

# BayCon: Model-agnostic Bayesian Counterfactual Generator

Piotr Romashov<sup>1</sup>, Martin Gjoreski<sup>1</sup>, Kacper Sokol<sup>2</sup>, Vanina Martinez<sup>3</sup>, Marc Langheinrich<sup>1</sup>

<sup>1</sup>Università della Svizzera italiana, Switzerland

<sup>2</sup>RMIT University, Australia

<sup>3</sup>Universidad de Buenos Aires, Argentina

[martin.gjoreski@usi.ch@com](mailto:martin.gjoreski@usi.ch@com)

## Abstract

Generating counterfactuals to discover hypothetical predictive scenarios is the de facto standard for explaining machine learning models and their predictions. However, building a counterfactual explainer that is time-efficient, scalable, and model-agnostic, in addition to being compatible with continuous and categorical attributes, remains an open challenge. To complicate matters even more, ensuring that the contrastive instances are optimised for feature sparsity, remain close to the explained instance, and are not drawn from outside of the data manifold, is far from trivial. To address this gap we propose BayCon: a novel counterfactual generator based on probabilistic feature sampling and Bayesian optimisation. Such an approach can combine multiple objectives by employing a surrogate model to guide the counterfactual search. We demonstrate the advantages of our method through a collection of experiments based on six real-life datasets representing three regression tasks and three classification tasks.

## 1 Introduction

The “right to explanation” suggested by the General Data Protection Regulation (GDPR) [Goodman and Flaxman, 2017] challenged the Machine Learning (ML) community to build explainability into predictive models and their outputs. This paradigm shift – where predictive performance is no longer the only (and main) objective – gives rise to two distinct viewpoints. One argues that algorithmic black boxes should continue to be optimised for predictive power with explainability needs, possibly, fulfilled through post-hoc methods due to an apparent incompatibility of these two goals, thus forcing one of them to be sacrificed for the other.<sup>1</sup> The second standpoint disputes this trade-off as purely anecdotal and persuasively argues for building inherently transparent ML models, especially for high-stakes decisions [Rudin, 2019].

Counterfactuals are an explainability approach uniquely positioned in this space as they can be generated post-hoc but remain truthful with respect to the underlying black box (i.e., exhibit full fidelity). They enable ML users to understand what the output of a predictive model would have been had the instance in question changed in a particular way. This type of counterfactual analysis helps the explainees to simulate certain aspects of the ML model, thus improving its interpretability [Hoffman et al., 2018]. Notably, evidence from psychology and cognitive sciences suggests that people use counterfactual reasoning daily to analyse what could have happened had they acted differently [Byrne, 2005].

However, the number of counterfactuals that can be generated to explain any event (a selected datapoint) may be overwhelming [Byrne, 2019]. In addition to a large counterfactual search space, methods that are currently available tend to work for either classification or regression tasks, be restricted to a specific model family (e.g., differentiable predictors), struggle with large datasets (both in the number of instances and features), be computationally inefficient, or output out-of-distribution counterfactuals. We address these challenges with BayCon: a novel model-agnostic Bayesian counterfactual generator. To the best of our knowledge, it is the first counterfactual explainer based on Bayesian optimisation, making it fast to produce a sizeable number of high-quality contrastive instances. Our approach is model-agnostic and compatible with regression and classification tasks. It outperforms other state-of-the-art counterfactual generation methods on six real-life datasets, which demonstrates its effectiveness. Our evaluation uses three regression and three classification datasets with between 8 to 125 categorical and numerical attributes showing BayCon’s speed and versatility.

Existing methods for generating counterfactual explanations focus predominantly on differentiable models applied to continuous features [Wachter et al., 2018; Dhurandhar et al., 2018; Moore et al., 2019; Lash et al., 2017]. This creates a blind spot for non-differentiable models trained on datasets with mixed feature types, which are relatively ubiquitous [Rudin, 2019]. To address this gap, several authors proposed (Mixed) Integer Programming approaches [Cui et al., 2015; Russell, 2019; Kentaro et al., 2020]. Another counterfactual generation method, which is somewhat similar to BayCon, is

---

<sup>1</sup> <https://www.wired.com/story/googles-ai-guru-computers-think-more-like-brains/>

Multi-Objective Counterfactual Explanations (MOC) [Dandl et al., 2020]. MOC is model-agnostic, compatible with regression and classification tasks, and capable of processing numerical and categorical features. Given that both MOC and BayCon attempt to address the same set of counterfactual generation shortcomings, albeit with different approaches, we directly compare them in a set of experiments using six diverse evaluation metrics (see Tables 2 and 3). Additionally, we show how BayCon complies with recent guidelines for designing counterfactual generation methods, thus making it the preferred approach [Keane et al., 2021].

## 2. Preliminaries

Given an instance selected to be explained for a pre-trained ML model, BayCon generates similar instances that lead to the desired prediction, i.e., counterfactuals. A naïve approach is to generate all the possible feature–value combinations or to iteratively generate random instances, discarding the ones with unchanged prediction. However, for datasets with a considerable number of features this search space can be overwhelmingly large, rendering the naïve approaches impractical. A more appropriate strategy could use an informed search based on the record of previously generated and evaluated counterfactuals. These datapoints can be used to map the search space and the behaviour of the ML model. Based on this approximation, promising counterfactuals can be generated more efficiently. Bayesian optimisation can be a vehicle to realise such an informed search stochastically.

### 2.1. Counterfactual Explanations Desiderata

The BayCon optimisation pipeline is designed to produce contrastive explanations of the highest quality, both with respect to their technical and social properties. To this end, our method adheres to the latest guidelines prescribing how to generate desirable counterfactuals [Keane et al., 2021].

*What’s Plausible?* BayCon optimises for plausibility by minimising the distance to the explained instance in addition to automatically extracting feature constraints from the underlying training dataset. Moreover, our method allows the user to specify immutable features such as age, and indicate attribute values that are invalid, e.g., fractional number of rooms in a house. All these restrictions are used to guide quasi-random feature sampling (explained in Section 3.4).

*The Shape of Sparsity.* Counterfactuals should strive to tweak the smallest possible number of features to make the explanations parsimonious, hence appealing to humans [Keane et al., 2021]. However, the desired level of sparsity may depend on the user and the dataset, therefore we incorporate the number of altered feature values into the

optimisation function used by BayCon. Additionally, the user can specify the maximum number of altered features.

*Covering Coverage.* Counterfactuals should be feasible and actionable [Poyiadzi et al., 2020]. In particular, out-of-distribution counterfactuals – which can amount to 36% of all the generated explanations for some methods – should be avoided [Laugel et al., 2019]. BayCon uses Local Outlier Factor (LOF) to prevent such counterfactuals from being presented to the explainee.

*Comparative Testing.* BayCon is compared to state-of-the-art counterfactual explainers on six publicly available datasets using well-defined evaluation metrics.

### 2.2. Optimisation Objective

To assess the quality of generated counterfactual explanations, we designed a suitable objective function. It captures: (1) the distance in the feature space, (2) the distance in the output space, and (3) the number of altered features, all scaled to the  $[0, 1]$  range. Figure 1 shows example optimisation scores for the Bike dataset (cf. Table 1). Each point in the plot is a candidate counterfactual. The x-axis represents output of the ML model for which we are generating counterfactuals; the y-axis shows the Gower distance between each counterfactual and the explained instance; the z-axis captures the number of changed features; and the marker colour indicates the optimisation objective score given by Equation 1 (higher is better). In this example, the explained instance is predicted as 3141 (rented bikes) and the desired output range (provided by the explainee) is set to  $[4500, 5000]$ . The figure shows that: (i) the optimisation scores for counterfactuals whose predictions (y-axis) are outside of the user-specified range are close to 0 and increase as the model’s output approaches the desired range; (ii) the optimisation scores decrease as the Gower distance increases; and (iii) the optimisation scores are higher for counterfactuals that require a lower number of features to be changed.

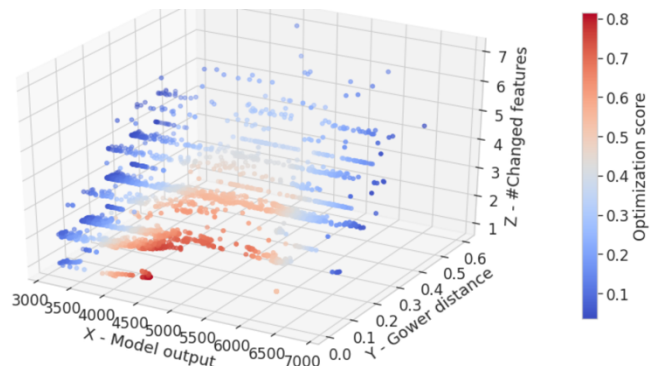


Figure 1. Example BayCon optimisation scores.

$$F(\bar{c}, \bar{x}) = S_x * S_y * S_f \quad (1)$$

*Similarity in the feature space ( $S_x$ ).* Gower distance is a distance metric used for mixed feature spaces. For *categorical* attributes it checks whether the two features have identical values – the distance component is 0 if the features are the same and 1 otherwise. For *numerical* features it calculates the absolute value of the difference between the attributes, divided by the numerical range of the feature. All of these individual components are then added up and divided by the number of attributes, which places the distance in the  $[0, 1]$  range. Next, we integrate this metric – the Gower distance between the explained instance  $\bar{x}$  and a counterfactual candidate  $\bar{c}$  – into our optimisation function ( $S_x$  in Equation 2) by subtracting it from 1:

$$S_x(\bar{c}, \bar{x}) = 1 - d_{Gower} \quad (2)$$

*Similarity in the output space ( $S_y$ ).* For classification tasks,  $S_y$  is 1 if the output of the ML model predicts the candidate counterfactual as requested by the user, and 0 otherwise. For regression problems, we define  $S_y$  as:

$$S_y = \begin{cases} 1, & \text{if } y_c \in [y_{\min}, y_{\max}] \\ 1 - \frac{|y_c - d|}{|y_x - d| + \theta}, & \text{otherwise} \end{cases}, \text{ where} \quad (3)$$

$$d = \begin{cases} y_{\min}, & \text{if } |y_c - y_{\min}| < |y_c - y_{\max}| \\ y_{\max}, & \text{otherwise} \end{cases}. \quad (4)$$

In Equation 3,  $y_x$  is the output of the ML model for the explained instance;  $y_c$  is the output of the ML model for the candidate counterfactual; and  $[y_{\min}, y_{\max}]$  is the target output range specified by the user. If  $y_c$  is in the desired range,  $S_y = 1$  (the maximum value). Otherwise,  $S_y$  captures the closeness of  $y_c$  to the borders (calculated via  $d$ ) of the desired range.  $S_y$  is designed to be within the  $[0, 1]$  interval.

*Proportion of tweaked features ( $S_f$ ).* This objective simply counts the number of features in the candidate counterfactual that are different when compared to the explained instance. This score is also in the  $[0, 1]$  range.

$$S_f(\bar{c}, \bar{x}) = \frac{\text{\# of different features between } \bar{c} \text{ and } \bar{x}}{\text{Overall \# of features}} \quad (5)$$

For comparison, MOC formalises counterfactual search as a multi-objective optimisation problem solved with

Nondominated Sorting Genetic Algorithm II (NSGA-II). The objectives used by MOC are: (i) prediction closeness to the desired goal, (ii) closeness to the initial instance in the feature space, (iii) number of changed features, and (iv) plausibility of counterfactual candidates based on the probability distribution over the feature values. BayCon mirrors objectives (i), (ii) and (iii) with the aforementioned scores:  $S_y$ ,  $S_x$ , and  $S_f$  respectively. Objective (iv) is addressed implicitly by the LOF filtering.

### 3 Methodology

Bayesian optimisation allows to utilise prior beliefs about a problem to help navigate the sampling. This is achieved by following a simplified version of the Bayes' theorem: posterior probability of a function  $F$  given data  $D$  (or evidence) is proportional to the likelihood of  $D$  (given  $F$ ) and the prior probability of  $F$ :

$$P(F|D) \propto P(D|F) P(F). \quad (5)$$

In our case,  $D$  consists of  $n$  observed counterfactuals and their black-box prediction:  $D = \{(\bar{c}_1, F(\bar{c}_1, \bar{x})), \dots, (\bar{c}_n, F(\bar{c}_n, \bar{x}))\}$ .

#### 3.1 Surrogate Model

To estimate the posterior of our objective function (Equation 1), we employ a surrogate model. It is an ML model typically learnt with regression algorithms based on a Gaussian Process (GP) because such models provide access to the full probability distribution [Snoek et al. 2015, Rasmussen et al., 2006]. By exploiting the mean and the standard deviation of the output distribution, one can balance the exploitation (higher mean) and exploration (higher standard deviation) trade-off. Since GPs are computationally expensive –  $O(n^3)$  complexity – ensemble regression models such as Random Forest can be used instead [Hutter et al., 2011]. In such a case, the mean and variance are calculated based on the predictions of all the individual models within the ensemble. In our case, the input of the surrogate model is defined as:

$$\text{input} = [\Delta k_1, \dots, \Delta k_n, \text{count}(\Delta k), d_{Gower}]. \quad (6)$$

In this equation  $\Delta k_i$  represents the distance between  $\bar{c}$  and  $\bar{x}$  for feature  $i$ ;  $\text{count}(\Delta k)$  is the number of features changed in  $\bar{c}$  as compared to  $\bar{x}$ ; and the last input is the Gower distance between  $\bar{c}$  and  $\bar{x}$ . Therefore, for any given input the surrogate model outputs an estimation of our optimisation score.

### 3.3 Acquisition Function

The mean  $\mu(\bar{S}_c)$  and variance  $\sigma(\bar{S}_c)$  calculated on the output of the surrogate model are used as input to an acquisition function, which is responsible for selecting the most promising counterfactuals. This function optimises the conditional probability of the feature space to identify regions with promising counterfactuals. BayCon uses *Expected Improvement* as its acquisition function [Moćkus, 1974], see also Equation 8 in Appendix A. In our experiments, the constant that controls the trade-off between global search and local optimisation (i.e., exploration/exploitation) is set as  $\xi = 0.01$  [Lizotte et al. 2008]. Intuitively, this acquisition function checks the improvement that each candidate counterfactual brings with respect to the maximum known value  $S_b$ , i.e.,  $\mu(\bar{S}_c) - S_b$ , and scales this improvement with respect to the uncertainty given by  $\sigma(\bar{S}_c)$ . If two counterfactuals have a similar mean value, the one with higher uncertainty is preferred by the acquisition function. A visual depiction of this process for a toy example is presented in Appendix A.

### 3.4 Generating Candidate Counterfactuals

*Initial Neighbourhood Generation.* Given the assumption that good counterfactuals should be close to the explained instance, our search is focused on its neighbourhood. To generate this space, for each feature we sample values at random with replacements from a truncated (based on the feature ranges) normal distribution centred around the initial instance. Categorical attributes are sampled uniformly over the set of possible values.

*Exploring Best Counterfactual Neighbourhoods.* Since good counterfactuals should come from dense regions, we explore neighbourhoods of explanations with best scores. We reuse the generation procedure applied to the initial instance, this time centred around the best counterfactuals.

*Random Feature Sampling.* To enable a higher degree of exploration, we sample values of numerical features uniformly at random from within their ranges. Categorical attributes are sampled uniformly over the set of possible values.

*Rounding.* To avoid indistinguishable counterfactuals that only differ beyond an  $n^{\text{th}}$  decimal place for numerical features, we perform  $k$ -bins discretisation with equal-width bins. We used  $k = 100$  for our experiments, which provides the minimum difference of 1% relative to attribute range.

*Selecting Features To Be Tweaked.* To increase sparsity, i.e., change the fewest possible features per counterfactual, we randomly select attributes to update based on a skewed distribution where the probability of changing  $n$  features is

double that of changing  $n+1$ . Only the selected features are then updated using the procedures described in the previous steps (neighbourhood generation or random sampling).

*Filtering.* BayCon is an iterative algorithm. At each step, we prune candidate counterfactuals whose score is below the current best. Also, prior to outputting the explanations we remove out-of-distribution explanations with LOF, which measures the local density deviation of each counterfactual with respect to its training set neighbourhood. Explanations that have a substantially lower density than their neighbours are therefore removed.

Algorithm 1 captures implementation details of BayCon.

## 4 Experiments

We compare BayCon against other counterfactual generation methods on six real-life datasets. Our method is implemented in Python 3.6 and relies heavily on scikit-learn [Pedregosa et al., 2011]. All the experiments were run on a 3.70GHz Intel Core i9 CPU with 128GB of RAM. We imposed a 15-minute runtime limit for each execution. BayCon implementation and the experimentation code, including processed datasets and analysis of the results, are freely available on GitHub [URL anonymised for review].

### 4.1 Experimental Setting

We compare our proposed method to a *brute-force exhaustive counterfactual search* implemented in FAT Forensics [Sokol et al. 2020] and MOC based on its official implementation. FAT Forensics only yielded explanations for the Diabetes dataset given the imposed 15-minute time limit, hence it is not featured in our comparison. MOC, on the other hand, generated explanations for all the datasets but the House Sales (likely due to the size of its training set) as it is a state-of-the-art method.

For the comparison we used three classification (Cls) and three regression (Reg) datasets (see Table 1). All the datasets are available online; the Bike dataset can be downloaded from the UCI repository<sup>3</sup> and the others are available through the OpenML repository [Vanschoren et al. 2014].

For each classification dataset we selected 10 random instances to be explained, generating their counterfactual explanations 3 times to account for randomness (i.e., 30 runs per dataset). For each regression datasets, we selected 3 initial instances, one for each percentile of the output variable: the median as well as the 25<sup>th</sup> ( $y_{25}$  in Equation 7) and the 75<sup>th</sup> percentiles. Next, we generated explanations for 4 desired target ranges – to increase, to decrease, to be in an interval

<sup>2</sup> Continue search if no counterfactuals found, and less than 100 epochs have passed, otherwise epochs are reduced to 10.

<sup>3</sup> <http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset>

above  $(y_x + a, y_x + b)$  and to be in an interval below  $(y_x - b, y_x - a)$  the prediction of the explained instance, with  $a$  and  $b$  defined in Equation 7. Each experiment was repeated 3 times (i.e., 36 runs per dataset).

$$a = 0.5 * y_{25}; b = 0.75 * y_{25} \quad (7)$$

We explained predictions of two black-box models: a Random Forest (RF) and a Support Vector Machine (SVM). The models were trained with all the data excluding the explained instances. Since SVMs can be sensitive to feature scaling and model parameterisation, we applied min-max normalisation to the input features and tuned the model parameters using 3-fold cross validation on the training data.

## 4.2 Experimental Results

Table 2 compares BayCon and MOC with respect to the total compute time, the time to first solution and the number of generated counterfactuals (all times given in seconds). The time to first solution for MOC was calculated as the total time

---

### Algorithm 1 BayCon.

---

**Input:** black-box-model  $f$ , instance to be explained  $x^*$ , desired prediction  $p$ , training data  $X_T$ .

**Output:** counterfactuals  $CFs$ .

```

1:  $X = \text{generate neighbourhood}(x^*)$ 
2:  $y = f(X)$  # predict neighbourhood
3:  $S_X = \text{objective function}(X, y, p)$  # calculate scores
4:  $X_K, y_K = \text{update known instances}(X, y)$ 
5:  $g = \text{RandomForest}(X_K, S_X)$  # train surrogate model
6:
7: while continue search2 do
8:    $CF = \text{select counterfactuals}(X_K, y_K)$ 
9:    $CF_b, S_b = \text{select best}(CF, S_X)$ 
10:   $X = \text{generate neighbourhood}(CF_b)$ 
11:   $X_P = \text{update promising instances}(X)$ 
12:   $X += \text{random generation}(S_b)$ 
13:   $\mu, \sigma = g(X)$ 
14:   $X_R = \text{acquisition function rank}(X, \mu, \sigma)$ 
15:   $y = f(X_R)$  # get black-box predictions
16:   $S_X += \text{objective function}(X_R, y, p)$ 
17:   $X_K = \text{update known instances}(X_R, y)$ 
18:   $g.\text{retrain}(X_K, S_X)$  # update surrogate model
19: end while
20:
21:  $y = f(X_P)$  # get black-box predictions
22:  $CFs += \text{update with counterfactuals from}(X_P, y)$ 
23:  $CFs = \text{LOF filter}(CFs, X_T)$ 
24: return  $CFs$ 
```

---

Dataset	Features (Num/Cat)	Type	Samples
Diabetes	8/0	Cls	768
Kc2	22/0	Cls	522
Biodeg		Cls	1055
Bike	7/3	Reg	730
House Sales	19/2	Reg	21613
Tecator	125/0	Reg	240

Table 1: Datasets used for experimental evaluation.

divided by the number of generated explanations. While such a strategy gives MOC an advantage, BayCon outperformed it across the board. Additionally, the House Sales dataset caused MOC to timeout, which is likely due to the size of the training dataset.

Table 3 outlines the experimental comparison between BayCon and MOC using the three evaluation scores –  $S_y$ ,  $S_f$ ,  $S_x$  – proposed in Section 2. It presents the mean and standard deviation for each score and the accompanying result of the Mann-Whitney U rank test. The sample size, which depends on the number of counterfactuals generated in each experimental run, is also shown – a sample size of 833 indicates that for this experiment we compared the scores of 833 explanations generated by MOC with the same number of explanations generated by BayCon. Since the methods could generate a different number of explanations, we only took the top  $n$  counterfactuals (ranked by the evaluation score) with  $n$  determined by the smallest number of explanations per method

	Method	Total t	t to 1 <sup>st</sup> CF	#CFs
Diabetes	BC	3.5(1)	0.1(0)	398(173)
	MOC	80.7(40)	1.6(1)	58(28)
Kc2	BC	9.0(5)	0.4(1)	2529(1437)
	MOC	192.1(133)	8.5(21)	45(19)
Biodeg	BC	13.2(9)	0.3(0)	1138(755)
	MOC	302.7(167)	4.5(8)	100(49)
Bike	BC	4.7(2)	0.0(0)	1446(493)
	MOC	47.6(27)	1.1(2)	78(56)
H. Sales	BC	26.2(4)	0.2(0)	1723(674)
	MOC	/	/	/
Tecator	BC	83.3(34)	0.1(1)	42949(20154)
	MOC	429.1(98)	155.5(52)	3(1)
Average	BC	23.3(9.2)	.18(.33)	8363(3947)
	MOC	210.4(93)	3.16(17)	47(25)

Table 2: Experiment runtimes and numbers of generated counterfactuals given as: mean (standard deviation).

generated for a given experimental setup. Bolded p-values highlight the experiments in which BayCon outperformed MOC with statistical significance ( $p < 0.05$ ). This happened in most of the experiments, except for the Bike and the Tecator datasets predicted with an SVM, but only when measured by the  $S_y$  score. In these specific experiments, MOC found better counterfactuals in the output space ( $S_y$ ), but BayCon found better counterfactuals in the feature space ( $S_f$  and  $S_x$ ). Across all experiments, BayCon offered counterfactuals with a smaller number of changed features and smaller Gower distance, while achieving the target output.

## Conclusions and Future Work

Our experiments demonstrated that, compared to state-of-the-art methods, BayCon is more time-efficient, and generates larger and more diverse sets of counterfactuals (see Table 2). Furthermore, the explanations output by our algorithm are of better quality: they are placed closer to the explained instance and require fewer feature tweaks, thus making them more similar to it. In future work we plan to address the

counterfactual multiplicity by exploring various filtering, pruning and selection methods. We will also investigate visualisation techniques to help the users navigate the output explanations and select them based on (possibly implicit) user preferences. Moreover, we will conduct user studies to analyse the perceived quality and benefit of our counterfactuals to avoid “neglecting the users” [Keane et al., 2021].

## References

- [Brochu et al. 2008] Brochu, Eric, Vlad M. Cora, and Nando De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599, 2010.
- [Byrne, 2019] Byrne, Ruth MJ. Counterfactuals in Explainable Artificial Intelligence (XAI): Evidence from Human Reasoning. In IJCAI, pp. 6276-6282. 2019.
- [Byrne, 2005] Ruth M.J. Byrne. The Rational Imagination: How People Create Alternatives to Reality”. MIT Press, Cambridge, Massachusetts, 2005.

Bike												Kc2											
RF (sample size = 833)						SVM (sample size = 767)						RF (sample size = 528)						SVM (sample size = 512)					
$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$						
BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC						
$\mu$ 1.0	1.0	.90	.69	.98	.69	.72	1.0	.85	.69	.93	.65	1.0	1.0	.90	.87	.97	.96						
$\sigma$ 0.0	0.0	.01	.07	.03	.49	.23	0	.07	.06	.08	.57	0.0	0.0	.06	.09	.06	.09						
$p>.05$		$p<0.001$		$p<0.001$		$p<0.001$		$p<0.001$		$p<0.001$		$p>.05$		$p=.046$		$p<0.001$							

Diabetes												Tecator											
RF (sample size = 1,002)						SVM (sample size = 1,137)						RF (sample size = 6)						SVM (sample size = 20)					
$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$						
BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC						
$\mu$ 1.0	1.0	.84	.70	.94	.89	1.0	1.0	.83	.71	.97	.90	.95	1.0	.99	.94	.99	.93						
$\sigma$ 0.0	0.0	.06	.18	.03	.10	0.0	0.0	.08	.14	.02	.08	.05	0.0	.01	.02	.11	.01						
$p>.05$		$p<0.001$		$p<0.001$		$p>.05$		$p<0.001$		$p<0.001$		*Sample size too small for a statistical test						$p<0.001$		$p<0.001$		$p<0.001$	

Biodeg												House Sales											
RF (sample size = 1,437)						SVM (sample size = 1,857)						RF (sample size = 31,036)						SVM (sample size = 45,797)					
$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$	$S_y$	$S_f$	$S_x$						
BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC	BC	MOC						
$\mu$ 1.0	1.0	.95	.94	1.0	1.0	1.0	1.0	.98	.95	1.0	.99	.46	-	.96	-	.86	-						
$\sigma$ 0.0	0.0	.02	.03	.004	.007	1.0	1.0	.01	.03	.01	.01	.28	-	.04	-	.09	-						
$p>.05$		$p<0.001$		$p<0.001$		$p>.05$		$p<0.001$		$p<0.001$		*MOC did not finish within imposed 15-minute time limit											

Table 3: Comparison of evaluation scores for BayCon (BC) and MOC.

- [Cuieal.,2015] Zhicheng Cui, Wenlin Chen, Yujie He, and Yixin Chen. Optimal action extraction for random forests and boosted trees. In Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 179–188, 2015.
- [Dandl et al., 2020] Dandl, Susanne, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In International Conference on Parallel Problem Solving from Nature, pp. 448-469. Springer, Cham, 2020.
- [Dhurandharetal et al., 2018] Amit Dhurandhar, Pin Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent. In NIPS, pp. 590-601, 2018.
- [Goodman and Flaxman, 2017] B. Goodman, S. Flaxman, (2017). European Union regulations on algorithmic decision-making and a “right to explanation”, AI Magazine 38 (3) 50–57.
- [Hoffman et al., 2018] Hoffman, Robert, Tim Miller, Shane T. Mueller, Gary Klein, and William J. Clancey. Explaining explanation, part 4: a deep dive on deep nets. IEEE Intelligent Systems 33, no. 3 (2018): 87-95.
- [Hutter et al., 2011] Hutter, F., Hoos, H.H., Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. In International Conference on Learning and Intelligent Optimization, Springer, 2011.
- [Keane et al., 2021] Keane, Mark T., Eoin M. Kenny, Eoin Delaney, and Barry Smyth. If only we had better counterfactual explanations: Five key deficits to rectify in the evaluation of counterfactual XAI techniques. In IJCAI, pp. 4466-4474. 2021
- [Kentaro et al., 2020] Kanamori, Kentaro, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization. In IJCAI, pp. 2855-2862. 2020.
- [Lash et al., 2017] Michael T. Lash, Qihang Lin, W. Nick Street, Jennifer G. Robinson, and Jeffrey W. Ohlmann. Generalized inverse classification. In Proceedings of the SIAM International Conference on Data Mining, pp. 162–170, 2017.
- [Laugel et al., 2019] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard and Marcin Detyniecki. The dangers of post-hoc interpretability. In IJCAI-19, pp. 2801-2807, 2019.
- [Lizotte et al. 2008] Lizotte, D. J. Practical Bayesian Optimization. PhD Thesis, University of Alberta, Department of Computing Science, 2008.
- [Moćkus, 1974] Moćkus, J. On Bayesian methods for seeking the extremum. IFIP Technical Conference on Optimization Techniques (pp. 400-404). Berlin: Springer, 1974.
- [Moore et al., 2019] Jonathan Moore, Nils Hammerla, and Chris Watkins. Explaining deep learning models with constrained adversarial examples. In Proceedings of the 16<sup>th</sup> Pacific Rim International Conference on Artificial Intelligence, pp. 43–56, 2019.
- [Pedregosa et al., 2011] Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J. Scikit-learn: Machine learning in Python. The Journal of Machine Learning research. 12:2825-30, 2011.
- [Poyiadzi et al., 2020] Poyiadzi R., Sokol K., Santos-Rodriguez R., De Bie T., Flach P. FACE: Feasible and actionable counterfactual explanations. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society pp. 344–350, 2020.
- [Rasmussen et al., 2006] Rasmussen, C., and Williams, C. K. Gaussian Processes for Machine Learning. Cambridge, Massachusetts: The MIT Press, 2006.
- [Rudin, 2019] Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence 1, no. 5:206-215.
- [Russell, 2019] Chris Russell. Efficient search for diverse coherent explanations. In Proceedings of the Conference on Fairness, Accountability, and Transparency, 2019.
- [Snoek et al. 2015] Snoek, Jasper, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In International conference on machine learning, pp. 2171-2180. PMLR, 2015.
- [Sokol et al. 2020] Sokol, Kacper, Alexander Hepburn, Rafael Poyiadzi, Matthew Clifford, Raul Santos-Rodriguez, and Peter Flach. FAT Forensics: A Python Toolbox for Implementing and Deploying Fairness, Accountability and Transparency Algorithms in Predictive Systems. Journal of Open Source Software 5, no. 49 (2020): 1904.
- [Vanschoren et al. 2014] Joaquin Vanschoren, Jan N. Van Rijn, Bernd Bischl, and Luis Torgo. OpenML: networked science in machine learning. ACM SIGKDD Explorations Newsletter 15, no. 2 (2014): 49-60.
- [Wachter et al., 2018] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. Harvard journal of law & technology, 31:841–887, 2018.

## Appendices

### A. Bayesian Optimisation Example

*Expected Improvement* acquisition function is defined as [Moćkus, 1974]:

$$EI(\bar{S}_c, S_b) = \begin{cases} (\mu(\bar{S}_c) - S_b - \xi)\Phi(Z) + \sigma(\bar{S}_c)\phi(Z), & \text{if } \sigma(SM(\bar{x})) > 0 \\ 0, & \text{if } \sigma(\bar{S}_c) = 0 \end{cases} \quad (8)$$

$$Z = \begin{cases} (\mu(\bar{S}_c) - S_b - \xi)/\sigma(\bar{S}_c) > 0 \\ 0, & \text{if } \sigma(\bar{S}_c) = 0 \end{cases}$$

$\bar{S}_c$  is the output of the surrogate model for the counterfactual  $\bar{c}$ ;  $S_b$  is the maximum known value from candidate counterfactuals seen thus far;  $\Phi(\cdot)$  is the normal cumulative distribution function; and  $\xi$  is a constant that controls the trade-off between global search and local optimisation (i.e., exploration/exploitation).

Figure 2 presents a one-dimensional – a single attribute given along the x-axis – toy example of Bayesian optimisation. The plot shows an approximation of the objective function using a surrogate model over four sampling iterations of the objective function. The figure also displays the acquisition function, which is captured by the green plot. This

function has a high value where the surrogate model either predicts a high-valued objective (exploitation) or where the prediction uncertainty is high (exploration), therefore regions of high objective value and high uncertainty are explored first. Note that the far-left area of the plot remains unsampled; while it has high uncertainty, it is also (correctly) predicted to offer little improvement over the highest-value observation.

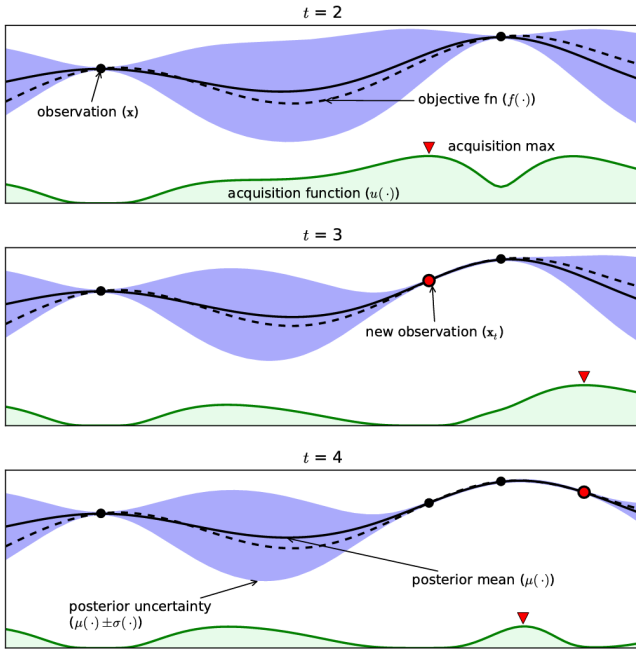


Figure 2. Toy example of Bayesian optimisation taken from [Lizotte et al. 2008].