
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 9, 2025

ABSTRACT

We propose a novel training regime called 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 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 this 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 advances in **representation learning**: instead of relying on features and rules that are carefully hand-crafted by humans, modern machine learning (ML)

18 models are tasked with learning these representations from scratch, guided by narrow objectives such as predictive
 19 accuracy (I. Goodfellow, Bengio, and Courville 2016). Modern advances in computing have made it possible to
 20 provide such models with ever greater degrees of freedom to achieve that task, which has often led them to outperform
 21 traditionally more parsimonious models. Unfortunately, in doing so they also learn increasingly complex and highly
 22 sensitive representations that we can no longer easily interpret.

23 This trend towards complexity for the sake of performance has come under serious scrutiny in recent years. At the very
 24 cusp of the deep learning revolution, Szegedy et al. (2013) showed that artificial neural networks (ANN) are sensitive
 25 to adversarial examples: counterfactuals of model inputs that yield vastly different model predictions despite being
 26 “imperceptible” in that they are semantically indifferent from their factual counterparts. Despite partially effective
 27 mitigation strategies such as **adversarial training** (I. J. Goodfellow, Shlens, and Szegedy 2014), truly robust deep
 28 learning (DL) remains unattainable even for models that are considered shallow by today’s standards (Kolter 2023).

29 Part of the problem is that high degrees of freedom provide room for many solutions that are locally optimal with
 30 respect to narrow objectives (Wilson 2020)¹. Based purely on predictive performance, these solutions may seem to
 31 provide compelling explanations for the data, when in fact they are based on purely associative, semantically mean-
 32 ingless patterns. This poses two related challenges: firstly, it makes these models inherently opaque, since humans
 33 cannot simply interpret what type of explanation the complex learned representations correspond to; secondly, even
 34 if we could resolve the first challenge, it is not obvious how to mitigate models from learning representations that
 35 correspond to meaningless and implausible explanations.

36 The first challenge has attracted an abundance of research on **explainable AI** (XAI) which aims to develop tools to
 37 derive explanations from complex model representations. This can mitigate a scenario in which we deploy opaque
 38 models and blindly rely on their predictions. On countless occasions, this scenario has already occurred in practice
 39 and caused real harm to people who were affected adversely and often unfairly by automated decision-making systems
 40 (ADMS) involving opaque models (O’Neil 2016). Effective XAI tools can aide us in monitoring models and providing
 41 recourse to individuals to turn adverse outcomes (e.g. “loan application rejected”) into positive ones (“application
 42 accepted”). Wachter, Mittelstadt, and Russell (2017) propose **counterfactual explanations** as an effective approach
 43 to achieve this: they explain how factual inputs need to change in order for some fitted model to produce some desired
 44 output, typically involving minimal perturbations.

45 To our surprise, the second challenge has not yet attracted any consolidated research effort. Specifically, there has
 46 been no concerted effort towards improving model **explainability**, which we define here as the degree to which learned
 47 representations correspond to explanations that are interpretable and deemed **plausible** by humans (see Definition 3.1).
 48 Instead, the choice has typically been to improve the capacity of XAI tools to identify the subset explanations that are
 49 both plausible and valid for any given model, independent of whether the learned representations are also compatible
 50 with implausible explanations (Altmeyer et al. 2024). Fortunately, recent findings indicate that explainability can arise
 51 as byproduct of regularization techniques aimed at other objectives such as robustness, generalization and generative
 52 capacity Altmeyer et al. (2024).

53 Building on these findings, we introduce **counterfactual training**: a novel regularization technique geared explicitly
 54 towards aligning model representations with plausible explanations. Our contributions are as follows:

- 55 • We discuss existing related work on improving models and consolidate it through the lens of counterfactual
 56 explanations (Section 2).
- 57 • We present our proposed methodological framework that leverages faithful counterfactual explanations during
 58 the training phase of models to achieve the explainability objective (Section 3).
- 59 • Through extensive experiments we demonstrate the counterfactual training improve model explainability
 60 while maintaining high predictive performance. We run ablation studies and grid searches to understand
 61 how the underlying model components and hyperparameters affect outcomes. (Section 4).

62 Despite limitations of our approach discussed in Section 5, we conclude that counterfactual training provides a practi-
 63 cal framework for researchers and practitioners interested in making opaque models more trustworthy Section 6. We
 64 also believe that this work serves as an opportunity for XAI researchers to reevaluate the premise of improving XAI
 65 tools without improving models.

66 2 Related Literature

67 To the best of our knowledge, our proposed framework for counterfactual training represents the first attempt to use
 68 counterfactual explanations during training to improve model explainability. In high-level terms, we define model

¹For clarity: we follow standard ML convention in using “degrees of freedom” to refer to the number of parameters estimated from data.

69 explainability as the extent to which valid explanations derived for an opaque model are also deemed plausible with
 70 respect to the underlying data and stakeholder requirements. To make this more concrete, we follow Augustin, Meinke,
 71 and Hein (2020) in tying the concept of explainability to the quality of counterfactual explanations that we can
 72 generate for a given model. The authors show that counterfactual explanations—understood here as minimal input
 73 perturbations that yield some desired model prediction—are generally more meaningful if the underlying model is
 74 more robust to adversarial examples. We can make intuitive sense of this finding when looking at adversarial training
 75 (AT) through the lens of representation learning with high degrees of freedom: by inducing models to “unlearn”
 76 representations that are susceptible to worst-case counterfactuals (i.e. adversarial examples), AT effectively removes
 77 some implausible explanations from the solution space.

78 2.1 Adversarial Examples are Counterfactual Explanations

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

89 In the natural language processing (NLP) domain, counterfactuals have similarly been used to improve models through
 90 data augmentation: Wu et al. (2021), propose *Polyjuice*, a general-purpose counterfactual generator for language mod-
 91 els. They demonstrate empirically that augmenting training data through *Polyjuice* counterfactuals improves robust-
 92 ness in a number of NLP tasks. Balashankar et al. (2023) also use *Polyjuice* to augment NLP datasets through diverse
 93 counterfactuals and show that classifier robustness improves up to 20%. Finally, Luu and Inoue (2023) introduce
 94 Counterfactual Adversarial Training (CAT), which also aims at improving generalization and robustness of language
 95 models. Specifically, they propose to proceed as follows: firstly, they identify training samples that are subject to
 96 high predictive uncertainty; secondly, they generate counterfactual explanations for those samples; and, finally, they
 97 fine-tune the given language model on the augmented dataset that includes the generated counterfactuals.

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

104 Two recent works are closely related to ours in that they use counterfactuals during training with the explicit goal
 105 of affecting certain properties of post-hoc counterfactual explanations. Firstly, Ross, Lakkaraju, and Bastani (2024)
 106 propose a way to train models that are guaranteed to provide recourse for individuals to move from an adverse outcome
 107 to some positive target class with high probability. The approach proposed by Ross, Lakkaraju, and Bastani (2024)
 108 builds on adversarial training, where in this context susceptibility to targeted adversarial examples for the positive
 109 class is explicitly induced. The proposed method allows for imposing a set of actionability constraints ex-ante: for
 110 example, users can specify that certain features (e.g. *age*, *gender*, ...) are immutable. Secondly, Guo, Nguyen, and
 111 Yadav (2023) are the first to propose an end-to-end training pipeline that includes counterfactual explanations as part
 112 of the training procedure. In particular, they propose a specific network architecture that includes a predictor and CE
 113 generator network, where the parameters of the CE generator network are learnable. Counterfactuals are generated
 114 during each training iteration and fed back to the predictor network. In contrast to Guo, Nguyen, and Yadav (2023),
 115 we impose no restrictions on the neural network architecture at all.

116 2.2 Beyond Robustness

117 Improving the adversarial robustness of models is not the only path towards aligning representations with plausible
 118 explanations. In a work closely related to this one, Altmeyer et al. (2024) show that explainability can be improved
 119 through model averaging and refined model objectives. The authors propose a way to generate counterfactuals that
 120 are maximally **faithful** to the model in that they are consistent with what the model has learned about the underlying
 121 data. Formally, they rely on tools from energy-based modelling to minimize the divergence between the distribution
 122 of counterfactuals and the conditional posterior over inputs learned by the model. Their proposed counterfactual
 123 explainer, *ECCCo*, yields plausible explanations if and only if the underlying model has learned representations that

124 align with them. They 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.

126 Once again it helps to look at these findings through the lens of representation learning with high degrees of freedom.
 127 Deep ensembles are approximate Bayesian model averages, which are most called for when models are underspecified
 128 by the available data (Wilson 2020). Averaging across solutions mitigates the aforementioned risk of relying on a
 129 single locally optimal representations that corresponds to semantically meaningless explanations for the data. Previous
 130 work by Schut et al. (2021) similarly found that generating plausible (“interpretable”) counterfactual explanations is
 131 almost trivial for deep ensembles that have also undergone adversarial training. The case for JEMs is even clearer:
 132 they involve a hybrid objective that induces both high predictive performance and generative capacity (Grathwohl et al.
 133 2020). This is closely related to the idea of aligning models with plausible explanations and has inspired our proposed
 134 counterfactual training objective, as we explain in Section 3.

135 3 Counterfactual Training

136 Counterfactual training combines ideas from adversarial training, energy-based modelling and counterfactuals expla-
 137 nations with the explicit objective of aligning representations with plausible explanations that comply with user re-
 138 quirements. In the context of CE, plausibility has broadly been defined as the degree to which counterfactuals comply
 139 with the underlying data generating process (Poyiadzi et al. 2020; Guidotti 2022; Altmeyer et al. 2024). Plausibility
 140 is a necessary but insufficient condition for using CE to provide algorithmic recourse (AR) to individuals affected by
 141 opaque models in practice. This is because for recourse recommendations to be **actionable**, they need to not only
 142 result in plausible counterfactuals but also be attainable. A plausible CE for a rejected 20-year-old loan applicant, for
 143 example, might reveal that their application would have been accepted, if only they were 20 years older. Ignoring all
 144 other features, this complies with the definition of plausibility if 40-year-old individuals are in fact more credit-worthy
 145 on average than young adults. But of course this CE does not qualify for providing actionable recourse to the applicant
 146 since *age* is not a mutable feature. For our intents and purposes, counterfactual training aims at improving model ex-
 147 plainability by aligning models with counterfactuals that meet both desiderata, plausibility and actionability. Formally,
 148 we define explainability as follows:

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

153 We say that M_θ is **explainable** to the extent that faithfully generated counterfactuals are plausible (i.e. consistent with
 154 the data) and actionable. Formally, we define these properties as follows:

- 155 1. (Plausibility) $\int^A p(\mathbf{x}'|\mathbf{y}^+) d\mathbf{x} \rightarrow 1$ where A is some small region around \mathbf{x}' .
- 156 2. (Actionability) Permutations Δ are subject to actionability constraints.

157 We consider counterfactuals as faithful to the extent that they are consistent with what the model has learned about the
 158 input data. Let $p_\theta(\mathbf{x}|\mathbf{y}^+)$ denote the conditional posterior over inputs, then formally:

- 159 3. (Faithfulness) $\int^A p_\theta(\mathbf{x}'|\mathbf{y}^+) d\mathbf{x} \rightarrow 1$ where A is defined as above.

160 The definitions of faithfulness and plausibility in Definition 3.1 are the same as in Altmeyer et al. (2024), with adapted
 161 notation. Actionability constraints in Definition 3.1 vary and depend on the context in which M_θ is deployed. In this
 162 work, we focus on domain and mutability constraints for individual features x_d for $d = 1, \dots, D$. We limit ourselves
 163 to classification tasks for reasons discussed in Section 5.

164 3.1 Our Proposed Objective

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

169 To train models with high explainability as defined in Definition 3.1, we propose to leverage counterfactuals in the
 170 following objective,

²For detailed background information on gradient-based counterfactual search and convergence see Section H.1.

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

171 where $\text{yloss}(\cdot)$ is any conventional classification loss that induces discriminative performance (e.g. cross-entropy).
 172 The two additional components in Equation 1 are explained in more detail below. For now, they can be sufficiently de-
 173 scribed as inducing explainability directly and indirectly by penalizing: 1) the contrastive divergence, $\text{div}(\cdot)$, between
 174 counterfactuals \mathbf{x}'_T and observed samples x and, 2) the adversarial loss, $\text{advloss}(\cdot)$, with respect to nascent counterfac-
 175 tuals $\mathbf{x}'_{t \leq T}$. The tradeoff between the different components can be governed by adjusting the strengths of the penalties
 176 λ_{div} and λ_{adv} .

177 3.1.1 Directly Inducing Explainability through Contrastive Divergence

178 Grathwohl et al. (2020) observe that any classifier can be re-interpreted as a joint energy-based model (JEM)
 179 that learns to discriminate output classes conditional on inputs and generate inputs. They show that JEMs can be
 180 trained to perform well at both tasks by directly maximizing the joint log-likelihood factorized as $\log p_{\theta}(\mathbf{x}, \mathbf{y}) =$
 181 $\log p_{\theta}(\mathbf{y}|\mathbf{x}) + \log p_{\theta}(\mathbf{x})$. The first factor can be optimized using conventional cross-entropy as in Equation 1. To
 182 optimize $\log p_{\theta}(\mathbf{x})$ Grathwohl et al. (2020) minimize the contrastive divergence between samples drawn from $p_{\theta}(\mathbf{x})$
 183 and training observations, i.e. samples from $p(\mathbf{x})$.

184 A key empirical finding in Altmeyer et al. (2024) was that JEMs tend to do well with respect to the plausibility objec-
 185 tive in Definition 3.1. If we consider samples drawn from $p_{\theta}(\mathbf{x})$ as counterfactuals, this is an expected finding, because
 186 the JEM objective effectively minimizes the divergence between the conditional posterior and $p(\mathbf{x}|y^+)$. To generate
 187 samples, Grathwohl et al. (2020) rely on Stochastic Gradient Langevin Dynamics (SGLD) using an uninformative
 188 prior for initialization. This is where we depart from their methodology: instead of generating samples through SGLD,
 189 we propose using counterfactual explainers to generate counterfactuals for observed training samples. Specifically, we
 190 have

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

191 where $\mathcal{E}_{\theta}(\cdot)$ denotes the energy function. In particular, we set $\mathcal{E}_{\theta}(\mathbf{x}, y) = -\mathbf{M}_{\theta}(\mathbf{x})[y^+]$ where y^+ denotes the index of
 192 the target class. We generate samples \mathbf{x}'_T by first randomly sampling the target class $y^+ \sim p(y)$ and then generating
 193 a counterfactual explanation for that target over T iterations using a gradient-based counterfactual generator. This is
 194 similar to how conditional sampling is used to draw from $p_{\theta}(\mathbf{x})$ in Grathwohl et al. (2020).

195 Intuitively, the gradient of Equation 2 decreases the energy of observed training samples (positive samples) while at
 196 same time increasing the energy of counterfactuals (negative samples) (Du and Mordatch 2020). As the generated
 197 counterfactuals get more plausible (Definition 3.1) over the cause of training, these two opposing effects gradually
 198 balance each out (Lippe 2024).

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

204 3.1.2 Indirectly Inducing Explainability through Adversarial Robustness

205 Based on our analysis in Section 2, counterfactuals \mathbf{x}' can be repurposed as additional training samples (Luu and Inoue
 206 2023; Balashankar et al. 2023) or adversarial examples (Freiesleben 2022; Pawelczyk et al. 2022). This leaves some
 207 flexibility with respect to the exact choice for $\text{advloss}(\cdot)$ in Equation 1. An intuitive functional form to use, though
 208 likely not the only reasonable choice, is inspired by adversarial training:

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

209 Under this choice, we consider nascent counterfactuals $\mathbf{x}'_{t \leq T}$ as adversarial examples as long as the magnitude of the
 210 perturbation to any individual feature is at most ε . This is closely aligned with Szegedy et al. (2013), who define an
 211 adversarial attack as an “imperceptible non-random perturbation”. Thus, we choose to work with a different distinction
 212 between CE and AE than Freiesleben (2022), who considers misclassification as the key distinguishing feature of AE.
 213 One of the key observations in this work is that we can leverage counterfactual explanations during training and get
 214 adversarial examples, essentially for free.

215 **3.2 Encoding Actionability Constraints**

216 Many existing counterfactual explainers support domain and mutability constraints out-of-the-box. In fact, both types
 217 of constraints can be implemented for any counterfactual explainer that relies on gradient descent in the feature space
 218 for optimization (Altmeyer, Deursen, et al. 2023). In this context, domain constraints can be imposed by simply
 219 projecting counterfactuals back to the specified domain, if the previous gradient step resulted in updated feature values
 220 that were out-of-domain. Mutability constraints can similarly be enforced by setting partial derivatives to zero to
 221 ensure that features are only mutated in the allowed direction, if at all.

222 Since actionability constraints are binding at test time, we should also impose them when generating \mathbf{x}' during each
 223 training iteration to align model representations with user requirements. Through their effect on \mathbf{x}' , both types of
 224 constraints influence model outcomes through Equation 2. Here it is crucial that we avoid penalizing implausibility
 225 that arises due to mutability constraints. For any mutability-constrained feature d this can be achieved by enforcing
 226 $\mathbf{x}[d] - \mathbf{x}'[d] := 0$ whenever perturbing $\mathbf{x}'[d]$ in the direction of $\mathbf{x}[d]$ would violate mutability constraints. Specifically,
 227 we set $\mathbf{x}[d] := \mathbf{x}'[d]$ if

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

231 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
 232 respect to feature d . Intuitively, we think of this simply in terms ignoring implausibility costs with respect to immutable
 233 features, which effectively forces the model to instead seek plausibility with respect to the remaining features. This
 234 in turn results in lower overall sensitivity to immutable features, which we demonstrate empirically for different
 235 classifiers in Section 4. Under certain conditions, this results holds theoretically[For the proof, see the supplementary
 236 appendix.]:

237 **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
 238 softmax activation \mathcal{S} (i.e. multinomial logistic regression) where $y \in \{1, \dots, K\} = \mathcal{K}$ and $\mathbf{x} \in \mathbb{R}^D$. If we assume
 239 multivariate Gaussian class densities with common diagonal covariance matrix $\Sigma_k = \Sigma$ for all $k \in \mathcal{K}$, then protecting
 240 an immutable feature from the contrastive divergence penalty (Equation 2) will result in lower classifier sensitivity to
 241 that feature relative to the remaining features, provided that at least one of those is mutable and discriminative.*

242 It is worth highlighting that Proposition 3.1 assumes independence of features. This raises a valid concern about the
 243 effect of protecting immutable features in the presence of proxy features that remain unprotected. We discuss this
 244 limitation in Section 5.

245 **3.3 Illustration**

246 To better convey the intuition underlying our proposed method, we illustrate different model outcomes in Example 3.1.

247 **Example 3.1** (Prediction of Consumer Credit Default). Suppose we are interested in predicting the likelihood that
 248 loan applicants default on their credit. We have access to historical data on previous loan takers comprised of a binary
 249 outcome variable ($y \in \{1 = \text{default}, 2 = \text{no default}\}$) two input features: 1) the subjects' *age*, which we define as
 250 immutable, and 2) the subjects' existing level of *debt*, which we define as mutable.

251 We have simulated this scenario using synthetic data with independent features and Gaussian class-conditional densities
 252 in Figure 1. The four panels in Figure 1 show the outcomes for different training procedures using the same model
 253 architecture each time (a linear classifier). In each case, we show the linear decision boundary (green) and the training
 254 data colored according to their ground-truth label: orange points belong to the target class, $y^+ = 2$, blue points belong
 255 to the non-target class, $y^- = 1$. Stars indicate counterfactuals in the target class generated at test time using generic
 256 gradient descent until convergence.

257 In panel (a), we have trained our model conventionally, and we do not impose mutability constraints at test time. The
 258 generated counterfactuals are all valid, but not plausible: they are clearly distinguishable from the ground-truth data. In
 259 panel (b), we have trained our model with counterfactual training, once again not imposing mutability constraints at test
 260 time. We observe that the counterfactuals are clearly plausible, therefore meeting the first objective of Definition 3.1.

261 In panel (c), we have used conventional training again, this time imposing the mutability constraint on *age* at test
 262 time. Counterfactuals are valid but involve some substantial reductions in *debt* for some individuals, in particular
 263 very young applicants. By comparison, counterfactual paths are shorter on average in panel (d), where we have used
 264 counterfactual training and protected immutable features as described in Section 3.2. In particular, we observe that due
 265 to the classifier's lower sensitivity to *age*, recourse recommendations with respect to *debt* are much more homogenous,

266 in that they do not disproportionately punish younger individuals. The counterfactuals are also plausible with respect
 267 to the mutable feature. Thus, we consider the model in panel (d) as the most explainable according to Definition 3.1.

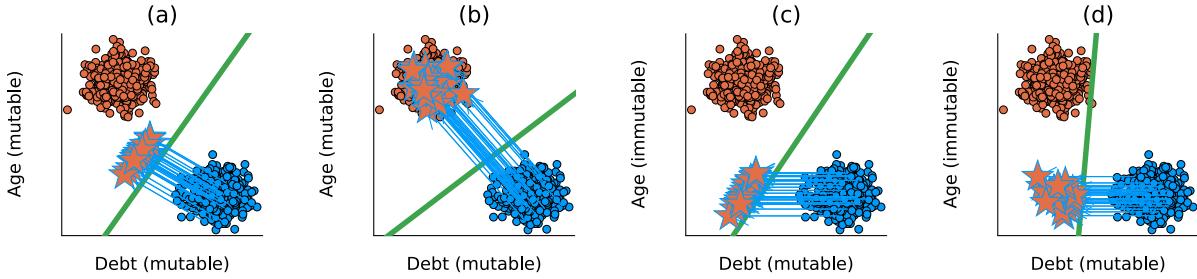


Figure 1: Visual illustration of how counterfactual training improves explainability. See Example 3.1 for details.

268 4 Experiments

269 In this section, we present experiments that we have conducted in order to answer the following research questions:

270 **Research Question 4.1 (Plausibility).** *Does our proposed counterfactual training objective (Equation 1) induce mod-
 271 els to learn plausible explanations?*

272 **Research Question 4.2 (Actionability).** *Does our proposed counterfactual training objective (Equation 1) yield more
 273 favorable algorithmic recourse outcomes in the presence of actionability constraints?*

274 Beyond this, we are also interested in understanding how robust our answers to RQ 4.1 and RQ 4.2 are:

275 **Research Question 4.3 (Hyperparameters).** *What are the effects of different hyperparameter choices with respect to
 276 Equation 1?*

277 4.1 Experimental Setup

278 4.2 Experimental Results

279 5 Discussion

280 5.1 Approach/Future Directions

- 281 1. Limited to classification models.
- 282 2. Training instabilities.
- 283 3. Hyperparameter sensitivity -> can we do better than grid search? (Bayes opt, ...)

284 5.2 Limitations

- 285 3. Proxy attributes of immutable features.
- 286 4. Increased training time.
- 287 5. Fairness and caveats (aware it's not a classical approach in this context, but there is a clear link).

288 6 Conclusion

289 References

- 290 Abbasnejad, Ehsan, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. 2020. “Counterfactual
 291 Vision and Language Learning.” In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition
 292 (CVPR)*, 10041–51. <https://doi.org/10.1109/CVPR42600.2020.01006>.
- 293 Altmeyer, Patrick, Arie van Deursen, et al. 2023. “Explaining Black-Box Models Through Counterfactuals.” In
 294 *Proceedings of the JuliaCon Conferences*, 1:130. 1.
- 295 Altmeyer, Patrick, Mojtaba Farmanbar, Arie van Deursen, and Cynthia CS Liem. 2024. “Faithful Model Explanations
 296 Through Energy-Constrained Conformal Counterfactuals.” In *Proceedings of the AAAI Conference on Artificial
 297 Intelligence*, 38:10829–37. 10.
- 298 Augustin, Maximilian, Alexander Meinke, and Matthias Hein. 2020. “Adversarial Robustness on in-and Out-
 299 Distribution Improves Explainability.” In *European Conference on Computer Vision*, 228–45. Springer.

- 300 Balashankar, Ananth, Xuezhi Wang, Yao Qin, Ben Packer, Nithum Thain, Ed Chi, Jilin Chen, and Alex Beutel. 2023.
 301 “Improving Classifier Robustness Through Active Generative Counterfactual Data Augmentation.” In *Findings of*
 302 *the Association for Computational Linguistics: EMNLP 2023*, 127–39.
- 303 Bezanson, Jeff, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. “Julia: A Fresh Approach to Numerical
 304 Computing.” *SIAM Review* 59 (1): 65–98. <https://doi.org/10.1137/141000671>.
- 305 Bouchet-Valat, Milan, and Bogumi Kamiski. 2023. “DataFrames.jl: Flexible and Fast Tabular Data in Julia.” *Journal*
 306 *of Statistical Software* 107 (4): 1–32. <https://doi.org/10.18637/jss.v107.i04>.
- 307 Byrne, Simon, Lucas C. Wilcox, and Valentin Churavy. 2021. “MPI.jl: Julia Bindings for the Message Passing
 308 Interface.” *Proceedings of the JuliaCon Conferences* 1 (1): 68. <https://doi.org/10.21105/jcon.00068>.
- 309 Chagas, Ronan Arraes Jardim, Ben Baumgold, Glen Hertz, Hendrik Ranocha, Mark Wells, Nathan Boyer, Nicholas
 310 Ritchie, et al. 2024. “Ronisbr/PrettyTables.jl: V2.4.0.” Zenodo. <https://doi.org/10.5281/zenodo.1383553>.
- 311 Christ, Simon, Daniel Schwabeneder, Christopher Rackauckas, Michael Krabbe Borregaard, and Thomas Breloff.
 312 2023. “Plots.jl – a User Extendable Plotting API for the Julia Programming Language.” <https://doi.org/https://doi.org/10.5334/jors.431>.
- 313 Danisch, Simon, and Julius Krumbiegel. 2021. “Makie.jl: Flexible High-Performance Data Visualization for Julia.”
 314 *Journal of Open Source Software* 6 (65): 3349. <https://doi.org/10.21105/joss.03349>.
- 315 Du, Yilun, and Igor Mordatch. 2020. “Implicit Generation and Generalization in Energy-Based Models.” <https://arxiv.org/abs/1903.08689>.
- 316 Freiesleben, Timo. 2022. “The Intriguing Relation Between Counterfactual Explanations and Adversarial Examples.”
 317 *Minds and Machines* 32 (1): 77–109.
- 318 Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy. 2014. “Explaining and Harnessing Adversarial Examples.”
 319 <https://arxiv.org/abs/1412.6572>.
- 320 Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
- 321 Grathwohl, Will, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swer-
 322 sky. 2020. “Your Classifier Is Secretly an Energy Based Model and You Should Treat It Like One.” In *International*
 323 *Conference on Learning Representations*.
- 324 Guidotti, Riccardo. 2022. “Counterfactual Explanations and How to Find Them: Literature Review and Benchmark-
 325 ing.” *Data Mining and Knowledge Discovery*, 1–55.
- 326 Guo, Hangzhi, Thanh H. Nguyen, and Amulya Yadav. 2023. “CounterNet: End-to-End Training of Prediction Aware
 327 Counterfactual Explanations.” In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery*
 328 and *Data Mining*, 577–89. KDD ’23. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3580305.3599290>.
- 329 Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. Springer New
 330 York. <https://doi.org/10.1007/978-0-387-84858-7>.
- 331 Innes, Michael, Elliot Saba, Keno Fischer, Dhairyा Gandhi, Marco Conchetto Rudolosso, Neethu Mariya Joy, Tejan
 332 Karmali, Avik Pal, and Viral Shah. 2018. “Fashionable Modelling with Flux.” <https://arxiv.org/abs/1811.01457>.
- 333 Innes, Mike. 2018. “Flux: Elegant Machine Learning with Julia.” *Journal of Open Source Software* 3 (25): 602.
 334 <https://doi.org/10.21105/joss.00602>.
- 335 Joshi, Shalmali, Oluwasanmi Koyejo, Warut Vigitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. “Towards Realistic
 336 Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems.” <https://arxiv.org/abs/1907.09615>.
- 337 Kolter, Zico. 2023. “Keynote Addresses: SaTML 2023 .” In *2023 IEEE Conference on Secure and Trustworthy*
 338 *Machine Learning (SaTML)*, xvi–. Los Alamitos, CA, USA: IEEE Computer Society. <https://doi.org/10.1109/SaTML54575.2023.00009>.
- 339 Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. 2017. “Simple and Scalable Predictive Uncer-
 340 tainty Estimation Using Deep Ensembles.” *Advances in Neural Information Processing Systems* 30.
- 341 Lippe, Phillip. 2024. “UVa Deep Learning Tutorials.” <https://uvadlc-notebooks.readthedocs.io/en/latest/>.
- 342 Luu, Hoai Linh, and Naoya Inoue. 2023. “Counterfactual Adversarial Training for Improving Robustness of Pre-
 343 Trained Language Models.” In *Proceedings of the 37th Pacific Asia Conference on Language, Information and*
 344 *Computation*, 881–88.
- 345 Murphy, Kevin P. 2022. *Probabilistic Machine Learning: An Introduction*. MIT Press.
- 346 O’Neil, Cathy. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*.
 347 Crown.
- 348 Pawelczyk, Martin, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. 2022. “Exploring
 349 Counterfactual Explanations Through the Lens of Adversarial Examples: A Theoretical and Empirical Analysis.”
 350 In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, edited by Gustau
 351 Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, 151:4574–94. Proceedings of Machine Learning Research.
 352 PMLR. <https://proceedings.mlr.press/v151/pawelczyk22a.html>.

- 358 Poyiadzi, Rafael, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. “FACE: Feasible and
 359 Actionable Counterfactual Explanations.” In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*,
 360 344–50.
- 361 Ross, Alexis, Himabindu Lakkaraju, and Osbert Bastani. 2024. “Learning Models for Actionable Recourse.” In
 362 *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS ’21. Red
 363 Hook, NY, USA: Curran Associates Inc.
- 364 Sauer, Axel, and Andreas Geiger. 2021. “Counterfactual Generative Networks.” <https://arxiv.org/abs/2101.06046>.
- 365 Schut, Lisa, Oscar Key, Rory Mc Grath, Luca Costabello, Bogdan Sacaleanu, Yarin Gal, et al. 2021. “Generating
 366 Interpretable Counterfactual Explanations By Implicit Minimisation of Epistemic and Aleatoric Uncertainties.” In
 367 *International Conference on Artificial Intelligence and Statistics*, 1756–64. PMLR.
- 368 Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus.
 369 2013. “Intriguing Properties of Neural Networks.” <https://arxiv.org/abs/1312.6199>.
- 370 Teney, Damien, Ehsan Abbasnedjad, and Anton van den Hengel. 2020. “Learning What Makes a Difference from
 371 Counterfactual Examples and Gradient Supervision.” In *Computer Vision–ECCV 2020: 16th European Confer-
 372 ence, Glasgow, UK, August 23–28, 2020, Proceedings, Part x 16*, 580–99. Springer.
- 373 Wachter, Sandra, Brent Mittelstadt, and Chris Russell. 2017. “Counterfactual Explanations Without Opening the Black
 374 Box: Automated Decisions and the GDPR.” *Harv. JL & Tech.* 31: 841. <https://doi.org/10.2139/ssrn.3063289>.
- 375 Wilson, Andrew Gordon. 2020. “The Case for Bayesian Deep Learning.” <https://arxiv.org/abs/2001.10995>.
- 376 Wu, Tongshuang, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2021. “Polyjuice: Generating Counterfactuals
 377 for Explaining, Evaluating, and Improving Models.” In *Proceedings of the 59th Annual Meeting of the Associa-
 378 tion for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing
 379 (Volume 1: Long Papers)*, edited by Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, 6707–23. Online:
 380 Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.acl-long.523>.

381 **G Notation**

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

387 **H Technical Details of Our Approach**

388 **H.1 Generating Counterfactuals through Gradient Descent**

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

391 **H.1.1 Background**

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

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

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

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

404 **H.1.2 Convergence**

405 An important consideration when generating counterfactual explanations using gradient-based methods is how to
406 define convergence. Two common choices are to 1) perform gradient descent over a fixed number of iterations T , or
407 2) conclude the search as soon as the predicted probability for the target class has reached a pre-determined threshold,
408 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
409 black-box model, $\mathbf{M}(\mathbf{x})$.

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

414 **H.2 Protecting Mutability Constraints with Linear Classifiers**

415 In Section 3.2 we explain that to avoid penalizing implausibility that arises due to mutability constraints, we impose a
416 point mass prior on $p(\mathbf{x})$ for the corresponding feature. We argue in Section 3.2 that this approach induces models to
417 be less sensitive to immutable features and demonstrate this empirically in Section 4. Below we derive the analytical
418 results in Proposition 3.1.

419 *Proof.* Let d_{mtbl} and d_{immtbl} denote some mutable and immutable feature, respectively. Suppose that $\mu_{y^-, d_{\text{immtbl}}} <$
420 $\mu_{y^+, d_{\text{immtbl}}}$ 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,
421 we assume that the immutable feature tends to take lower values for samples in the non-target class y^- than in the
422 target class y^+ . We assume the opposite to hold for the mutable feature.

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

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \mathbf{x}^\top \Sigma^{-1} (\mu_k - \mu_m) + \text{const} \quad (4)$$

425 By independence of x_1, \dots, x_D , the full log-likelihood ratio decomposes into:

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^D \frac{\mu_{k,d} - \mu_{m,d}}{\sigma_d^2} x_d + \text{const} \quad (5)$$

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

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^D (\theta_{k,d} - \theta_{m,d}) x_d + \text{const} \quad (6)$$

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

428 Based on Equation 5 and Equation 6 we can identify that $(\mu_{k,d} - \mu_{m,d}) \propto (\theta_{k,d} - \theta_{m,d})$ under the assumptions we
429 made above. Hence, we have that $(\theta_{y^-, d_{\text{immtbl}}} - \theta_{y^+, d_{\text{immtbl}}}) < 0$ and $(\theta_{y^-, d_{\text{mtbl}}} - \theta_{y^+, d_{\text{mtbl}}}) > 0$

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

$$\frac{\partial}{\partial \theta_{y^+, d}} (\text{div}(\mathbf{x}, \mathbf{x}', \mathbf{y}; \theta)) = \frac{\partial}{\partial \theta_{y^+, d}} ((-\mathbf{M}_\theta(\mathbf{x})[y^+]) - (-\mathbf{M}_\theta(\mathbf{x}')[y^+])) = x'_d - x_d \quad (7)$$

433 and equal to zero everywhere else.

434 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.
435 Similarly, we are more likely to have $(x'_{d_{\text{mtbl}}} - x_{d_{\text{mtbl}}}) > 0$ since $(\mu_{y^-, d_{\text{mtbl}}} > \mu_{y^+, d_{\text{mtbl}}})$.

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

440 By the same rationale, the contrastive divergence penalty can generally be expected to increase $\theta_{y^-, d_{\text{mtbl}}}$ exacerbating
441 $(\theta_{y^-, d_{\text{mtbl}}} - \theta_{y^+, d_{\text{mtbl}}}) > 0$. In words, this has the effect of making the classifier more sensitive to the mutable feature, in
442 that it would be more likely to predict class y^- as opposed to y^+ for higher values of d_{mtbl} .

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

446 H.3 Domain Constraints

447 We apply domain constraints on counterfactuals during training and evaluation. There are at least two good reasons for
448 doing so. Firstly, within the context of explainability and algorithmic recourse, real-world attributes are often domain
449 constrained: the *age* feature, for example, is lower bounded by zero and upper bounded by the maximum human
450 lifespan. Secondly, domain constraints help mitigate training instabilities commonly associated with energy-based
451 modelling (Grathwohl et al. 2020; Altmeyer et al. 2024).

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

$$\begin{aligned} x_d^{\text{LB}} &= \arg \min_{x_d} \{\mu_d - n_{\sigma_d} \sigma_d, \arg \min_{x_d} x_d\} \\ x_d^{\text{UB}} &= \arg \max_{x_d} \{\mu_d + n_{\sigma_d} \sigma_d, \arg \max_{x_d} x_d\} \end{aligned} \quad (8)$$

455 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
456 higher values and hence wider bounds may be appropriate depending on the application.

457 **H.4 Training Details**

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

460 **I Detailed Results**461 **I.1 Qualitative Findings for Image Data**

Note

Figure A2 shows much more plausible (faithful) counterfactuals for a model with CT than the model with conventional training (Figure A3). In fact, this is not even using *ECCo+* and still showing better results than the best results we achieved in our AAAI paper for JEM ensembles.

462

463 **I.2 Grid Searches**

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

- 466 1. **Generator Parameters** (Section I.2.2): Investigates the effect of changing hyperparameters that affect the
 467 counterfactual outcomes during the training phase.
- 468 2. **Penalty Strengths** (Section I.2.3): Investigates the effect of changing the penalty strengths in our proposed
 469 objective (Equation 1).
- 470 3. **Other Parameters** (Section I.2.4): Investigates the effect of changing other training parameters, including
 471 the total number of generated counterfactuals in each epoch.

472 We begin by summarizing the high-level findings in Section I.2.1. For each of the categories, Section I.2.2 to Sec-
 473 tion I.2.4 then present all details including the parameter grids, average performance outcomes and key evaluation
 474 metrics for the generated counterfactuals.

475 **I.2.1 High-Level Findings**

476 Overall, we find that Counterfactual Training (CT) achieves its key objectives consistently across all hyperparameter
 477 settings and also broadly across datasets. We do observe strong sensitivity to certain hyperparameters, but clear and
 478 manageable patterns emerge in those cases. With respect to the underlying training data distribution, we find that
 479 outcomes for CT seem to depend to some degree on class separability.

480 **I.2.1.1 Predictive Performance** We find that CT is associated with little to no decrease in average predictive
 481 performance for our synthetic datasets: test accuracy and F1-scores decrease by at most ~1 percentage point, but
 482 generally much less (Table A1, Table A2, Table A3). Variation across hyperparameters is negligible as indicated by
 483 small standard deviations for these metrics across the board.

484 **I.2.1.2 Key Counterfactual Outcomes**485 **I.2.2 Generator Parameters**

486 The hyperparameter grid with varying generator parameters during training is shown in Note 1. The corresponding
 487 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_{energy} : 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*

488

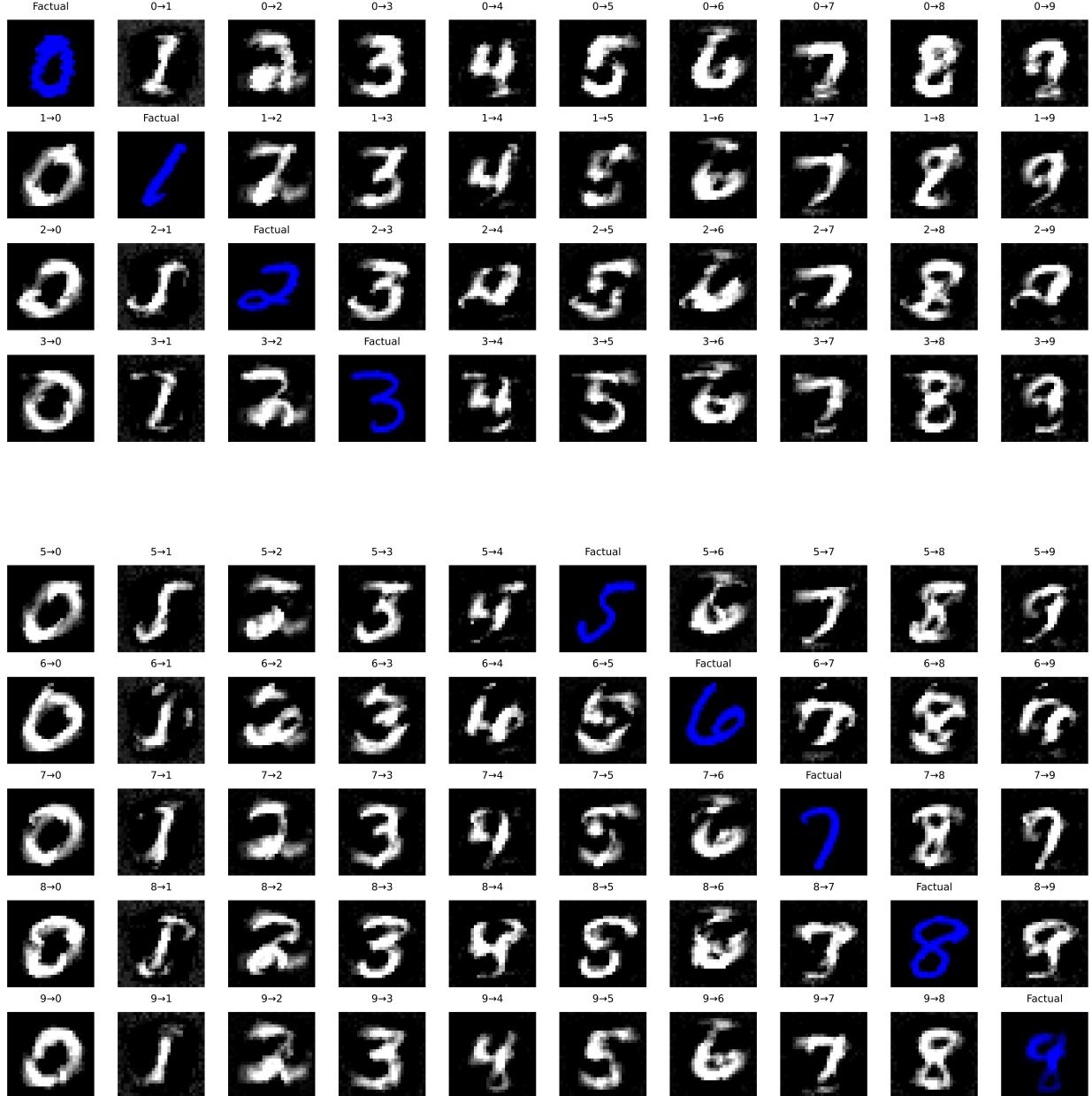


Figure A2: Counterfactual images for *MLP* with counterfactual training. The underlying generator, *ECCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

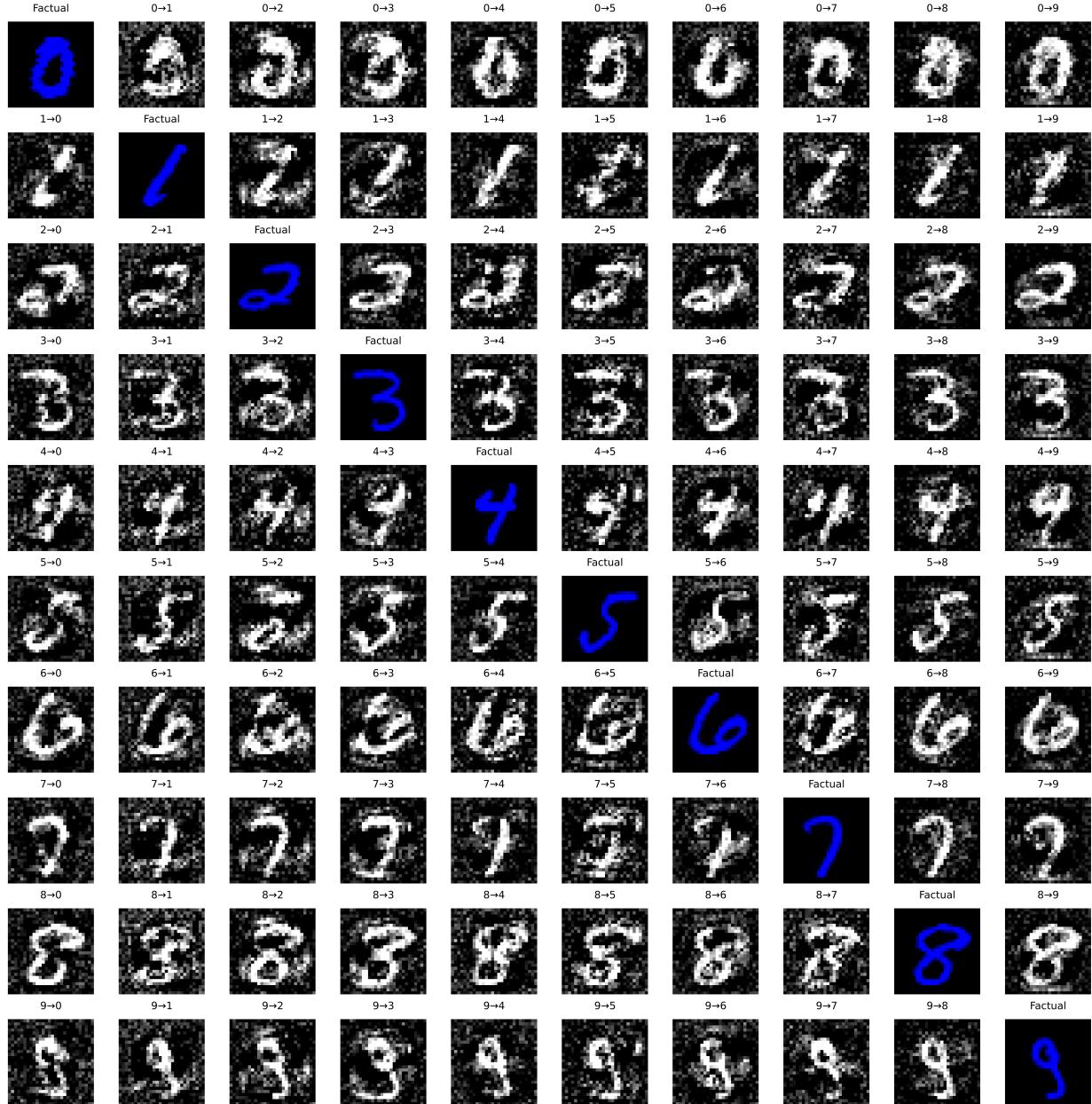


Figure A3: Counterfactual images for *MLP* with conventional training. The underlying generator, *ECCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

Note 2: Evaluation Phase

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

489

490 **I.2.2.1 Accuracy**

Table A1: 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
Circles	Accuracy	Full	0.997	0.00309
Circles	Accuracy	Vanilla	0.998	0.000557
Circles	F1-score	Full	0.997	0.00309
Circles	F1-score	Vanilla	0.998	0.000558
Lin Sep	Accuracy	Full	0.999	0.00201
Lin Sep	Accuracy	Vanilla	1	0
Lin Sep	F1-score	Full	0.999	0.00201
Lin Sep	F1-score	Vanilla	1	0
Moons	Accuracy	Full	0.999	0.000696
Moons	Accuracy	Vanilla	1	0.00111
Moons	F1-score	Full	0.999	0.000696
Moons	F1-score	Vanilla	1	0.00111
Over	Accuracy	Full	0.915	0.00477
Over	Accuracy	Vanilla	0.917	0.00123
Over	F1-score	Full	0.915	0.00478
Over	F1-score	Vanilla	0.917	0.00124

491 **I.2.2.2 Plausibility** The results with respect to the plausibility measure are shown in Figure A4 to Figure A7.

492 **I.2.2.3 Cost** The results with respect to the cost measure are shown in Figure A8 to Figure A11.

493 **I.2.3 Penalty Strengths**

494 The hyperparameter grid with varying penalty strengths during training is shown in Note 3. The corresponding evalua-
495 tion 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`

496

Note 4: Evaluation Phase

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

497

498 **I.2.3.1 Accuracy**

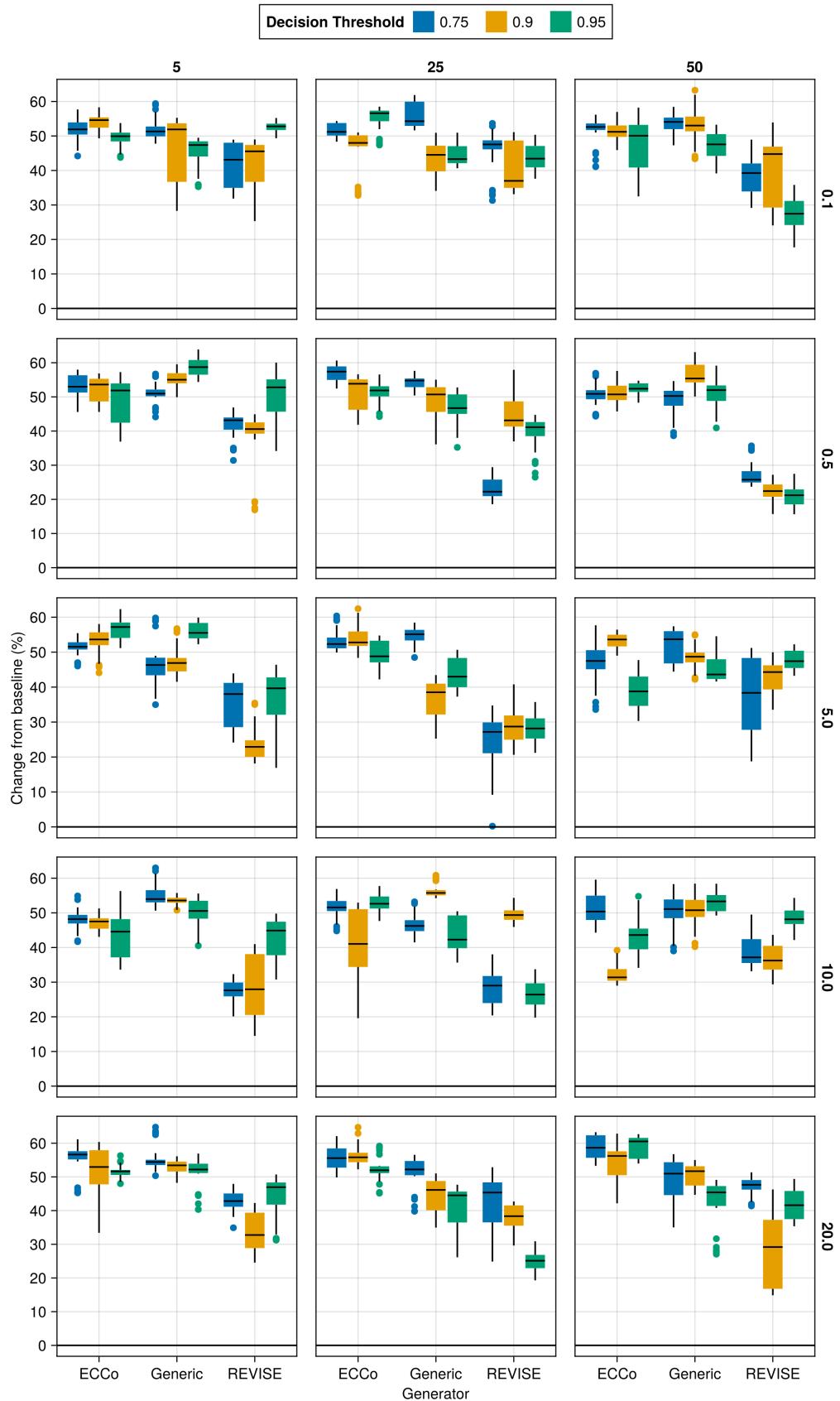


Figure A4: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

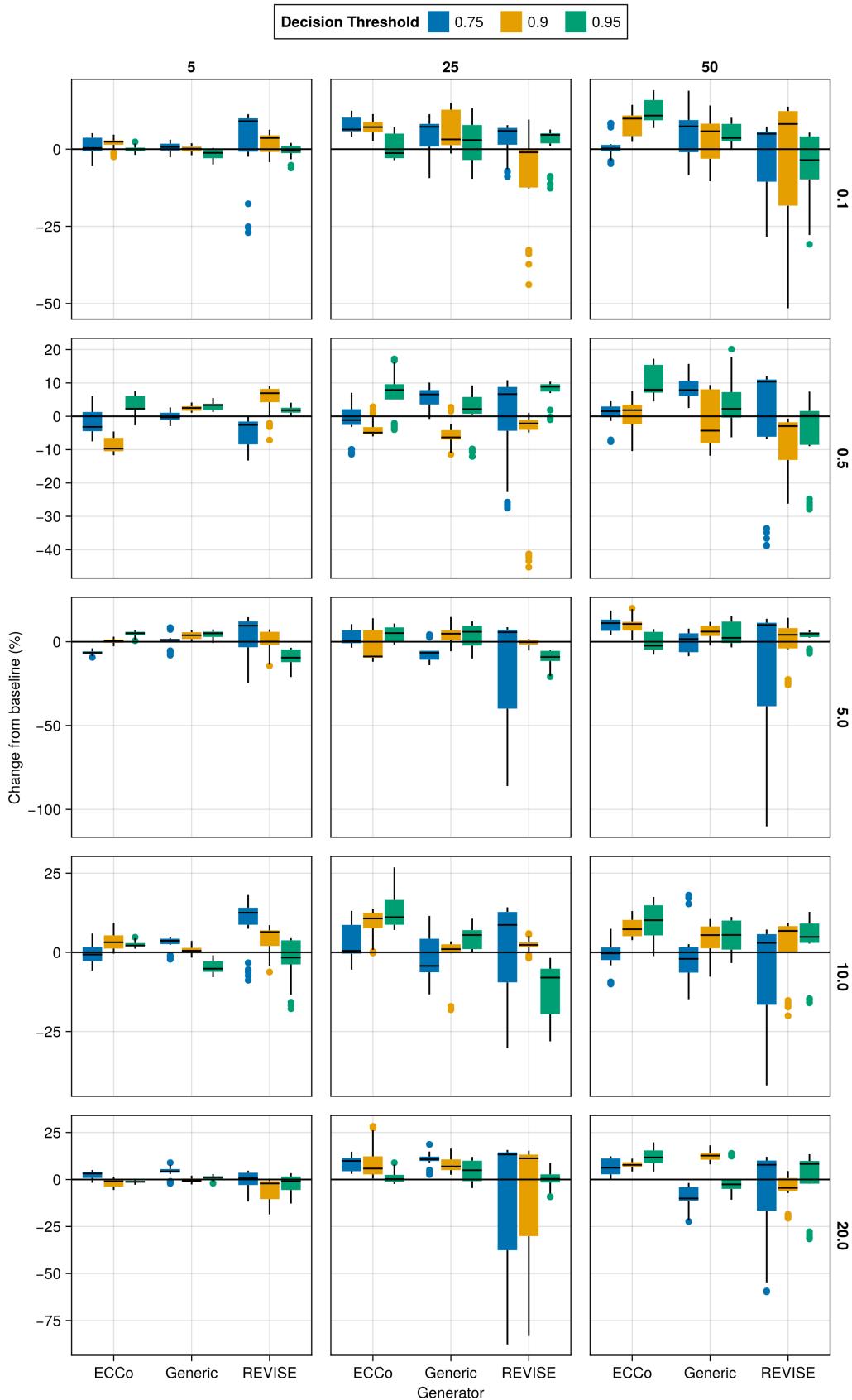


Figure A5: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

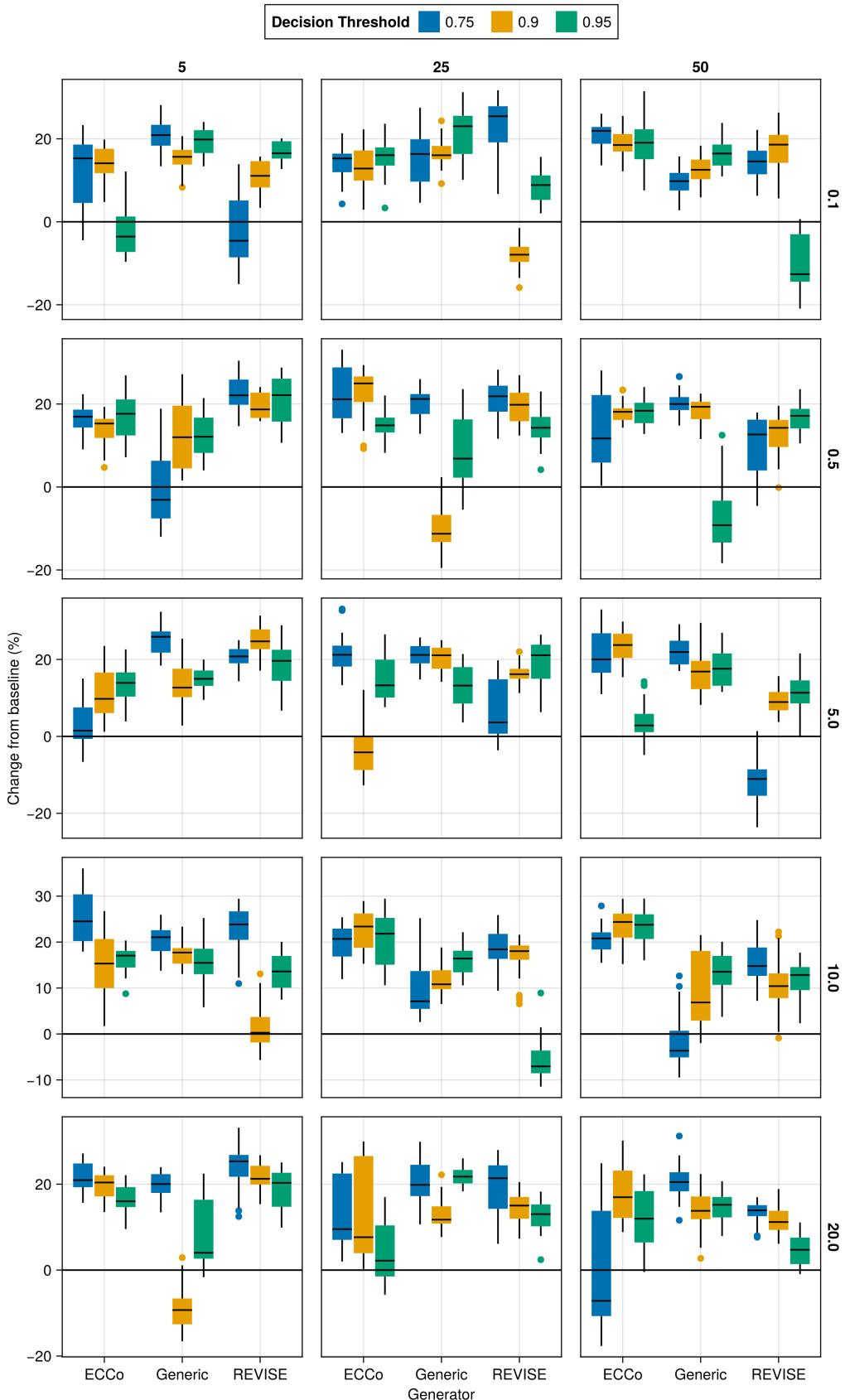


Figure A6: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

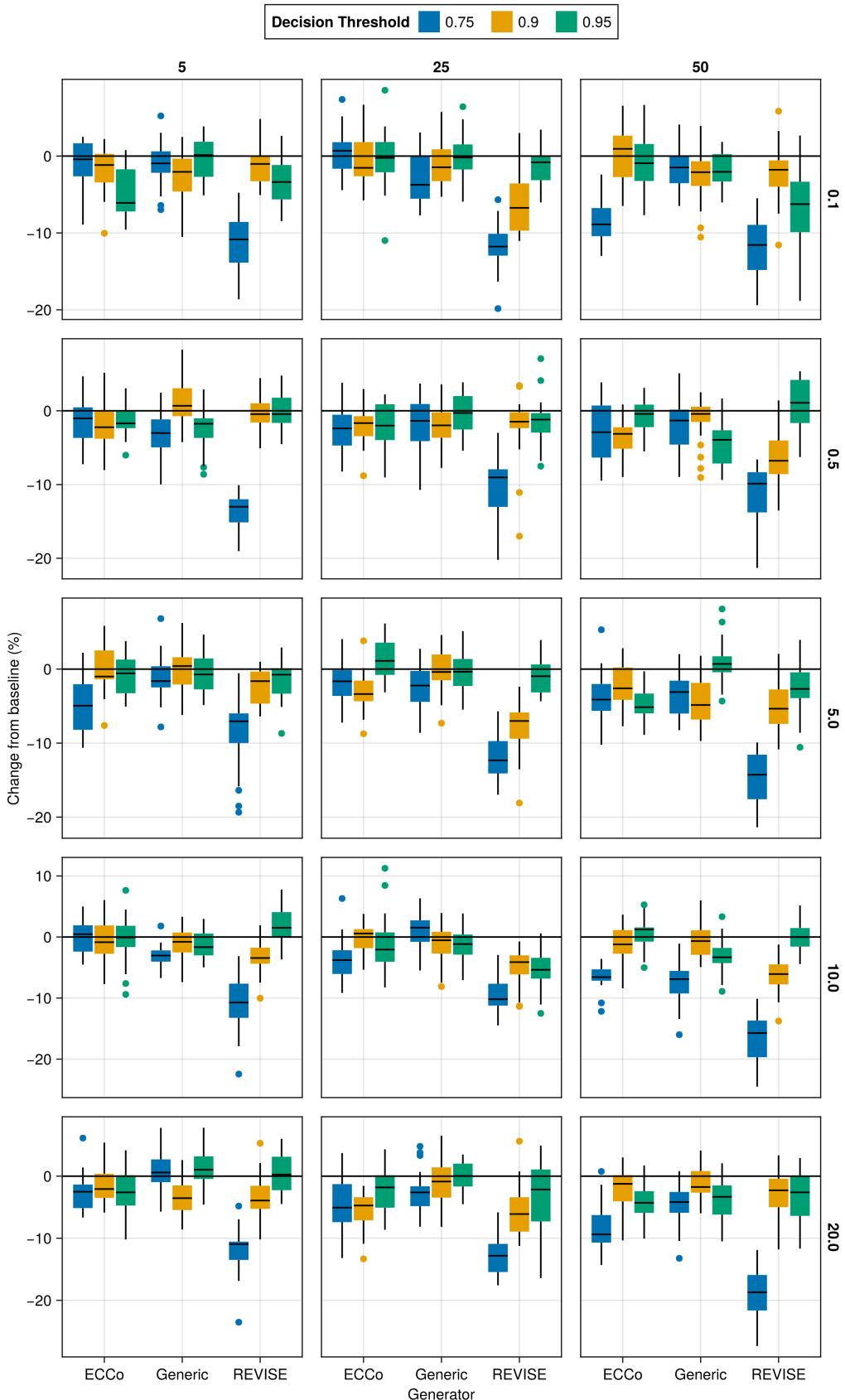


Figure A7: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

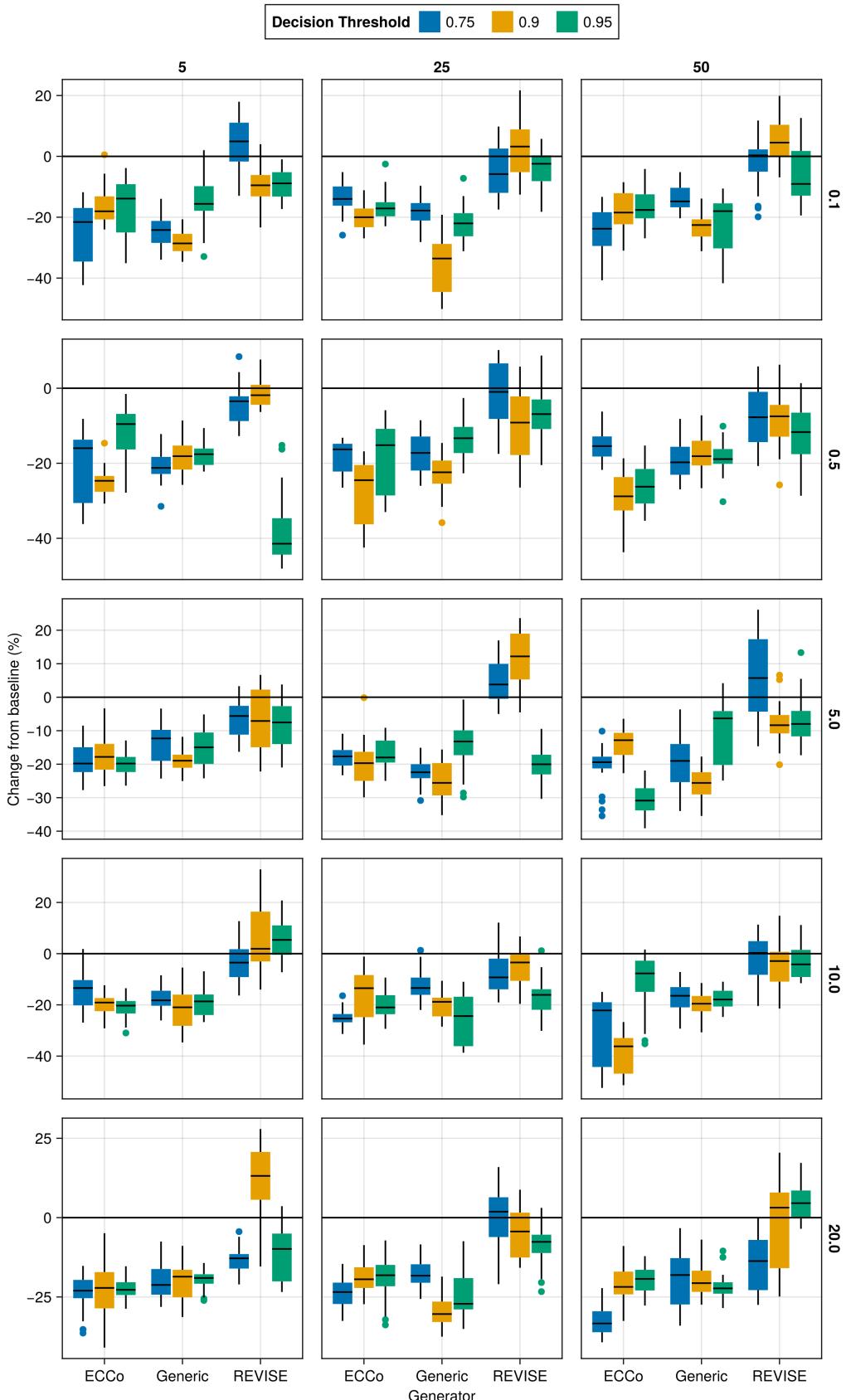


Figure A8: Average outcomes for the cost measure across hyperparameters. Data: Circles.

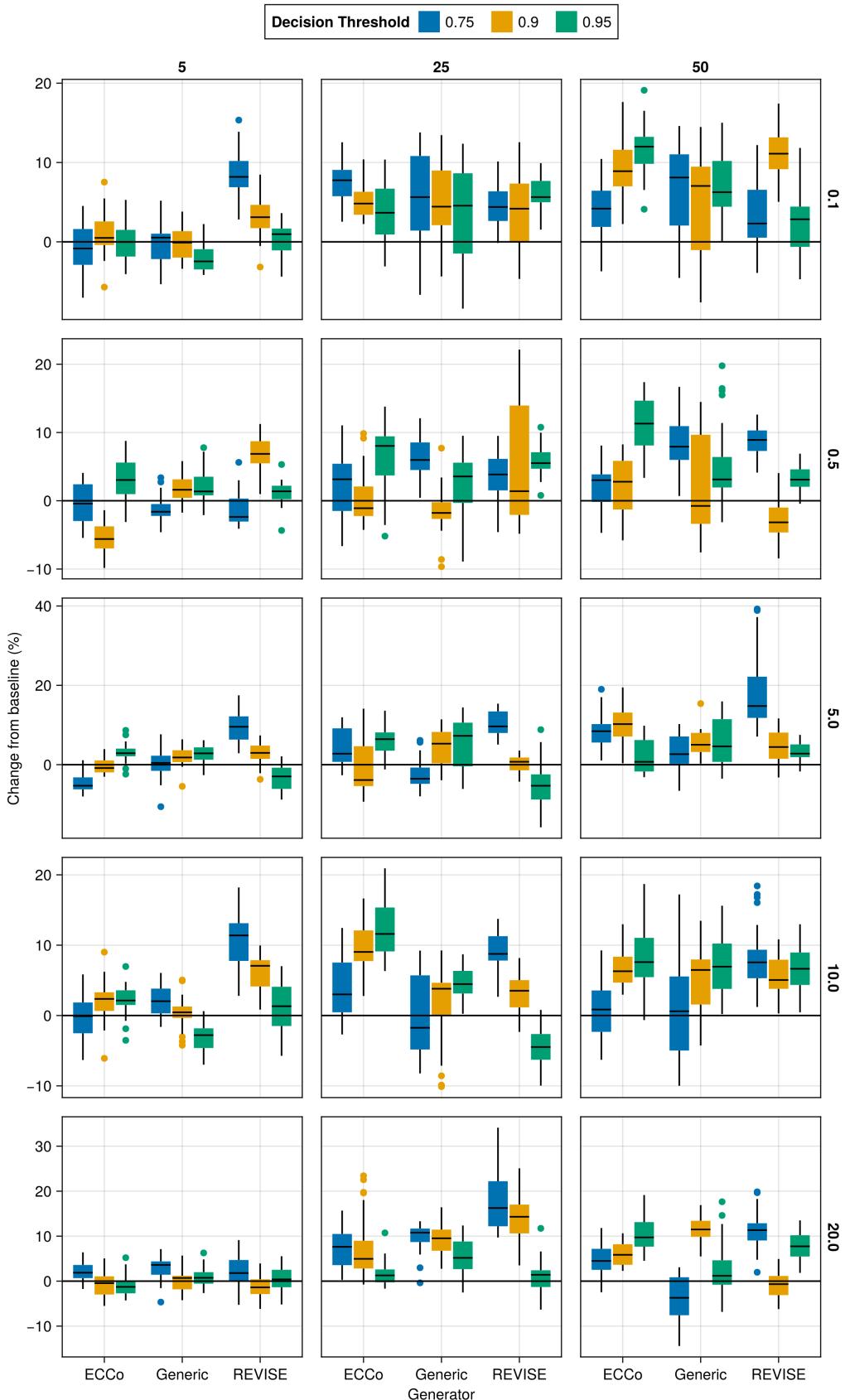


Figure A9: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

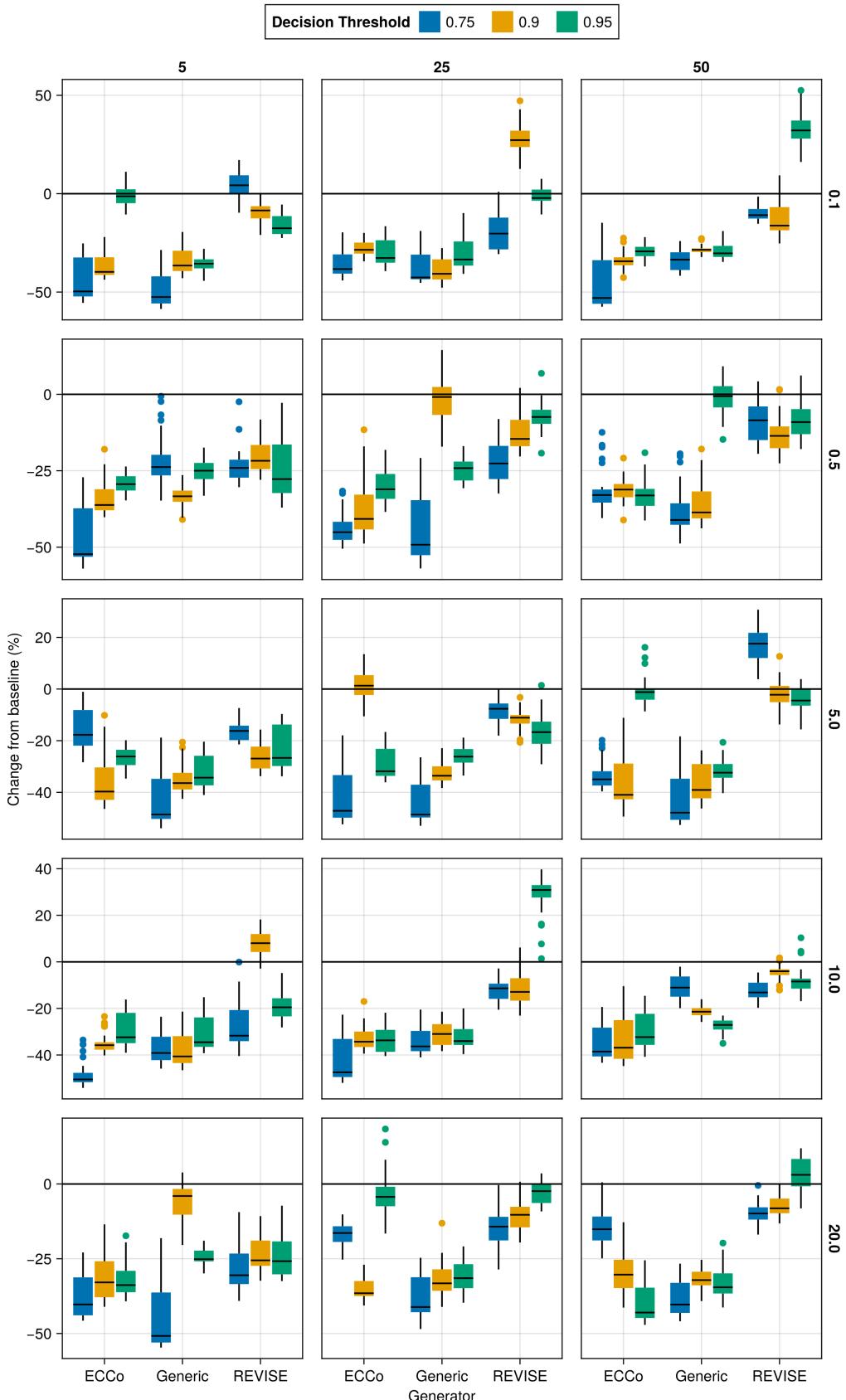


Figure A10: Average outcomes for the cost measure across hyperparameters. Data: Moons.

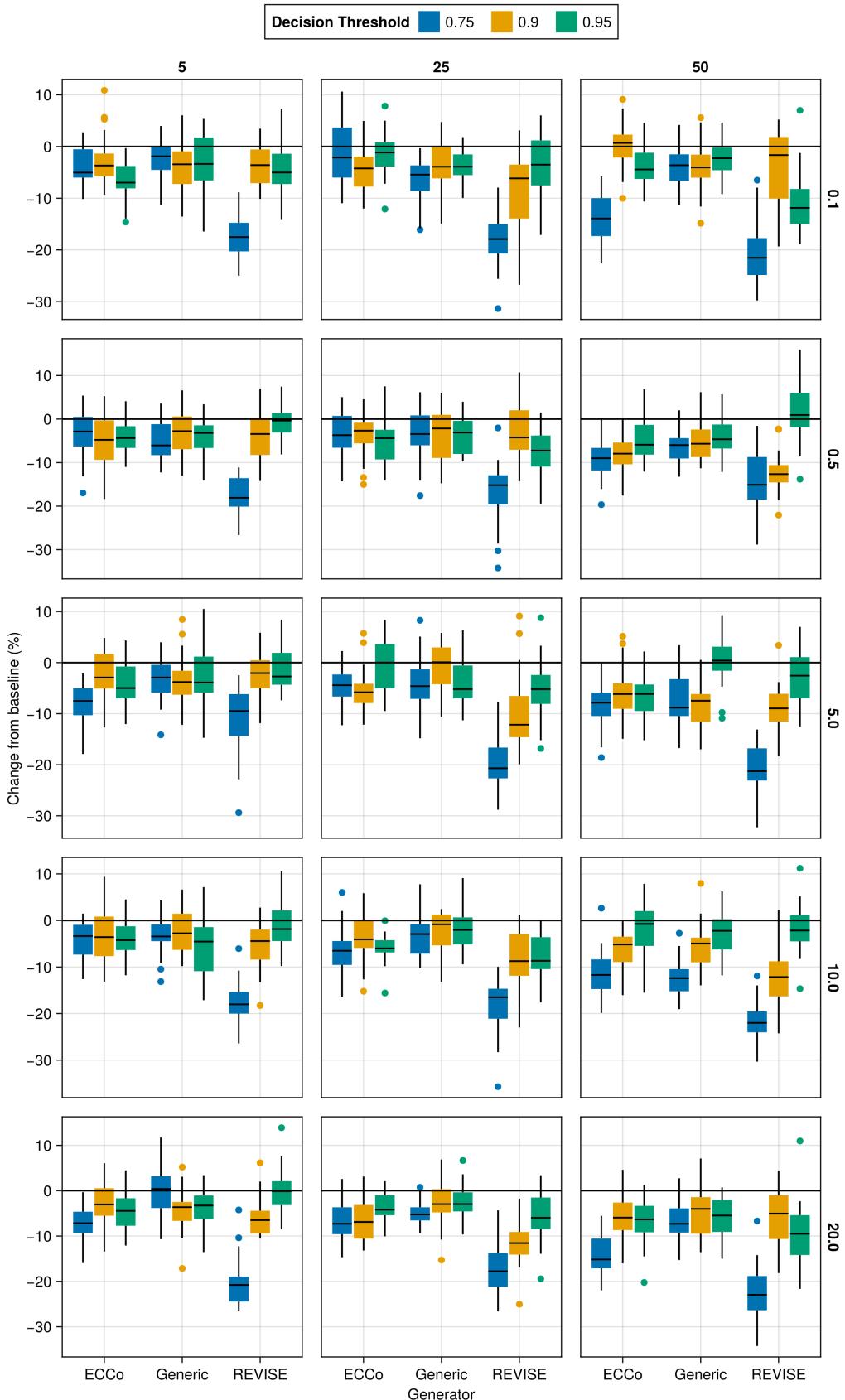


Figure A11: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.

Table A2: 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
Circles	Accuracy	Full	0.994	0.0144
Circles	Accuracy	Vanilla	0.998	0.000875
Circles	F1-score	Full	0.994	0.0145
Circles	F1-score	Vanilla	0.998	0.000875
Lin Sep	Accuracy	Full	0.998	0.00772
Lin Sep	Accuracy	Vanilla	1	0
Lin Sep	F1-score	Full	0.998	0.00773
Lin Sep	F1-score	Vanilla	1	0
Moons	Accuracy	Full	0.987	0.0351
Moons	Accuracy	Vanilla	0.998	0.0101
Moons	F1-score	Full	0.987	0.0352
Moons	F1-score	Vanilla	0.998	0.0102
Over	Accuracy	Full	0.911	0.0217
Over	Accuracy	Vanilla	0.916	0.00236
Over	F1-score	Full	0.911	0.0219
Over	F1-score	Vanilla	0.916	0.00236

499 **I.2.3.2 Plausibility** The results with respect to the plausibility measure are shown in Figure A12 to Figure A15.

500 **I.2.3.3 Cost** The results with respect to the cost measure are shown in Figure A16 to Figure A19.

501 **I.2.4 Other Parameters**

502 The hyperparameter grid with other varying training parameters is shown in Note 5. The corresponding evaluation
503 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`

504

Note 6: Evaluation Phase

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

505

506 **I.2.4.1 Accuracy**

Table A3: 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
Circles	Accuracy	Full	0.995	0.00431
Circles	Accuracy	Vanilla	0.998	0.000566
Circles	F1-score	Full	0.995	0.00432
Circles	F1-score	Vanilla	0.998	0.000566
Lin Sep	Accuracy	Full	0.999	0.00231

Continuing table below.

Dataset	Variable	Objective	Mean	Std
Lin Sep	Accuracy	Vanilla	1	0
Lin Sep	F1-score	Full	0.999	0.00231
Lin Sep	F1-score	Vanilla	1	0
Moons	Accuracy	Full	0.996	0.0136
Moons	Accuracy	Vanilla	0.988	0.022
Moons	F1-score	Full	0.996	0.0136
Moons	F1-score	Vanilla	0.988	0.022
Over	Accuracy	Full	0.914	0.00563
Over	Accuracy	Vanilla	0.918	0.00116
Over	F1-score	Full	0.914	0.0057
Over	F1-score	Vanilla	0.918	0.00116

507 **I.2.4.2 Plausibility** The results with respect to the plausibility measure are shown in Figure A20 to Figure A23.

508 **I.2.4.3 Cost** The results with respect to the cost measure are shown in Figure A24 to Figure A27.

509 **J Computation Details**

510 **J.1 Hardware**

511 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. 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.

520 **J.2 Software**

521 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, et al. 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).

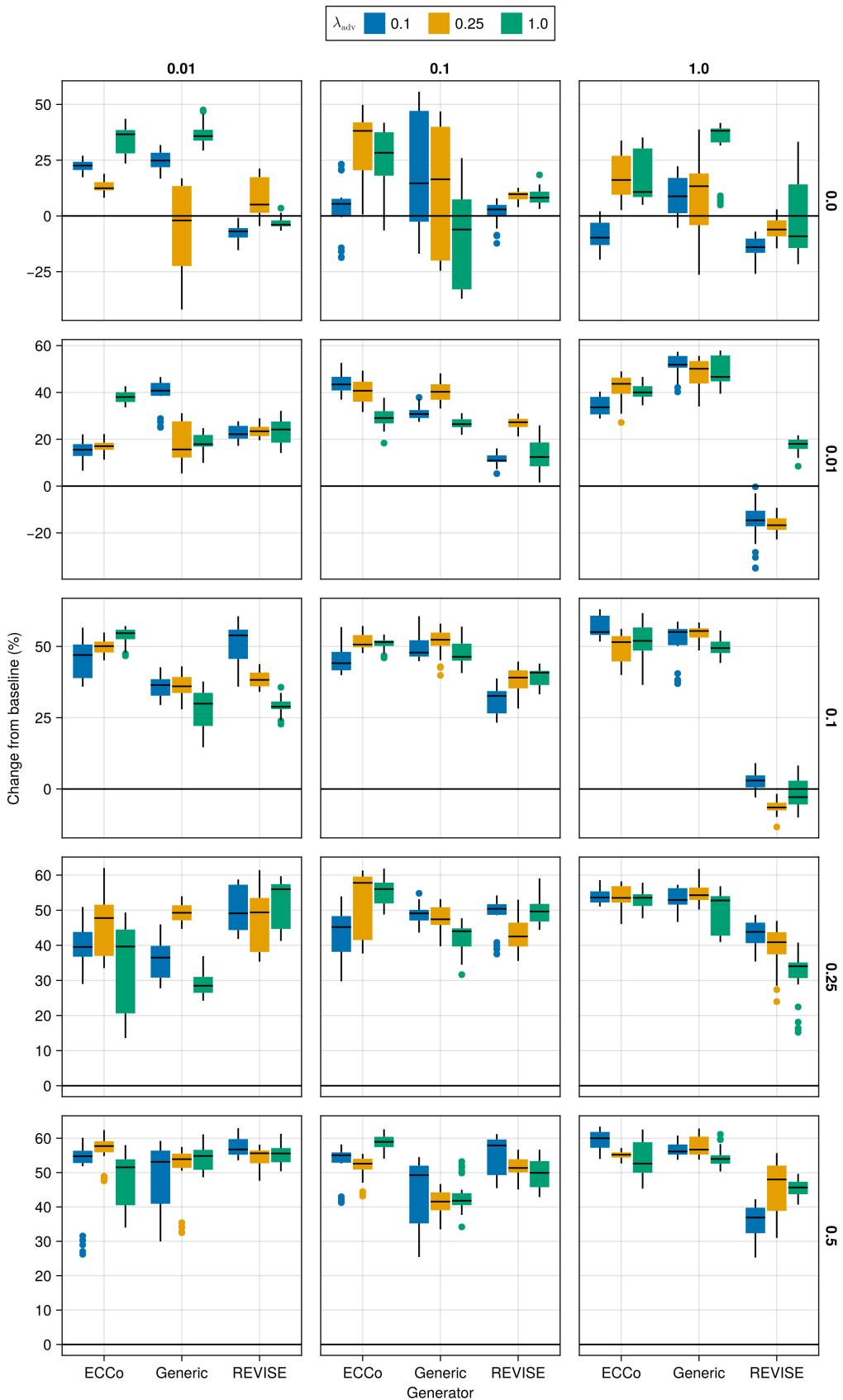


Figure A12: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

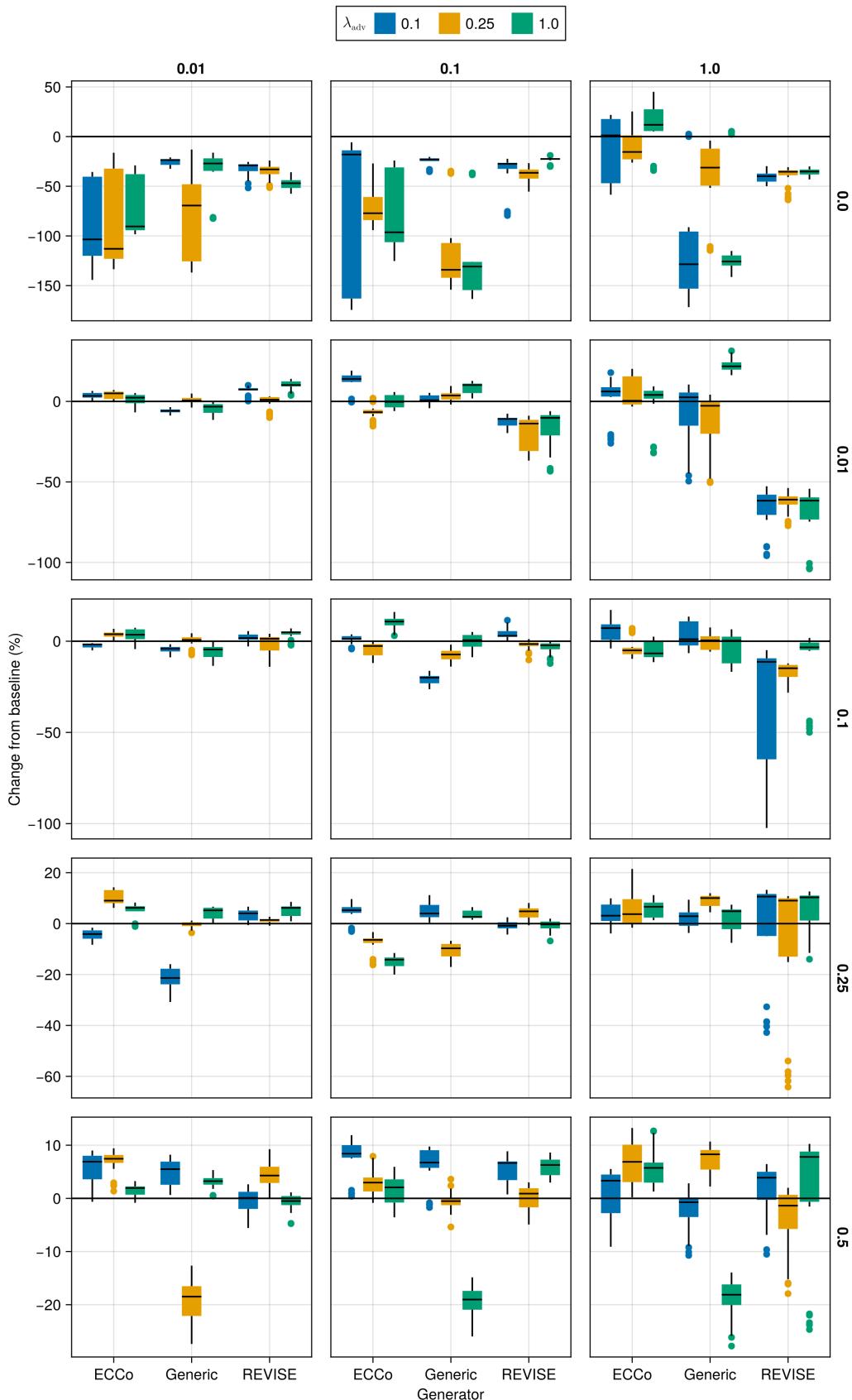


Figure A13: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

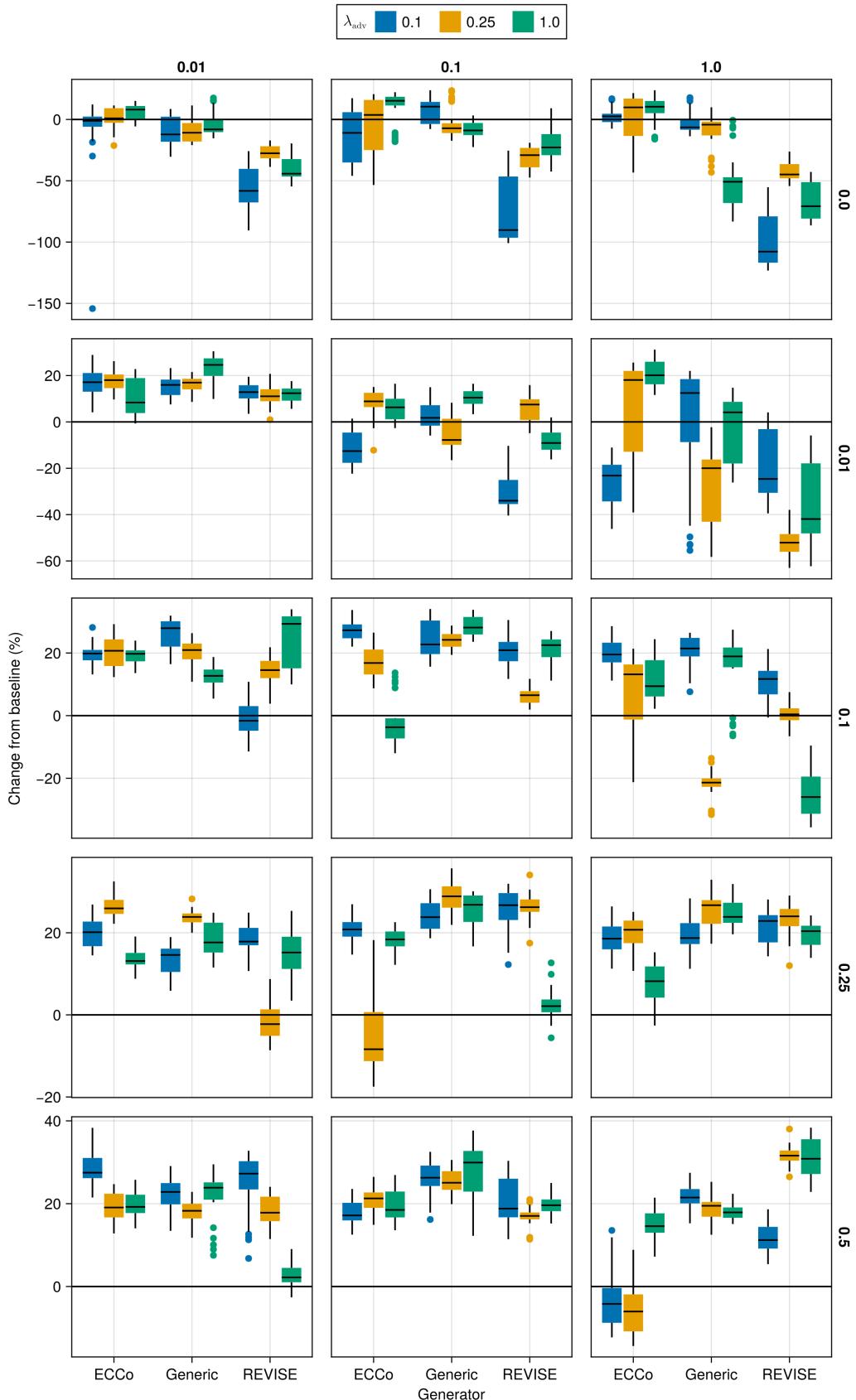


Figure A14: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

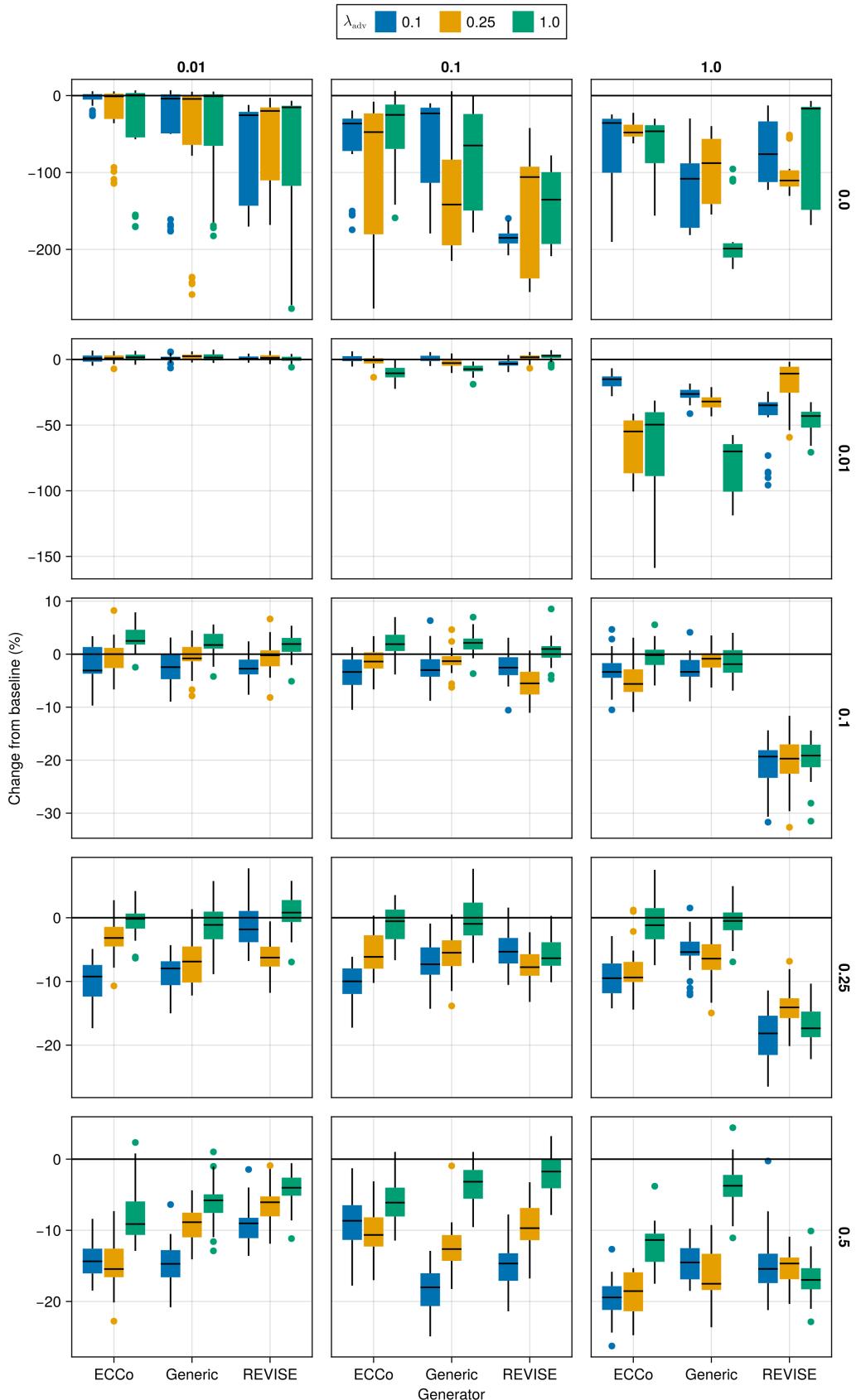


Figure A15: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

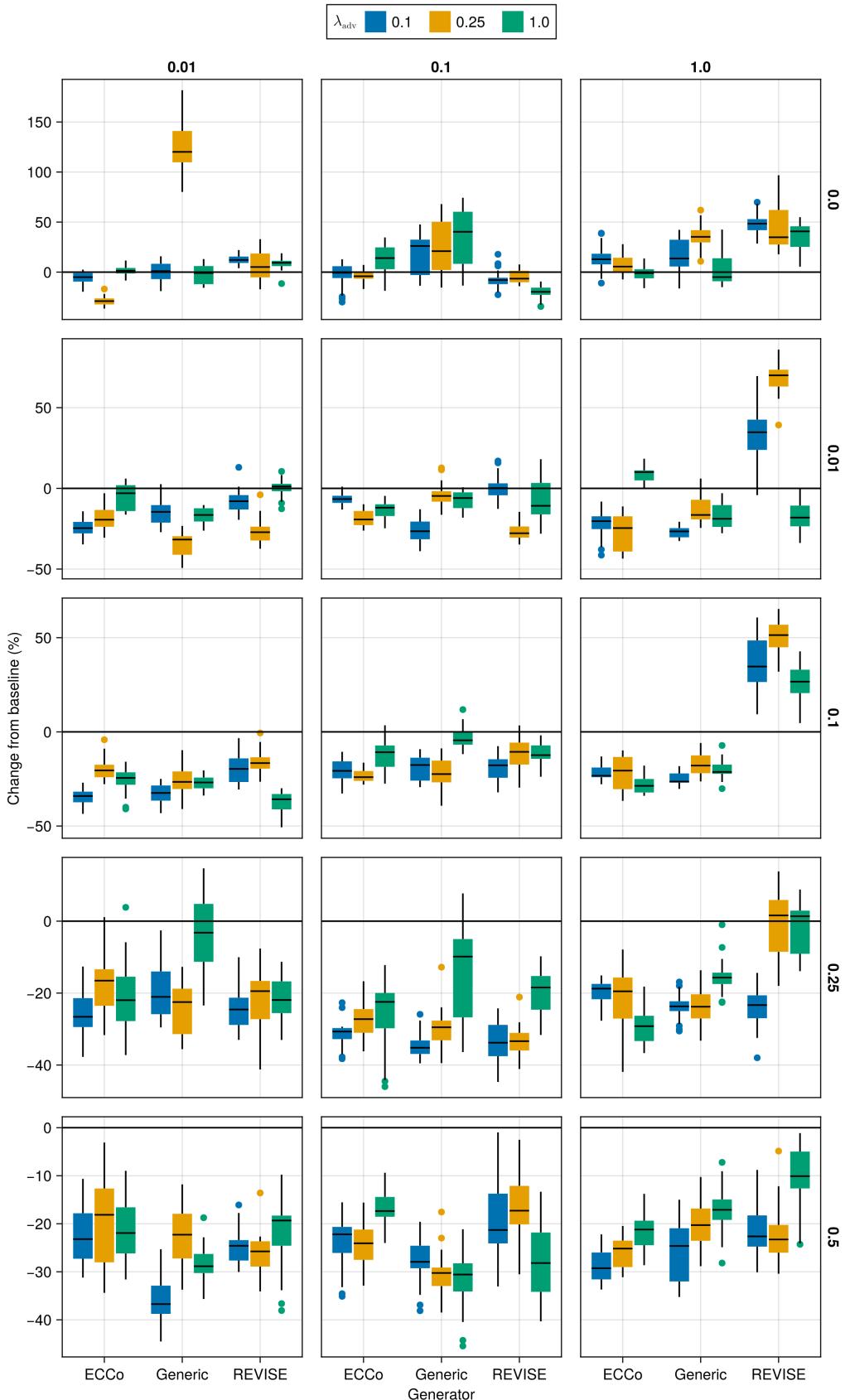


Figure A16: Average outcomes for the cost measure across hyperparameters. Data: Circles.

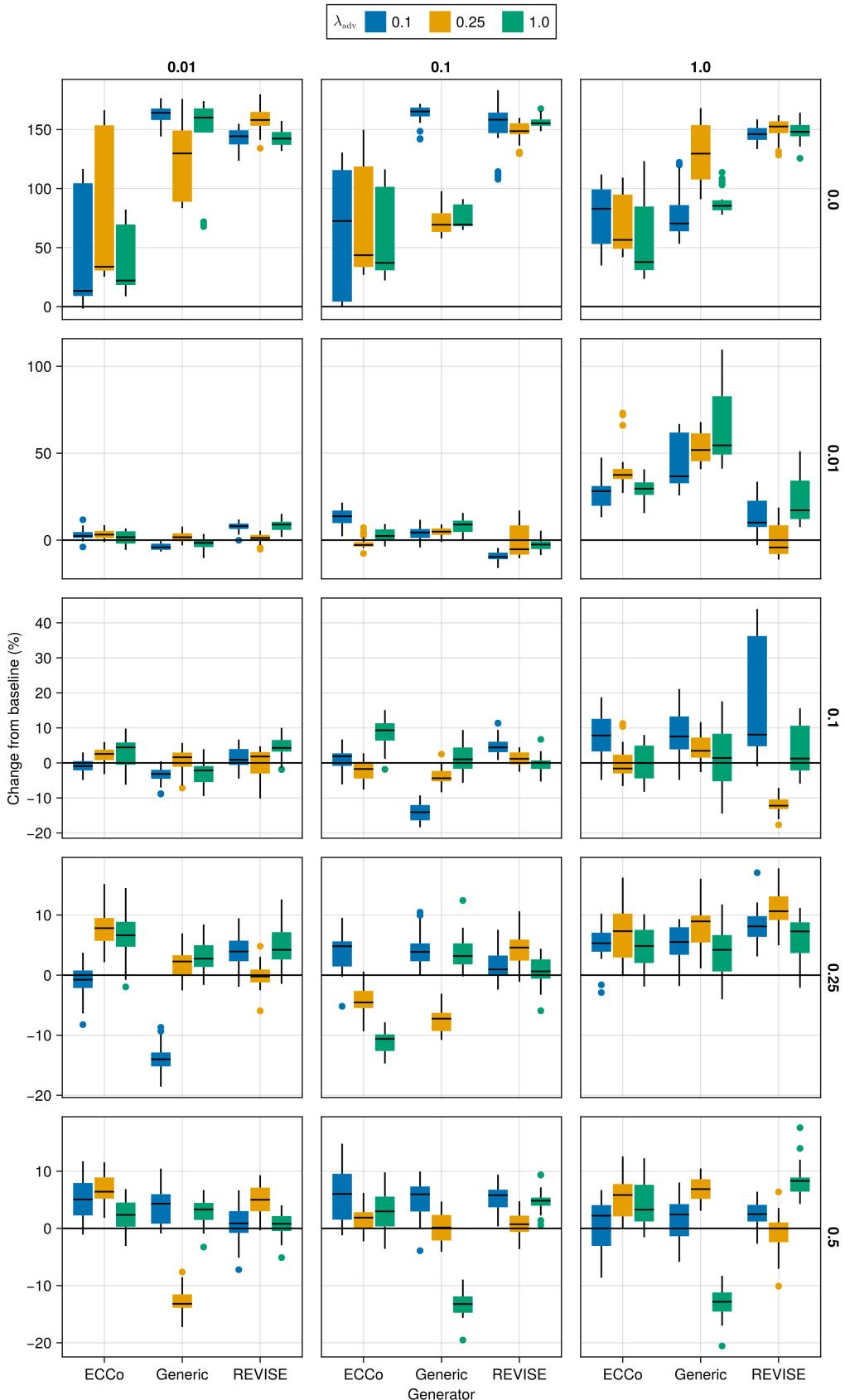


Figure A17: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

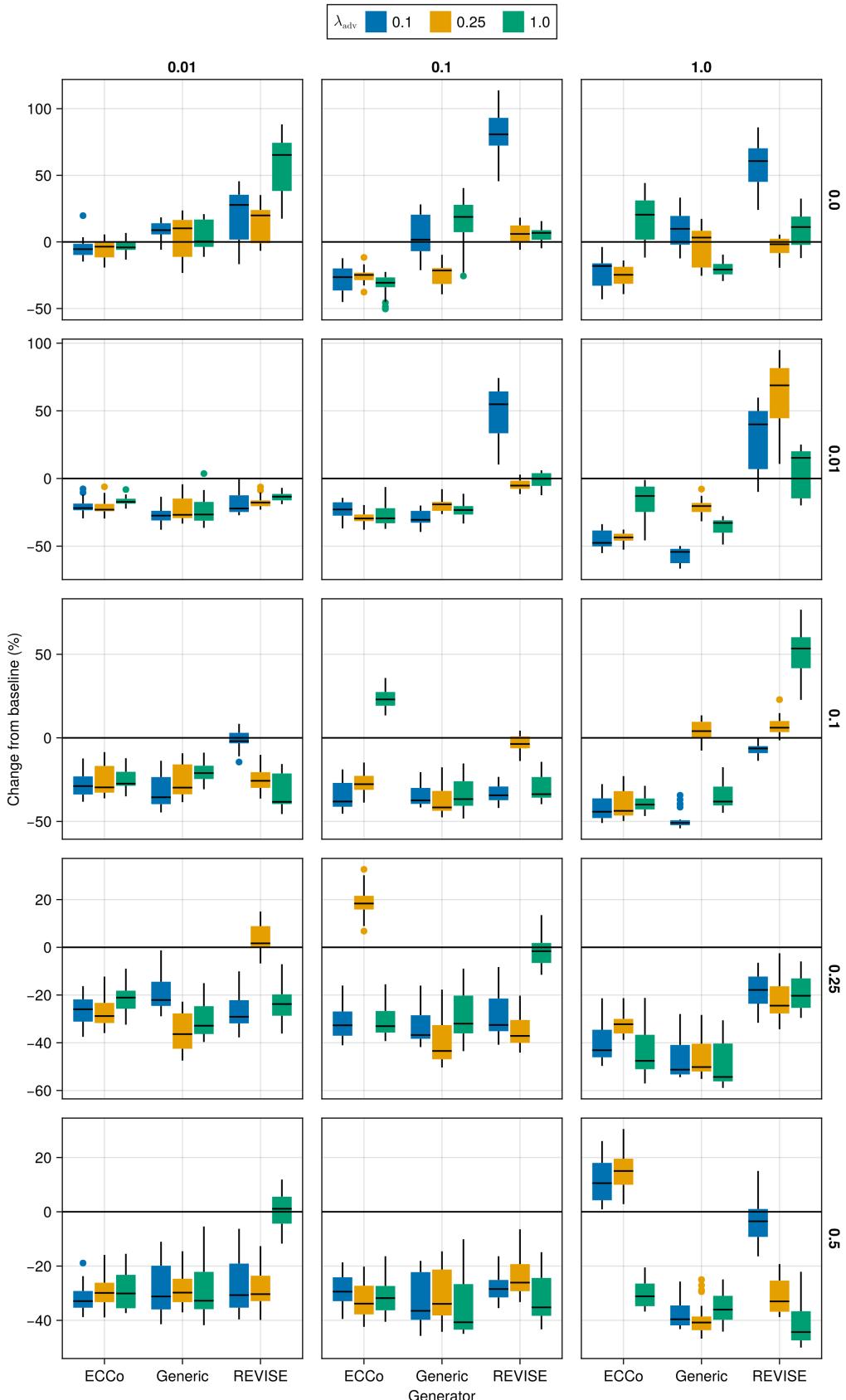


Figure A18: Average outcomes for the cost measure across hyperparameters. Data: Moons.

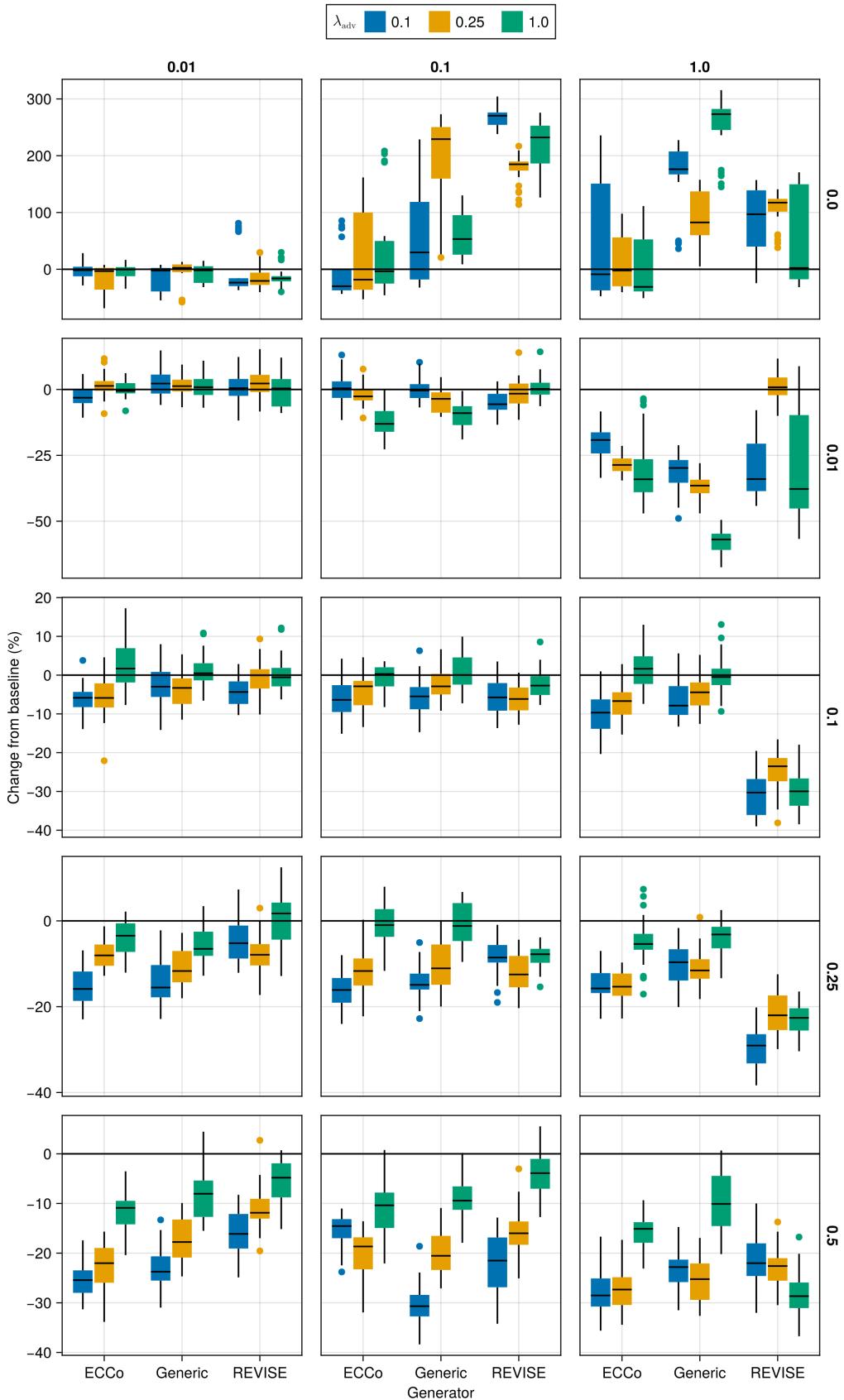


Figure A19: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.

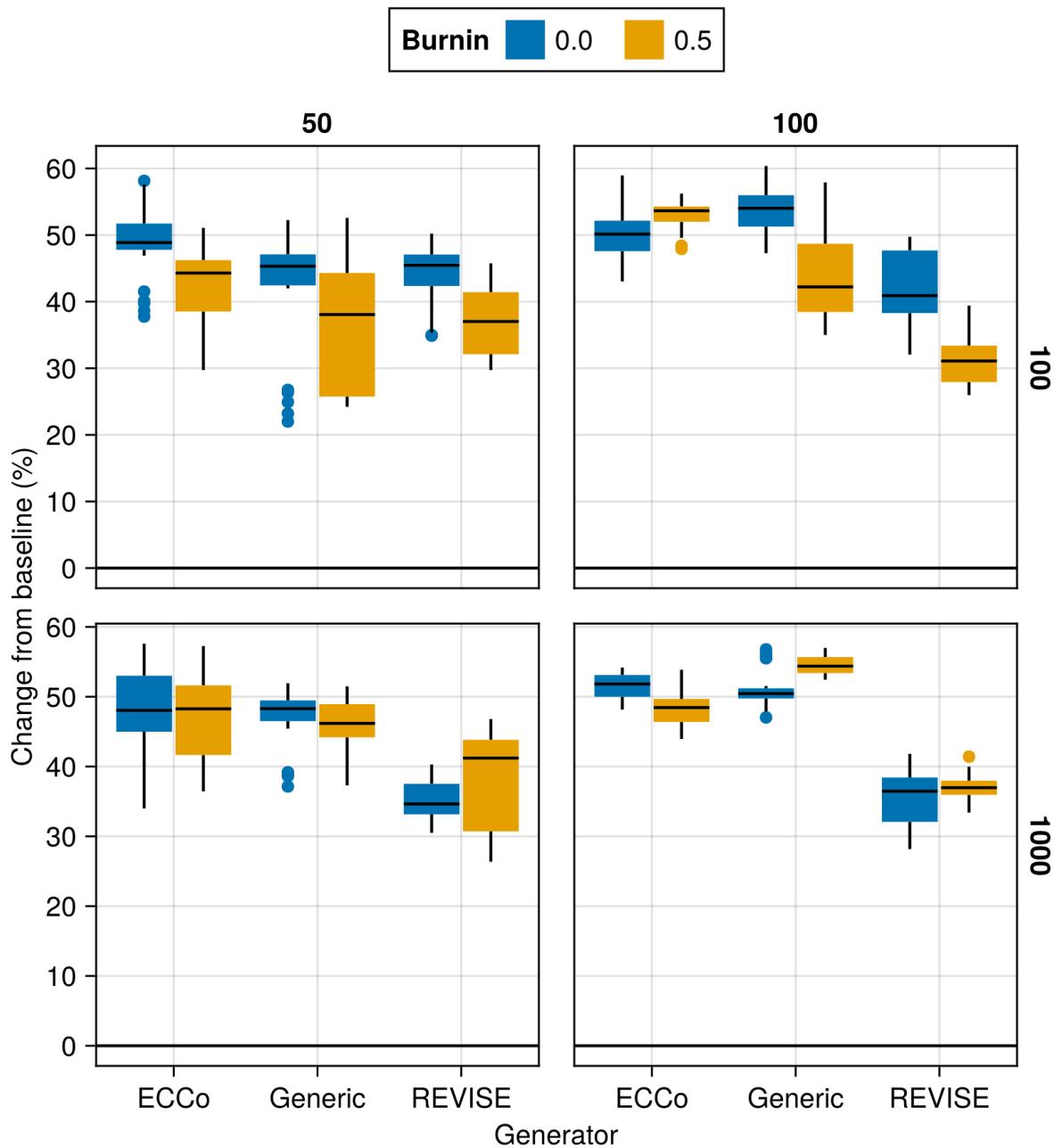


Figure A20: Average outcomes for the plausibility measure across hyperparameters. Data: Circles.

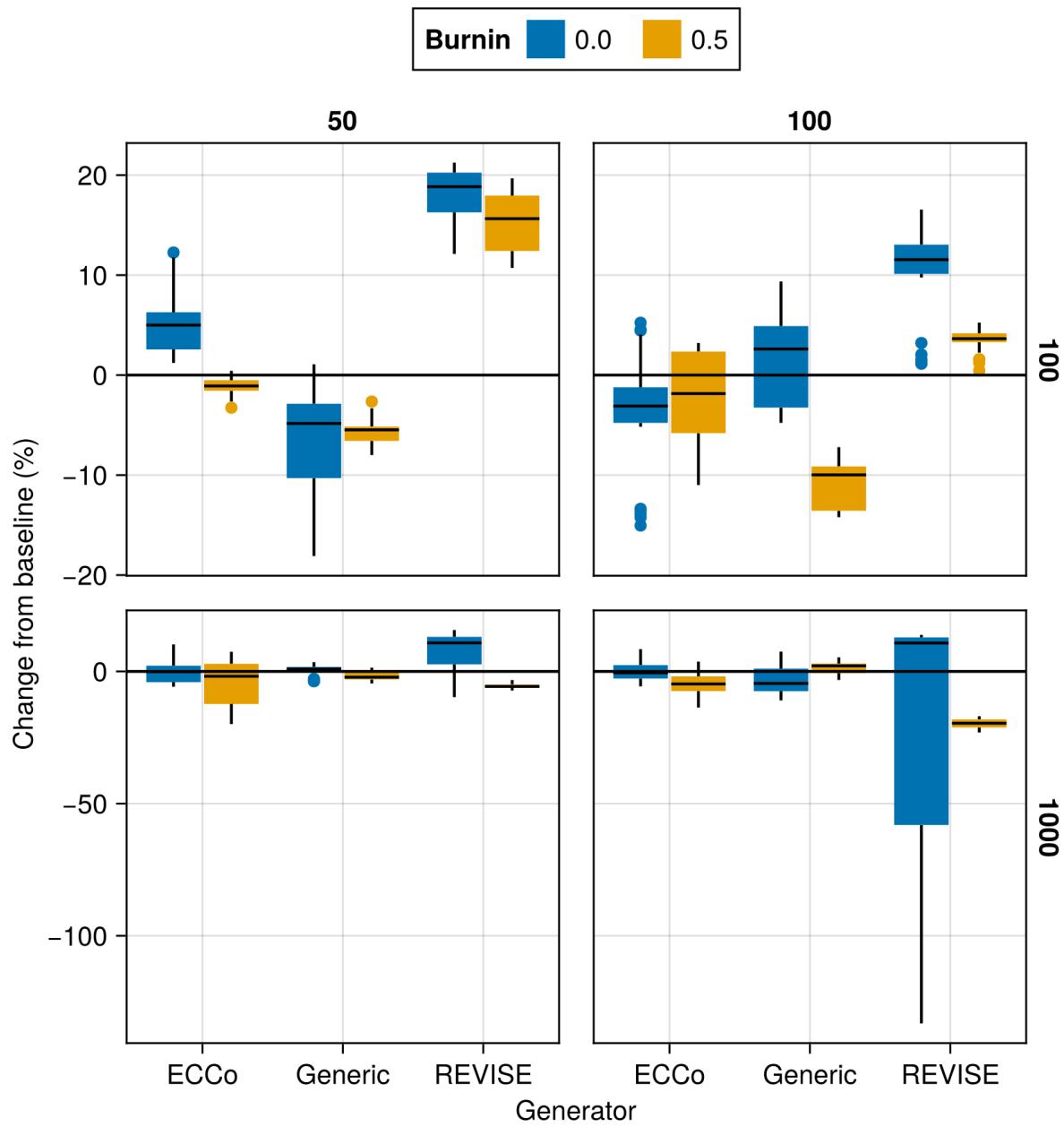


Figure A21: Average outcomes for the plausibility measure across hyperparameters. Data: Linearly Separable.

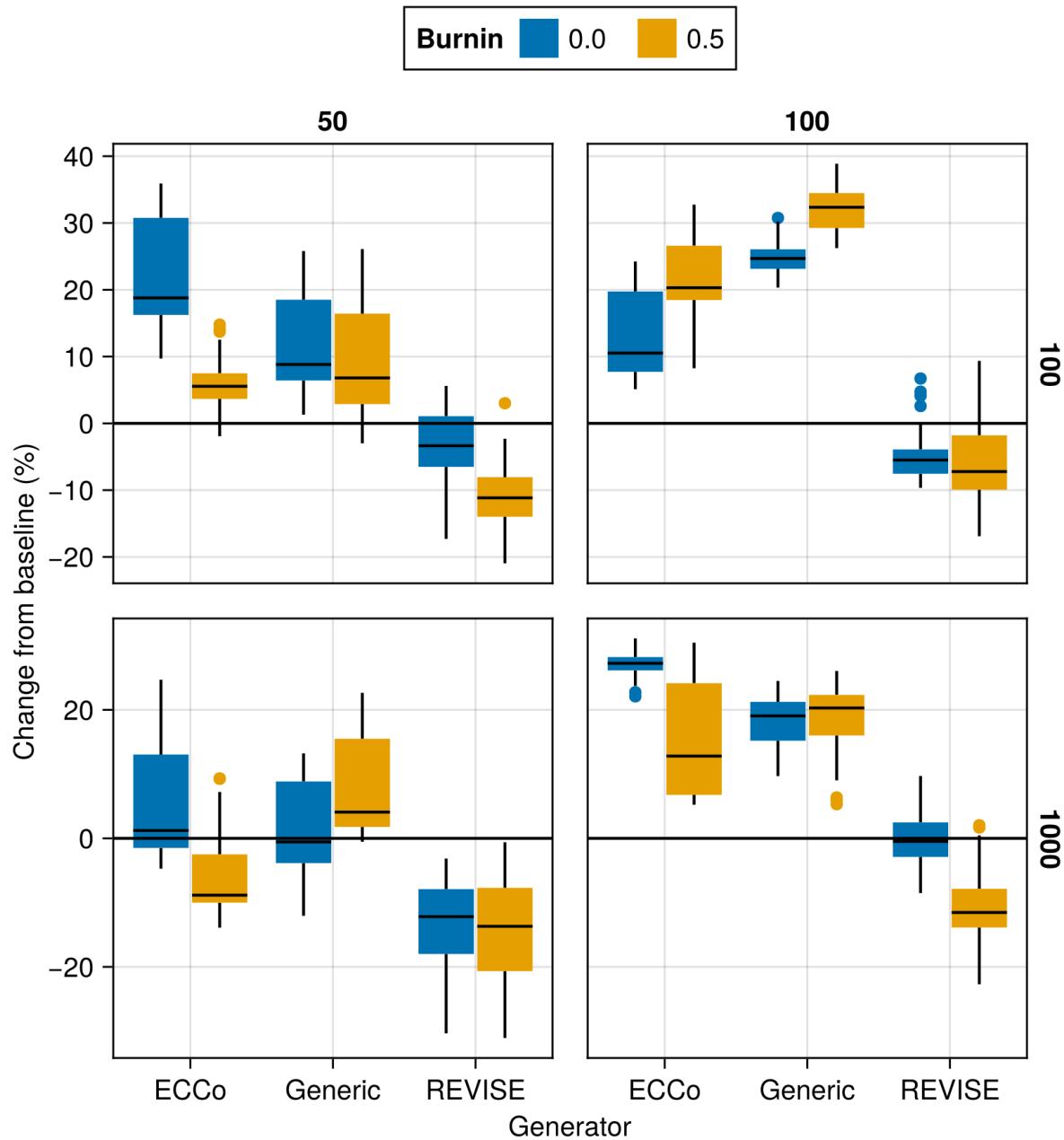


Figure A22: Average outcomes for the plausibility measure across hyperparameters. Data: Moons.

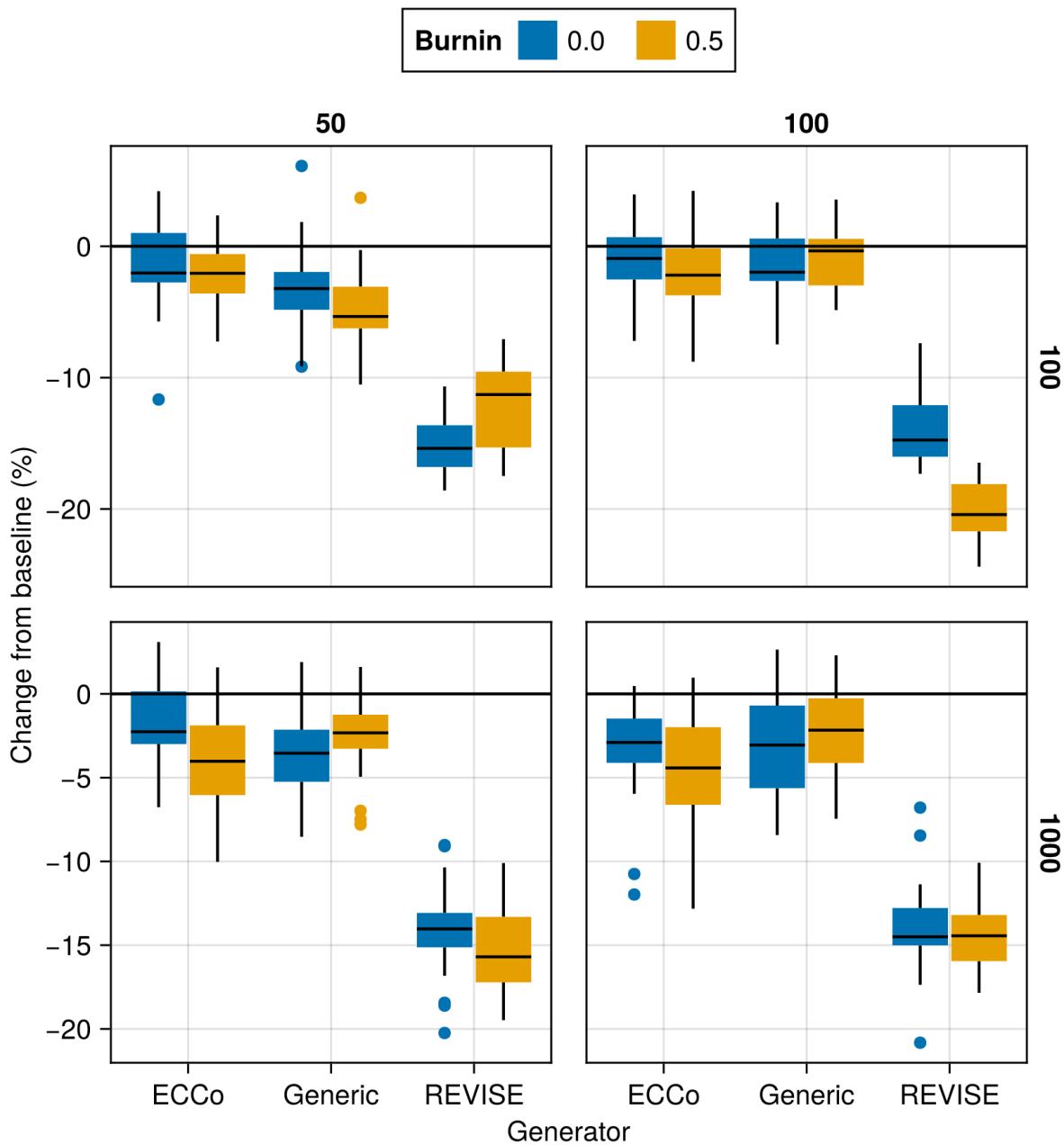


Figure A23: Average outcomes for the plausibility measure across hyperparameters. Data: Overlapping.

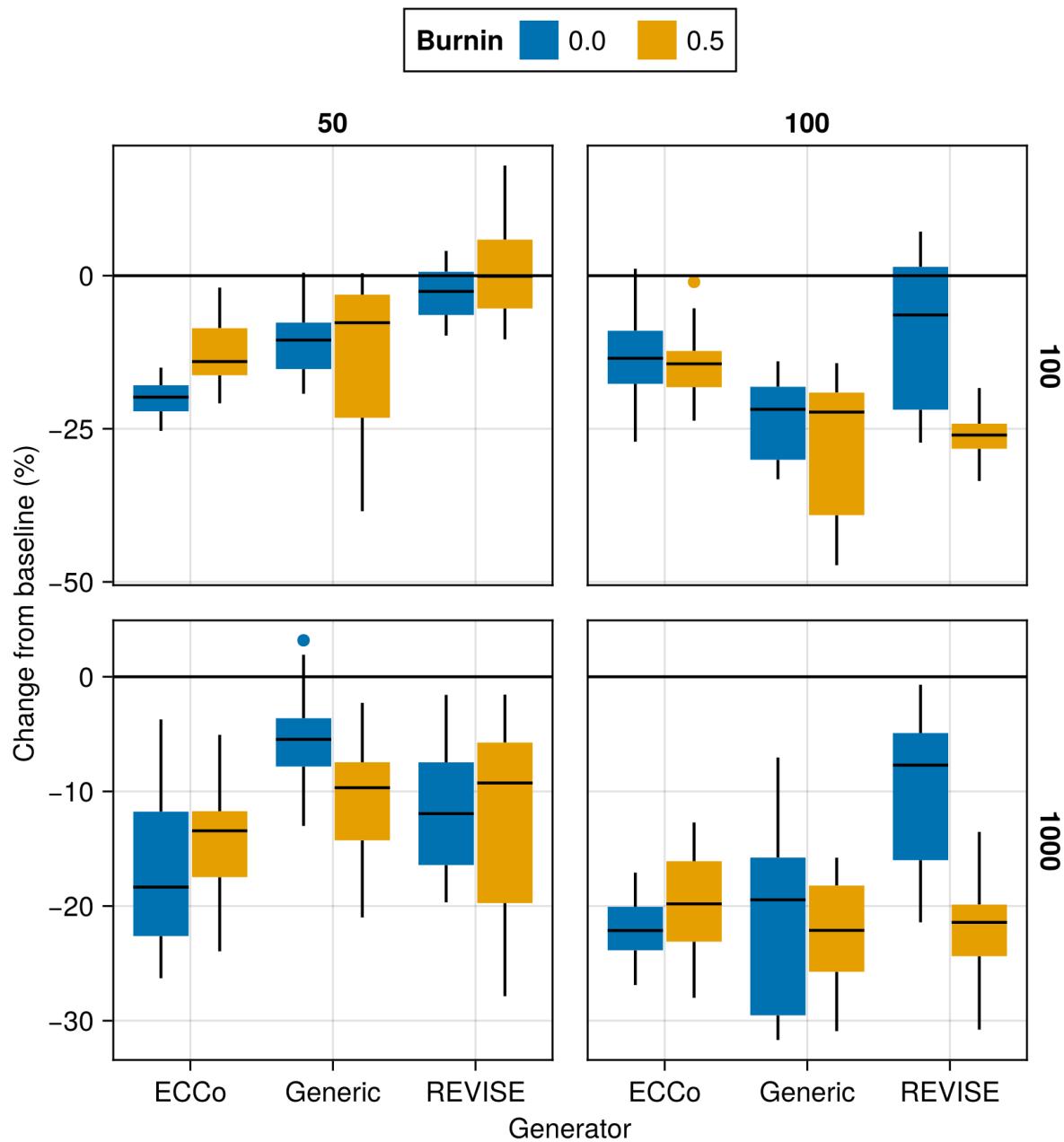


Figure A24: Average outcomes for the cost measure across hyperparameters. Data: Circles.

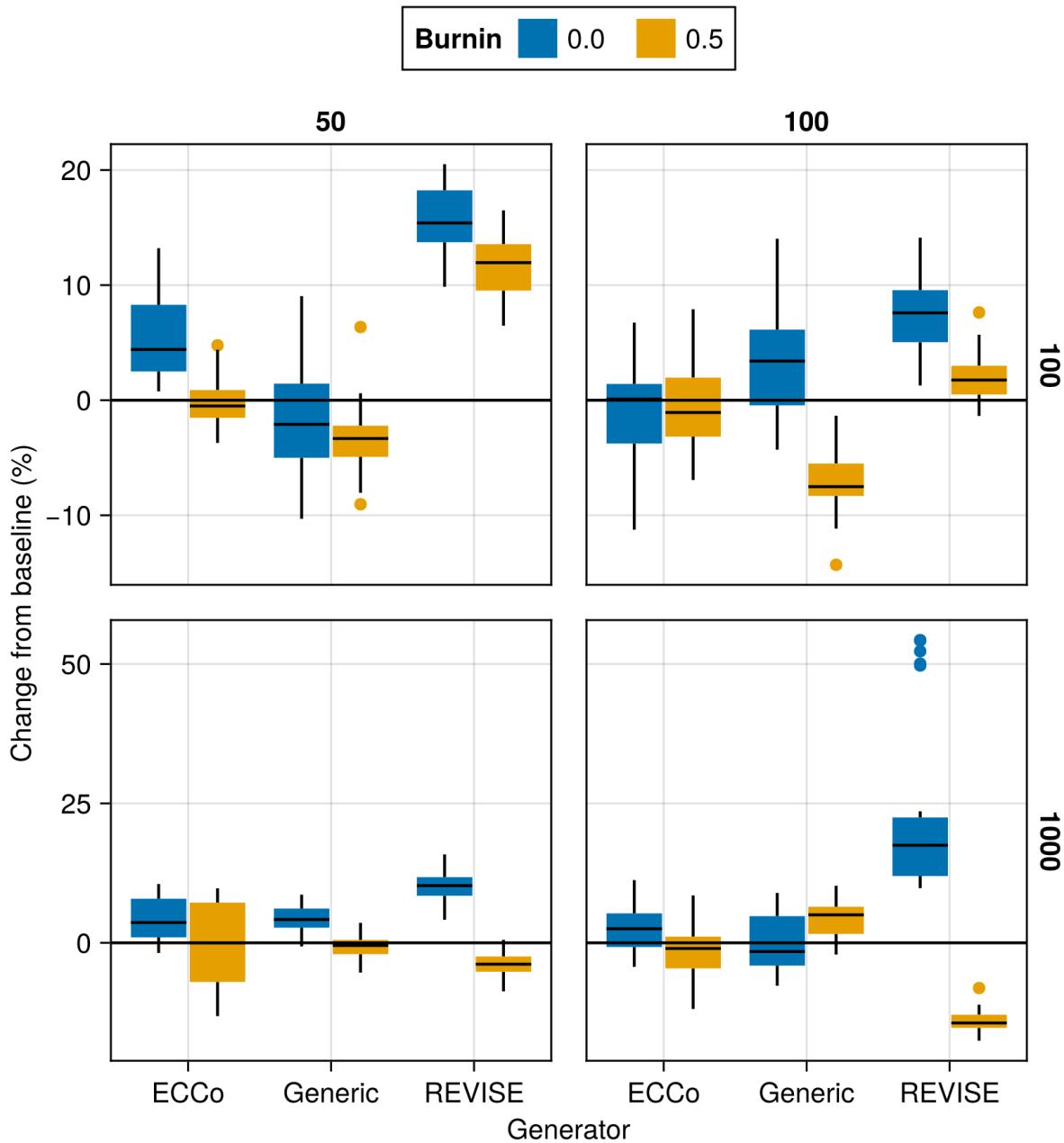


Figure A25: Average outcomes for the cost measure across hyperparameters. Data: Linearly Separable.

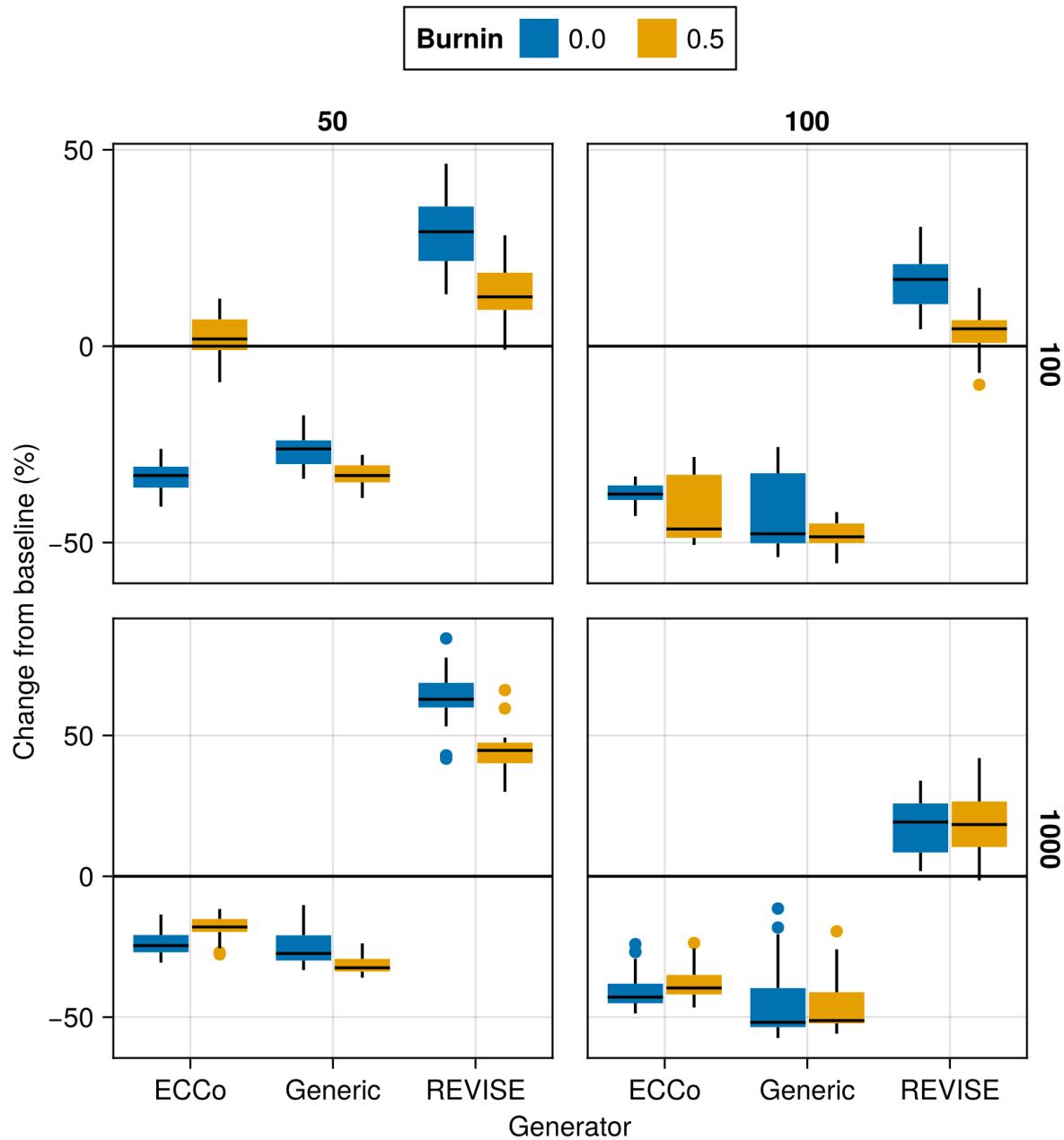


Figure A26: Average outcomes for the cost measure across hyperparameters. Data: Moons.

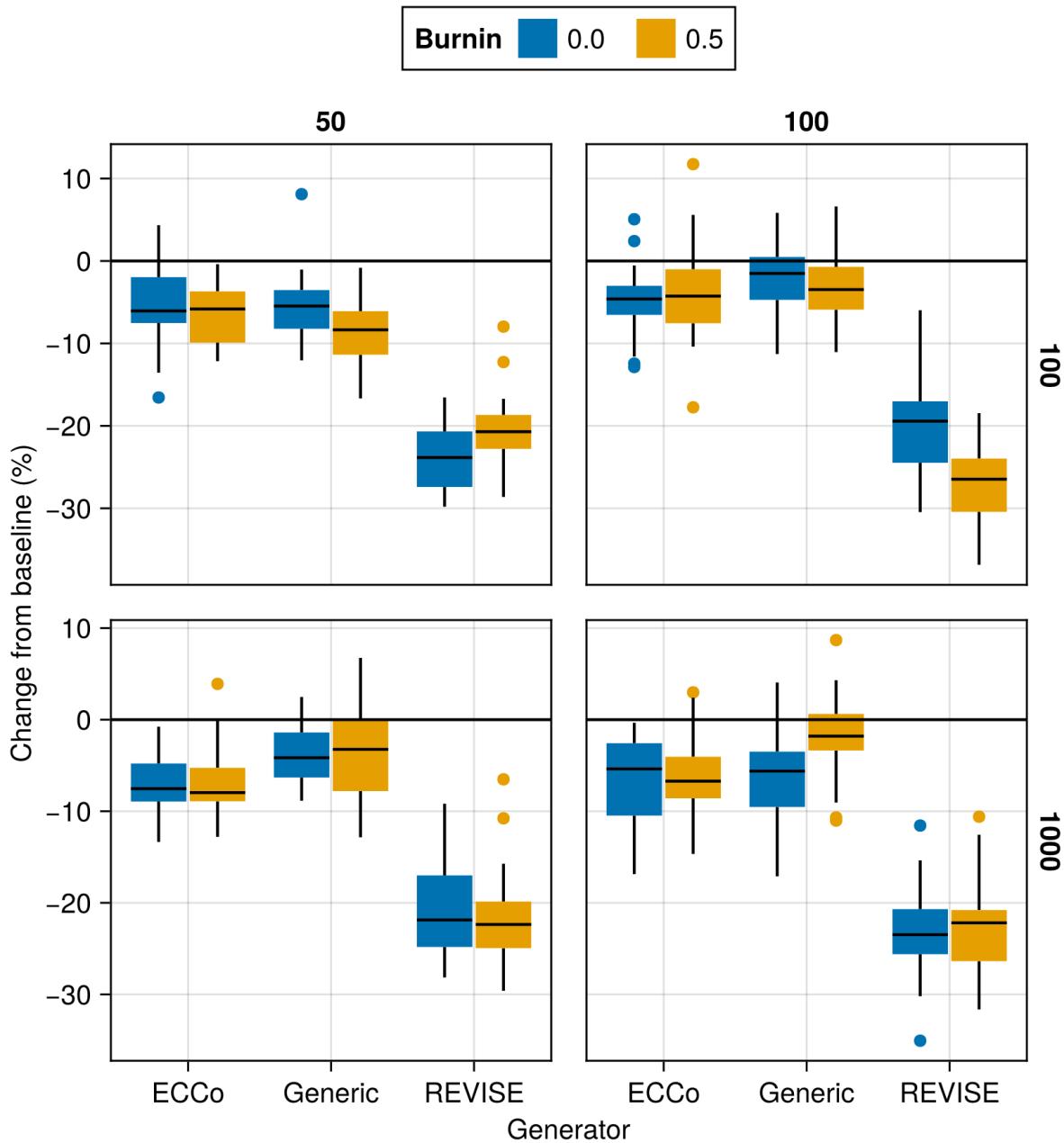


Figure A27: Average outcomes for the cost measure across hyperparameters. Data: Overlapping.