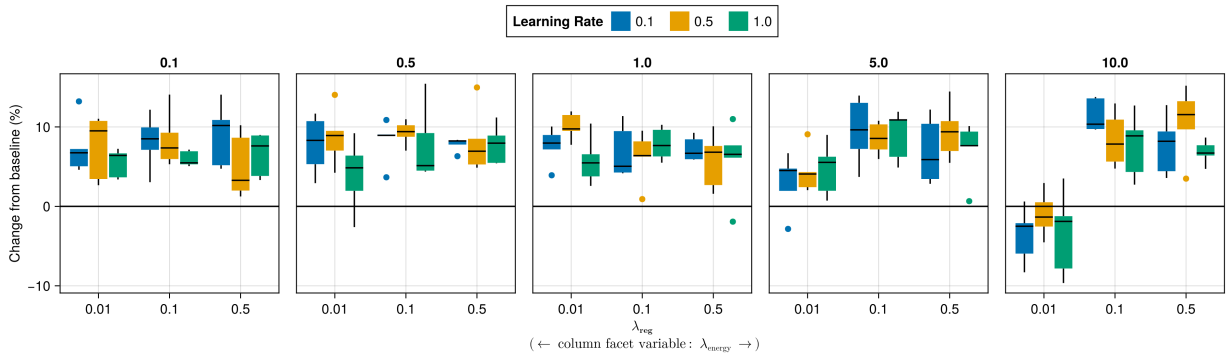

Figure 48: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Credit.

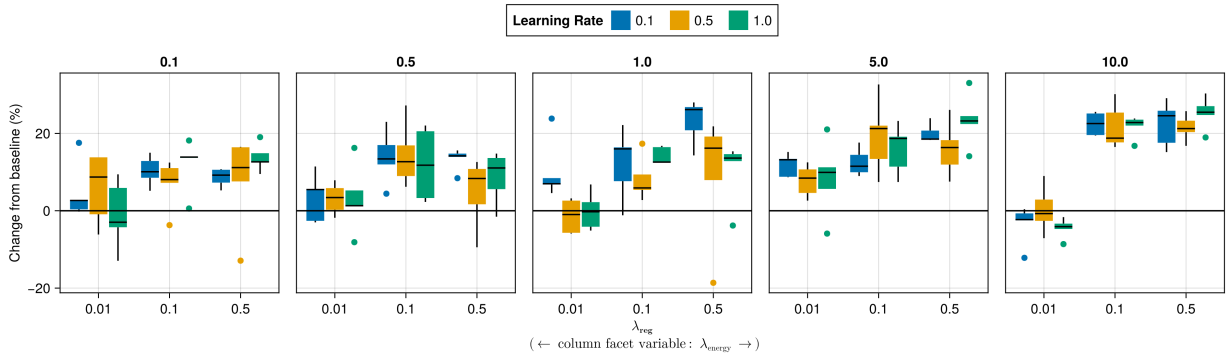

Figure 49: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: GMSC.

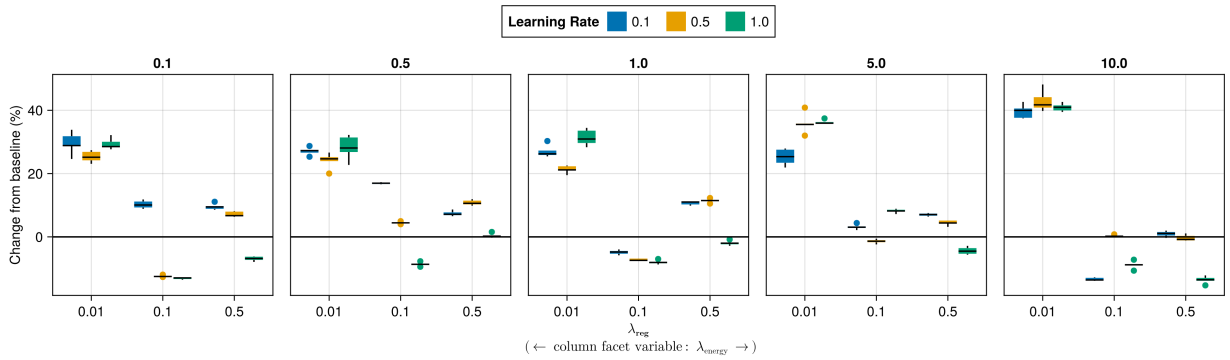

Figure 50: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Linearly Separable.

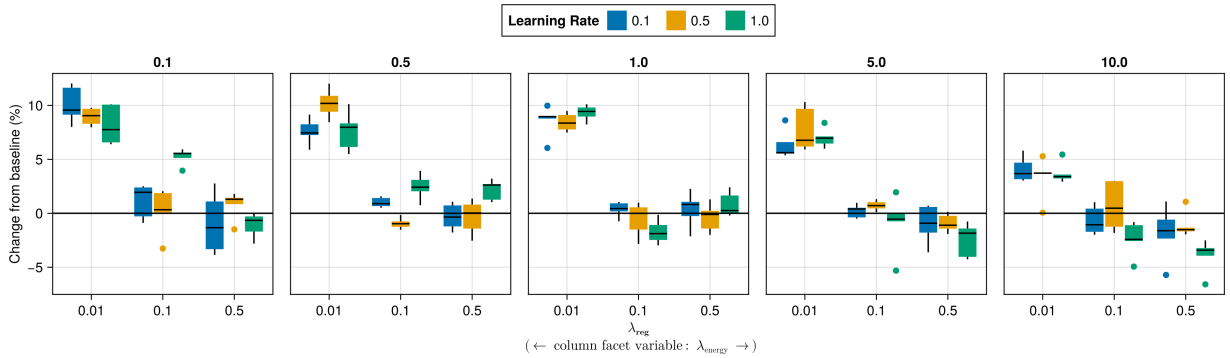

Figure 51: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: MNIST.

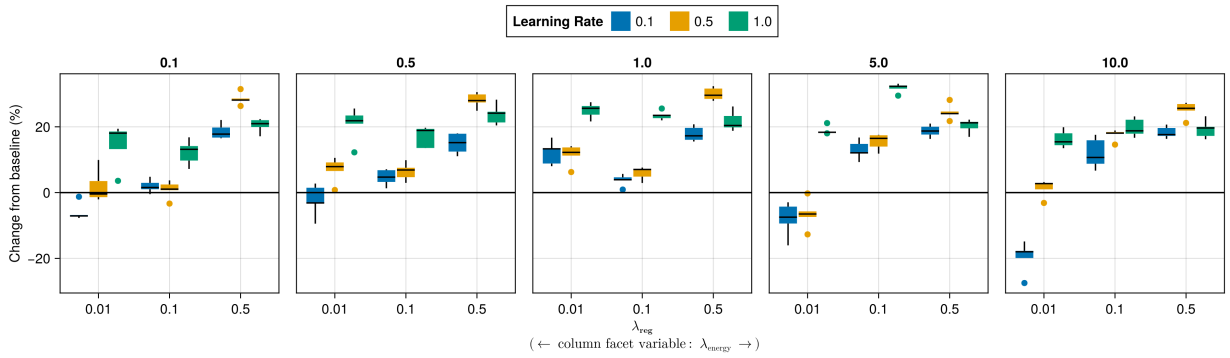

Figure 52: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Moons.

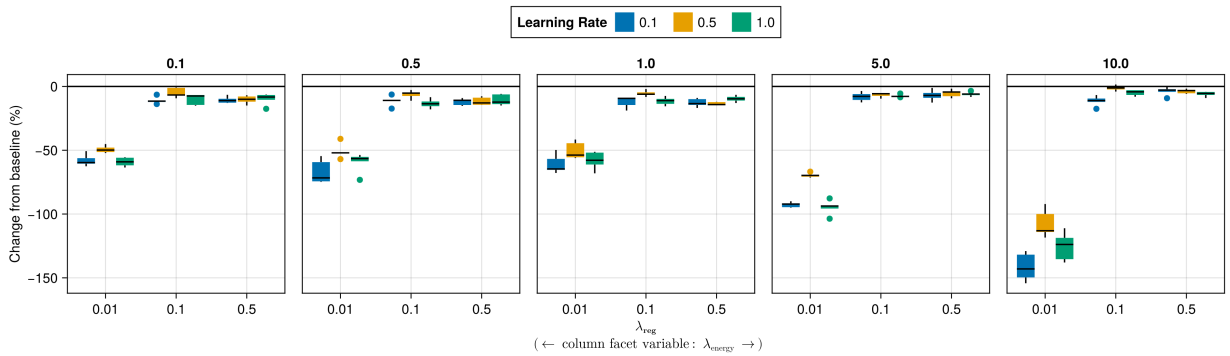

Figure 53: Average outcomes for the plausibility measure across key hyperparameters. This shows the % change from the baseline model for the distance-based implausibility metric (IP). Boxplots indicate the variation across evaluation runs and test settings (varying parameters for *ECCCo*). Data: Overlapping.

E.2.2 Proportion of Mature CE

The results with respect to the proportion of mature counterfactuals in each epoch are shown in Figure 54 to Figure 62.

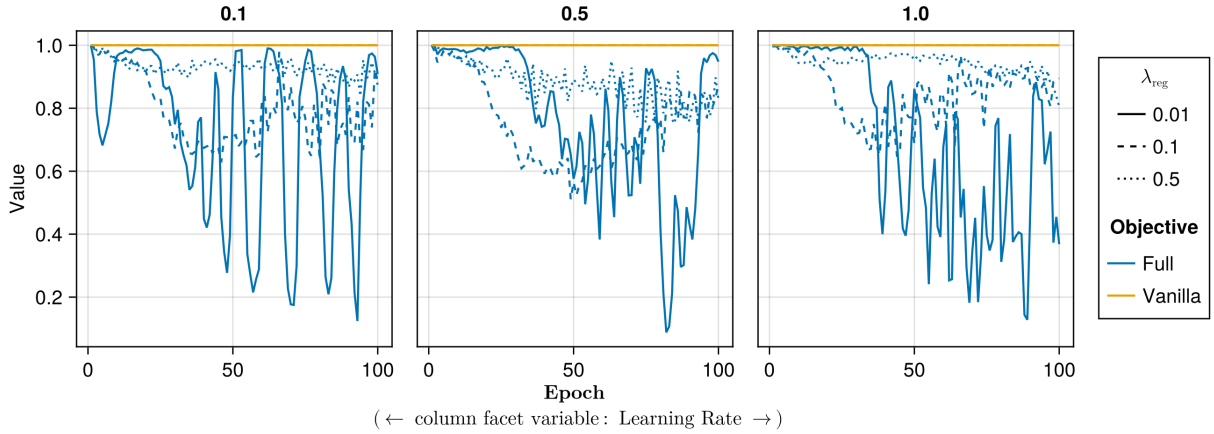

Figure 54: Proportion of mature counterfactuals in each epoch. Data: Adult.

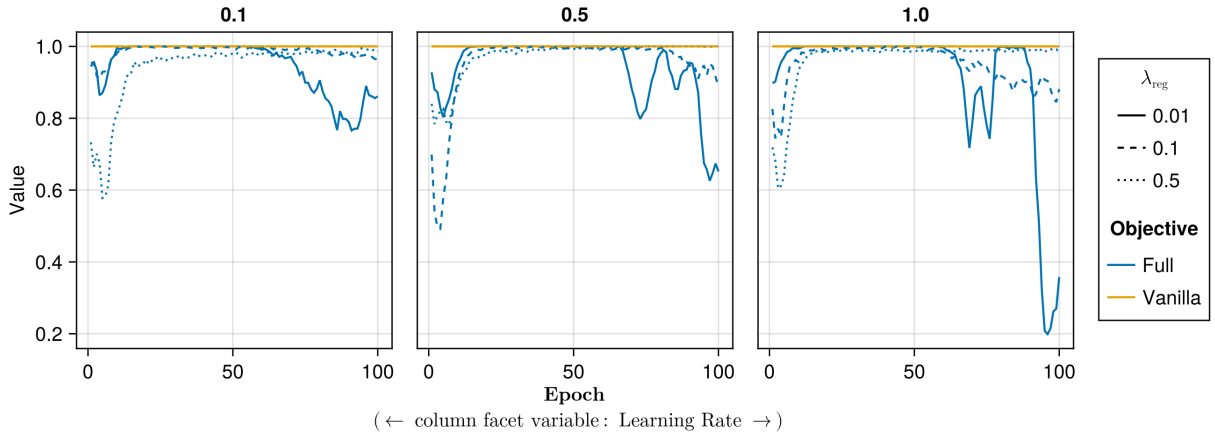

Figure 55: Proportion of mature counterfactuals in each epoch. Data: California Housing.

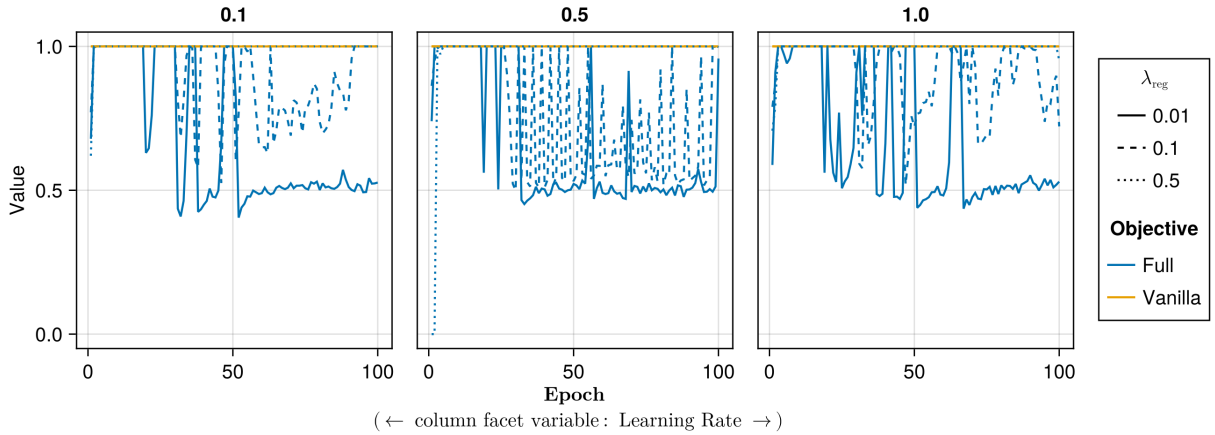

Figure 56: Proportion of mature counterfactuals in each epoch. Data: Circles.

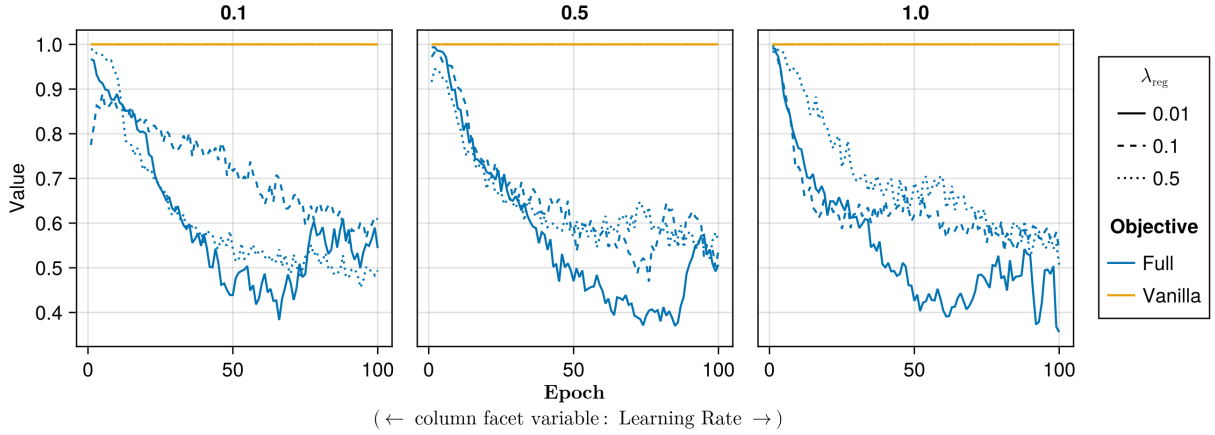

Figure 57: Proportion of mature counterfactuals in each epoch. Data: Credit.

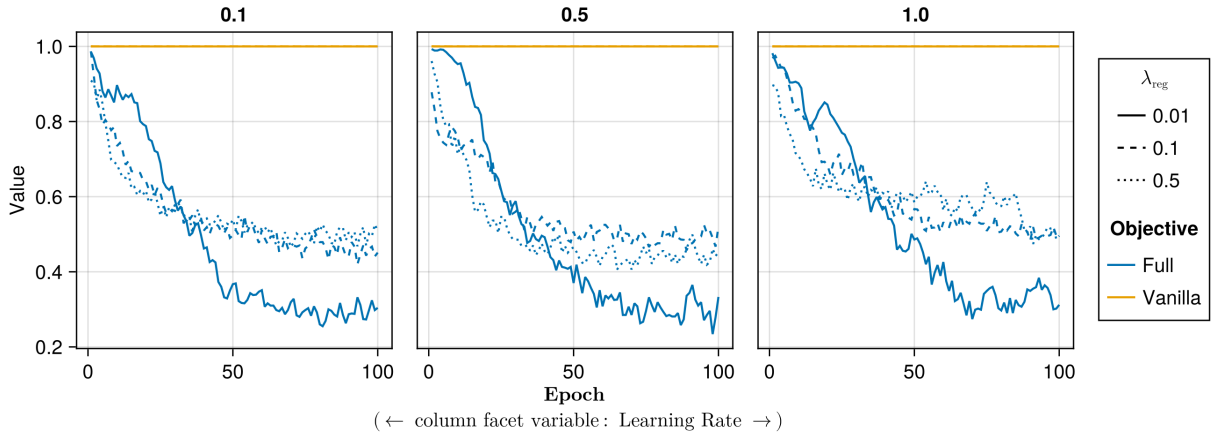

Figure 58: Proportion of mature counterfactuals in each epoch. Data: GMSC.

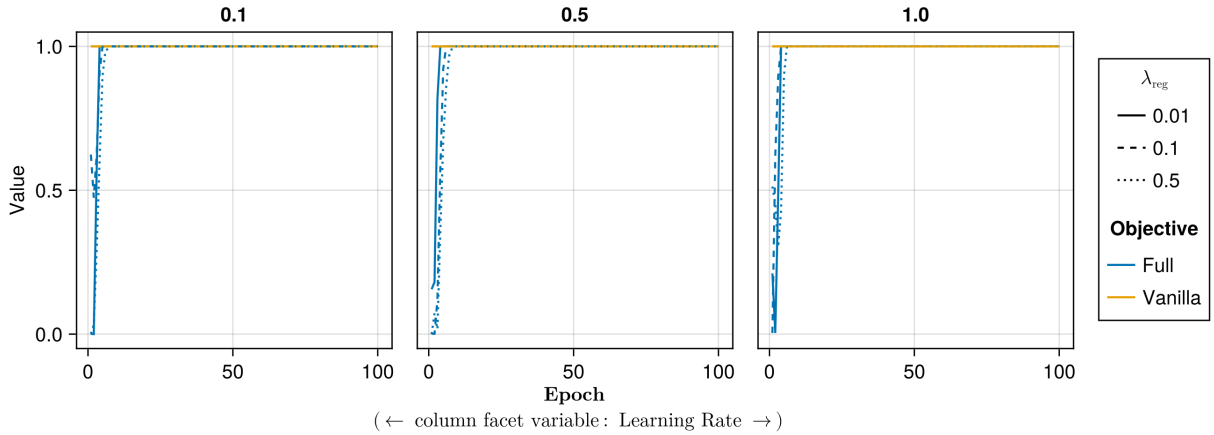

Figure 59: Proportion of mature counterfactuals in each epoch. Data: Linearly Separable.

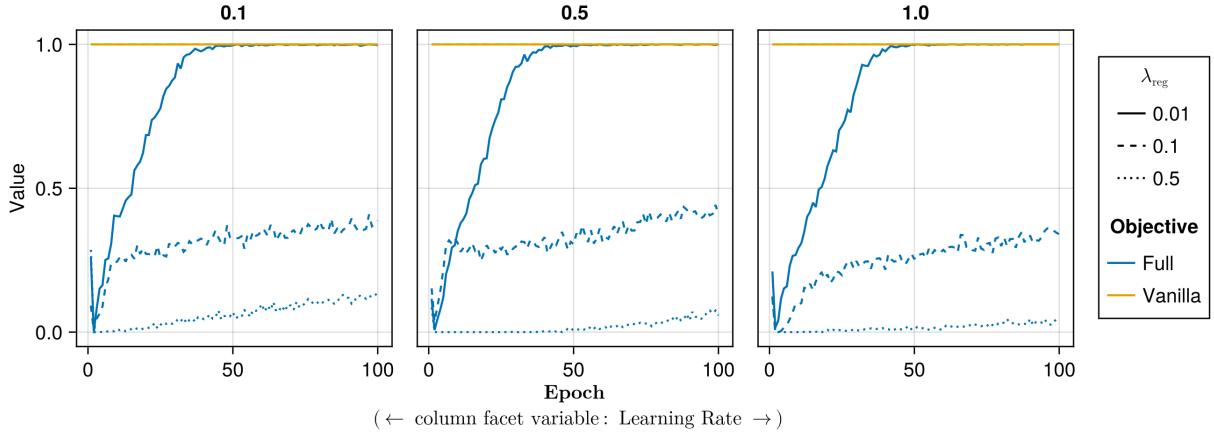

Figure 60: Proportion of mature counterfactuals in each epoch. Data: MNIST.

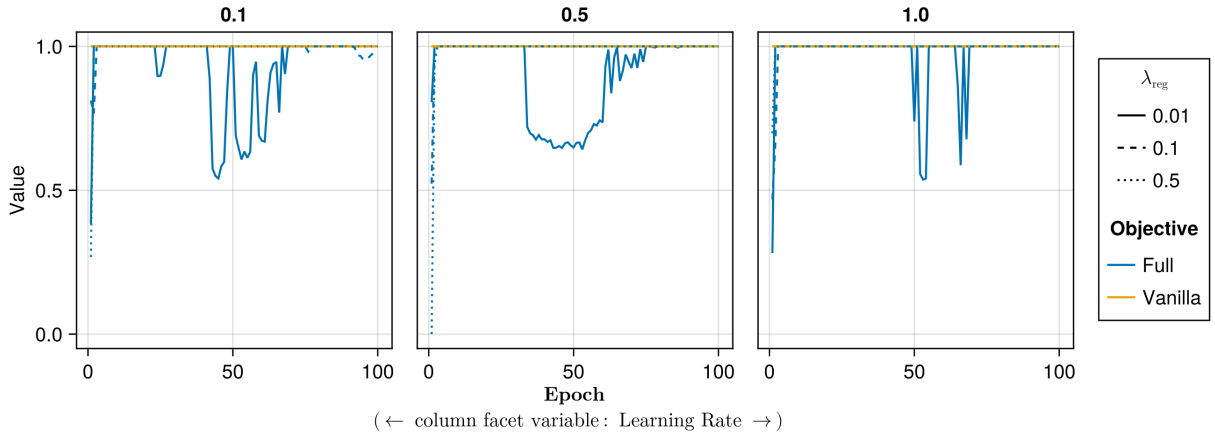

Figure 61: Proportion of mature counterfactuals in each epoch. Data: Moons.

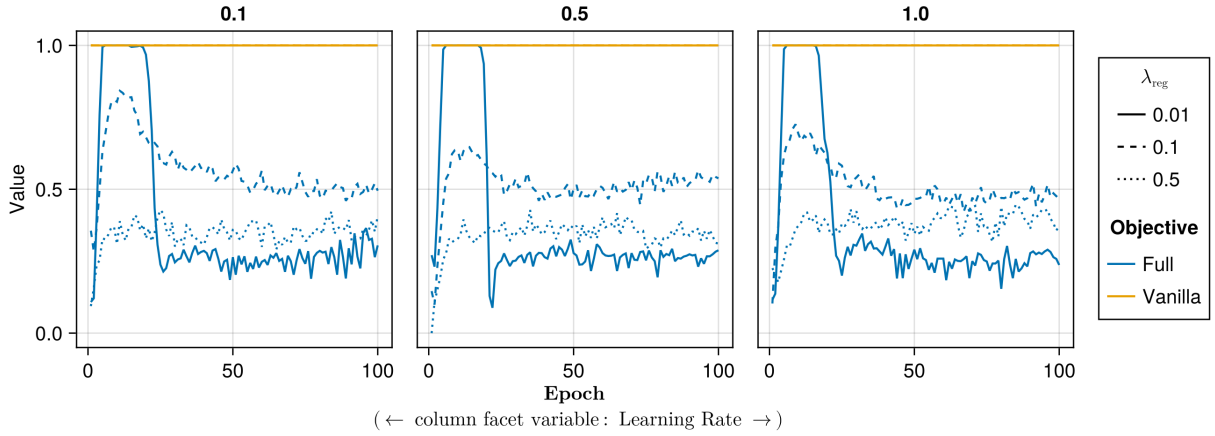

Figure 62: Proportion of mature counterfactuals in each epoch. Data: Overlapping.

Appendix F Computation Details

F.1 Hardware

We performed our experiments on a high-performance cluster ((DHPC) 2022). 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* used.

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: 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 but were generally more memory-intensive (see Section F.2 for additional details)

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

Our code has been open-sourced on GitHub as Julia package: CounterfactualTraining.jl. 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). We chose to work with CounterfactualExplanations.jl because it currently appears to be the most comprehensive and extensible package for counterfactual explanations. Despite its good interplay with Flux.jl, the package is not, however, optimized to be used in training. This has caused some issues with memory management and bottlenecked performance. The code is commented with clearly marked references to the paper (look for `# ----- PAPER REF -----`).

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).

F.3 Reproducibility

We have taken care to set random seeds for reproducibility using Julia's Random.jl package from the standard library. A global seed and (if applicable or wanted) dataset-specific seeds can be specified in TOML configuration files, environment variables or in interactive Julia sessions. Additional details can be found in the code base.

References

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.

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.

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.

(DHPC), Delft High Performance Computing Centre. 2022. "DelftBlue Supercomputer (Phase 1)." https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

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.

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.

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.

Sturmfels, Pascal, Scott Lundberg, and Su-In Lee. 2020. "Visualizing the Impact of Feature Attribution Baselines." *Distill* 5 (1): e22.

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan. 2017. "Axiomatic Attribution for Deep Networks." https://arxiv.org/abs/1703.01365.

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.