FACET: Robust Counterfactual Explanation Analytics

PETER M. VANNOSTRAND, Worcester Polytechnic Institute, USA
HUAYI ZHANG*, ByteDance, USA and Worcester Polytechnic Institute, USA
DENNIS M. HOFMANN, Worcester Polytechnic Institute, USA
ELKE A. RUNDENSTEINER, Worcester Polytechnic Institute, USA

Machine learning systems are deployed in domains such as hiring and healthcare, where undesired classifications can have serious ramifications for the user. Thus, there is a rising demand for explainable AI systems which provide actionable steps for lay users to obtain their desired outcome. To meet this need, we propose FACET, the first explanation analytics system which supports a user in *interactively* refining counterfactual explanations for decisions made by tree ensembles. As FACET's foundation, we design a novel type of counterfactual explanation called the *counterfactual region*. Unlike traditional counterfactuals, FACET's regions concisely describe portions of the feature space where the desired outcome is guaranteed, regardless of variations in exact feature values. This property, which we coin *explanation robustness*, is critical for the practical application of counterfactuals. We develop a rich set of novel explanation analytics queries which empower users to identify personalized counterfactual regions that account for their real-world circumstances. To process these queries, we develop a compact high-dimensional counterfactual region index along with index-aware query processing strategies for near real-time explanation analytics. We evaluate FACET against state-of-the-art explanation techniques on eight public benchmark datasets and demonstrate that FACET generates actionable explanations of similar quality in an order of magnitude less time while providing critical robustness guarantees. Finally, we conduct a preliminary user study which suggests that FACET's regions lead to higher user understanding than traditional counterfactuals.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**.

Additional Key Words and Phrases: Explainable AI, Counterfactual Explanation, Random Forest, Gradient Boosting Ensembles, Interpretable Machine Learning.

ACM Reference Format:
Peter M. VanNostrand, Huayi Zhang, Dennis M. Hofmann, and Elke A. Rundensteiner. 2023. FACET: Robust Counterfactual Explanation Analytics. *Proc. ACM Manag. Data* 1, 4 (SIGMOD), Article 242 (December 2023), 27 pages. https://doi.org/10.1145/3626729

1 INTRODUCTION

Machine learning systems are increasingly deployed to support decision-making tasks that profoundly affect people's lives including in high-stakes domains such as hiring, healthcare, and finance. As such, eXplainable AI (XAI) techniques are essential to examine the reasoning behind the

*Research conducted while a PhD Candidate at Worcester Polytechnic Institute

Authors' addresses: Peter M. VanNostrand, pvannostrand@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA; Huayi Zhang, hzhang4@wpi.edu, ByteDance, 1199 Coleman Ave, San Jose, California, USA and Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA; Dennis M. Hofmann, dmhofmann@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA; Elke A. Rundensteiner, rundenst@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA.

2836-6573/2023/12-ART242 $15.00
https://doi.org/10.1145/3626729

decision-making of these systems [2, 34]. Indeed, requirements for lay user appropriate explanations have even been codified into law [1, 38]. To meet these needs, *counterfactual explanations* [46] have emerged as a promising strategy. These explanations answer questions of the form "why A rather than B?" such as, "Why was my loan application rejected rather than accepted?"

Given an instance x predicted as class A, a counterfactual explanation alters the feature values of x to produce a hypothetical *counterfactual example* x' which would be predicted as class B [46]. Such explanations satisfy regulatory requirements [47] and have been shown to be easily understood by lay users [27, 34, 35]. To ensure the resulting example is relevant to x, state-of-art techniques aim to minimize the distance between x and x' using a distance metric such as L1 or L2 norm [9, 12, 16, 37]. However, in practice simply minimizing explanation distance often leads to unsatisfying counterfactual examples that fail to account for real-world factors such as the difficulty or impossibility of precisely controlling some features [3].

In this work, we consider practical counterfactual explanations for widely deployed AI models, namely, tree ensemble models such as Random Forests, Gradient Boosted Decision Trees, and Extra-Trees. These models are popular due to their high accuracy and minimal need for hyperparameter tuning [8], yet they unfortunately lack a standard for explanation capabilities suitable to lay users.

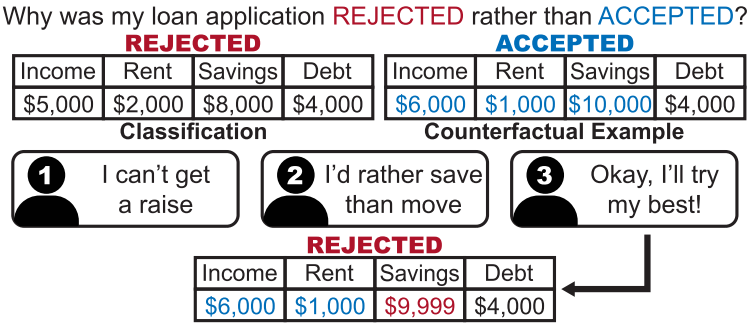

Fig. 1. Motivating loan application scenario.

Motivating Example. Here, we illustrate shortcomings of state-of-art approaches using a loan application example. As shown in Fig. 1, a user who receives an undesired loan rejection may seek a counterfactual explanation to determine what changes they should make to get their loan accepted. State-of-art XAI techniques typically generate a *single (static) counterfactual example* as shown above. While sufficient for some users, this explanation would not be *actionable* for a user who cannot alter the required features (User 1). Some users may also have strong preferences for changing one feature versus another based on their *personal* real-world circumstances (User 2). Further, even when a user finds an explanation sufficient, their real-world alterations may not exactly match the example (User 3). In this case, the user may expend a great deal of effort making alterations yet still be denied a much-needed loan. *These shortcomings combine to make static counterfactual examples useless for the practical explanation of high-stakes decisions.*

Challenges. Based on the motivating example above, we derive the following three key requirements for effective counterfactual explanation workflows.

- ***Explanation Personalization.*** As shown above, users' unique real-world considerations render certain counterfactual explanations unactionable. Rather than rigidly generating one arbitrary counterfactual, XAI systems must generate *personalized* explanations specific to the user's situation, constraints, and priorities. This is difficult as real users cannot painstakingly enumerate every combination of explanation criteria they find acceptable. Instead, practical systems must quickly generate initial explanations, and subsequently support efficient *interactive explanation refinement* in collaboration with the user to swiftly hone in on a satisfying personalized explanation. We coin this process *explanation exploration.*

- ***Explanation Robustness***. As shown above, a user who alters their instance to nearly (but not exactly) match a counterfactual example may not receive their desired outcome. We coin the term *explanation robustness* to refer to how tightly the user must match the prescribed changes, with more robust explanations tolerating larger deviations from the proposed feature values. Robustness is especially critical when features have large expected variations (e.g., savings). If an explanation is not robust to these fluctuations, the outcome may essentially be left to random chance. To overcome this, effective explanation systems must generate robust explanations.
- ***Explanation Efficiency.*** To generate actionable explanations, XAI systems must work interactively with humans-in-the-loop. Unfortunately, users have a tendency to become discouraged and cease interaction as response times increase [30]. Thus, XAI systems must work in near real-time to prevent users from giving up and falsely concluding that no suitable explanation exists for them. As classification is a function of all feature values, the number of possible counterfactual points is extremely large. Further, as a model's decision surface grows in complexity, it is increasingly difficult to determine which portions of the feature space produce the desired outcome. These factors make real-time explanation challenging, with optimal counterfactual explanation of tree ensembles proven to be NP-Hard [9].

State-of-the-Art and its Limitations. Existing XAI techniques have fundamental limitations which prevent them from meeting the above requirements. Model-opaque[1] techniques perform counterfactual explanation for arbitrary model architectures. MACE [12], a technique which generates counterfactual examples using local predicted class probabilities, quickly reaches intractable runtime due to repeated predictions (Sec. 7.2) failing the *explanation efficiency* requirement. Meanwhile, LORE [18] and LEWIS [17] are unable to guarantee the validity of explanations they generate, a critical prerequisite for *explanation robustness*.

Model-transparent techniques exist for tree ensemble explanation. AFT [45] generates counterfactual examples through exhaustive generation of altered instances, yet frequently fails to find a valid explanation (Sec. 7.2). RFOCSE [16] generates counterfactuals by fusing the ensemble into a single tree, but this is exponentially complex and runtime remains a significant issue (Sec. 7.2). None of these methods allow users to interact with the XAI system or input constraints, and thus fail to generate *personalized explanations.*

Recently, OAE [9] and OCEAN [37] transform ensemble explanation into an integer programming (IP) task, optimizing explanation distance. The latter includes user considerations as static IP constraints, but does not include humans-in-the-loop as required for *personalization*. Moreover, as the number of IP parameters grows with ensemble complexity, the runtime of this approach is exponential in the ensemble size (Sec. 7.7). Thus, these methods don't fulfill the requirements of *explanation efficiency* nor *interactivity*.

Proposed Solution. We propose FACET (Fast Actionable Counterfactuals for Ensembles of Trees), the first interactive system for generating *robust personalized counterfactual explanations*. To ensure explanation robustness, FACET introduces a novel type of counterfactual explanation, called the *counterfactual region*, Counterfactual regions generalize counterfactual examples to n-dimensional volumes which include a potentially infinite number of examples. Thus they provide a *minimum robustness guarantee* for all examples within. FACET's region explanations can be created for a wide variety of models, with demonstrated implementations for decision trees, random forests, and gradient-boosted tree ensembles (Sec. 4).

FACET, the first to provide *explanation personalization*, introduces rich explanation analytics. These analytics empower users to interactively identify counterfactual regions that are guaranteed to be both actionable and relevant to their real-world circumstances. During preprocessing, FACET

[1]https://www.acm.org/diversity-inclusion/words-matter

produces a large set of promising counterfactual regions apt as explanation candidates. To manage these regions, we design COREX, a custom counterfactual explanation index. COREX supports compact spatial encoding of the high-dimensional counterfactual regions and accelerates query execution. During explanation exploration, FACET provides index-aware processing strategies to realize each analytic operator. This enables FACET to efficiently service the user with relevant region explanations in near real-time, even when the explained model is complex and queries involve multiple interacting constraints.

Contributions. Our key contributions include:

(1) Propose *counterfactual regions*, a novel abstraction which extends the idea of counterfactual examples to provide a much needed guarantee of explanation robustness.
(2) Characterize counterfactual regions for ensembles of trees and develop a data-driven approach for generating a set of high interest regions which accurately reflect the ensemble's function.
(3) Design COREX, a compact custom bit vector index which manages a large number of high-dimensional counterfactual regions for efficient near real-time explanation querying.
(4) Develop a rich set of analytic operators which support users to interactively identify the best counterfactual explanation for them subject to actionability and robustness requirements.
(5) Evaluate FACET on eight public benchmark datasets. We demonstrate that FACET can explain complex ensembles an order of magnitude faster than the state-of-the-art while achieving similar explanation quality and substantially higher robustness.
(6) Perform a between-subjects pilot user study comparing FACET's regions to counterfactual examples and show that our regions may lead to better user understanding.

2 FACET EXPLANATION PARADIGM

Here we introduce FACET's fundamentally new type of explanation: *counterfactual regions*. We also overview FACET's query language for identifying personalized counterfactual region explanations.

2.1 Problem Setup

Given a classifier model $f : \mathcal{X} \rightarrow \mathcal{Y}$ which maps an instance x (e.g., a loan application) from the feature space $\mathcal{X} \in \mathbb{R}^n$ to a given class c in the set of classes C (e.g., $C = \{reject, accept\}$) [26, 47]. Let $y = f(x)$ be the observed factual class c_o for x (e.g., loan reject), and c_d the user's desired counterfactual class (e.g., loan accept), $c_d \neq c_o$.

Definition 1. (Counterfactual Example x'). *Given an instance x with $f(x) = c_o$, a counterfactual example x' corresponds to an altered version of x such that $f(x') = c_d$. To ensure that x' is relevant to x (i.e., requires only a few or small alterations), x' is chosen to minimize its distance to x according to a function δ such that:*

$$x' = \arg\min_{x'_i} \{\delta(x, x'_i) \mid f(x) = c_o \wedge f(x'_i) = c_d\}$$

Thus, a counterfactual example can be conceptualized as a recommendation for the smallest set of changes that a user should make to transform their instance x into x' and achieve the desired outcome. In principle, many design options are feasible for δ [26, 34]. Here we work with the popular and commonly adopted L1 (Manhattan) and L2 (Euclidean) norms [12, 16, 37, 45].

Remark 1. (Limitation of Counterfactual Examples). *By Def. 1, a counterfactual example x' will be of the desired class $f(x') = c_d$. However, if a user fails to make changes in the real world to exactly transform x into x', then the behavior of the model is not guaranteed. Indeed, deviating even a small amount from x' to some other point x'' could result in an unknown outcome $f(x'')$ which may not be counterfactual to x. Thus static counterfactual examples are unfortunately not* robust *to real-world feature value variations.*

Definition 2. (Counterfactual Robustness $\psi(x')$). *Given x with $f(x) = c_o$, and x' with $f(x') = c_d$. Let $v \in \mathbb{R}^n$ represent a variation (i.e., a small change in feature values) such that $x'' = x' + v$. The robustness of the example $\psi(x')$ is defined as the scale of the minimal variation v such that x'' is no longer counterfactual to x.*

$$\psi(x') = \left\| \arg\min_{v} \{ \delta(x', x'') \mid f(x'') = c_o,\ x'' = x' + v \} \right\|$$

Remark 2. (Robustness Relevance Trade-Off). *Comparing Def. 2 to Def. 1, we see that the goal of having relevant counterfactual examples is in tension with having robust counterfactuals. Indeed, choosing x' to minimize $\delta(x, x')$ (increasing relevancy) also unwantedly minimizes $\psi(x')$ resulting in a decreased robustness of the resulting explanation.*

Objectives. Considering Rem. 2, we aim to design a counterfactual explanation system that supports users in the following ways

(1) Understand what variations may result in non-counterfactual outcomes as some variations are more likely than others (e.g., a user may expect more variations in savings than in rent.)
(2) Empower them to trade-off between explanation relevance (requiring a larger more difficult change) and explanation robustness (allowing for more variation in the exact final value).

2.2 Abstraction of a Counterfactual Region

To achieve the objectives above (Sec. 2.1), we now design a novel type of counterfactual explanation called the *counterfactual region.*

Definition 3. (Counterfactual Region R). *Given a counterfactual example x' for an instance x, let $\alpha \in \mathbb{R}$ be a relaxation factor. Let R be the set of all points x'_i within distance α from x'. Then, R is called counterfactual if and only if all points within R are guaranteed to be counterfactual to x in the desired class c_d (w.r.t. model f).*

$$R = \left\{ x'_i \in \mathcal{X} \mid \delta(x', x'_i) \leq \alpha \right\} \ s.t.\ \forall x'_i \in R, f(x'_i) = c_d$$

Forming a counterfactual region explanation for x per the above definition provides the three following beneficial properties.

Property 1. (Class Homogeneity). *Given a counterfactual region R as in Def. 3, all points $x'_i \in R$ are guaranteed to be members of the desired counterfactual class $f(x'_i) = c_d$.*

Per Prop. 1, a counterfactual region compactly represents a large number of counterfactual examples. This provides the user with much needed flexibility in choosing their exact targeted feature values. E.g., if x' requires *income* = \$4732.12 and α = \$100, the user could target getting a raise to a point $x'_i \in R$ with *income* = \$4750. We can denote the homogeneous class c of region R as R^c with the user selecting the desired class for their instance $c = c_d, c \in C$.

Property 2. (Minimum Robustness Guarantee). *A counterfactual region R as in Def. 3 provides a minimum robustness guarantee for x'. That is, given a real-world variation in feature values $v \in \mathbb{R}^n$ with $\|v\| \leq \alpha$, it is guaranteed that $f(x' + v) = c_d$. Thus the robustness of x' (per Def. 2) is at least α, $\psi(x') \geq \alpha$.*

The minimum robustness guarantee indicates to the user how much they can deviate from x' while remaining assured that they will achieve the desired outcome. For example, for a loan explanation with α = \$100, a user could target x' with *savings* = \$10,000 ± \$100. This allows users to plan for expected feature variations and achieves Objective 1 by describing what variations are guaranteed to result in the desired outcome.

Property 3. (Subset Consistency). *Given a counterfactual region R_1 with α_1 centered on point x'_1. If the user selects a point $x'_2 \in R_1$, a new region R_2 can be created centered on x'_2 with $\alpha_2 = \alpha_1 - \delta(x'_1, x'_2)$. Then, R_2 is guaranteed to be counterfactual per Def. 3.*

Prop. 3 can be easily proven by showing that R_2 will be a subset of R_1, i.e., $R_2 \subset R_1$. As $\alpha_2 < \alpha_1$ Prop. 3 also implies that subsets of regions must have smaller minimum robustness guarantees.

Remark 3. (Region Robustness) *Given counterfactual region explanations R_1, R_2 with radii α_1, α_2, the region with the larger radius is more robust to variations and therefore preferred.*

Remark 4. (Region Relevancy) *Given counterfactual region explanations R_1, R_2 centered on points x'_1, x'_2, the region with the smaller distance $\delta(x, x'_i)$ is more relevant and therefore preferred.*

Considering the *Robustness Relevancy Trade-Off* in Rem. 2, Remarks 3 and 4 present conflicting notions of region desirability. We address this in part by leveraging Prop. 1. By presenting the user a region explanation R_1 and allowing them to select their preferred counterfactual example $x'_2 \in R_1$, we empower the user to balance this trade-off for their circumstances (Objective 2). For example, if R_1 has x'_1 with *savings* = \$10,000 and α_1 = \$100, a user prioritizing a counterfactual with low distance may choose a point $x'_2 \in R_1$ which is closer to x than x'_1 (say x'_2 = \$9,950). This could result in a region R_2 with α_2 = \$50. The user would then target altering their *savings* = \$9,950 ± \$50.

To maximize the flexibility afforded to users, FACET provides counterfactual explanations in the form of a tuple (R, x^*), where x^* is the nearest point to x in R and indicates the minimum effort alteration required to fall in R. The user can then select a value $x'_i \in R$ with sufficient robustness. In Sec. 4.4, we design algorithms for generating the initial region that maximizes α without substantially impacting the explanation distance.

Counterfactual Regions as Hyperrectangles. While the above (radial) α provides a strong guarantee of region robustness, it may lead to some misunderstanding as it causes the acceptable variations in one feature to be dependent on the observed variations in all other features. For example, given R centered on x' with α = \$100, x' is robust to a variation of ±\$100 in income OR ±100 in savings, but not both. This could make it hard for a user to plan effectively, especially in higher dimensions. To make the robustness guarantee for each feature independent and easy to comprehend, we refine the concept of counterfactual regions to correspond to n-dimensional axis-aligned hyperrectangles H as below.

Definition 4. (Hyperrectangular Counterfactual Region H.) *Given a counterfactual region R as in Def. 3. For each feature j, let θ_{jL} and θ_{jU} specify a lower and upper bound. Then $\{\theta_{jL} \leq x[j] < \theta_{jU}\} \forall j$ describes the set of points x whose j^{th} feature value falls within these bounds. Let θ_{jL}, θ_{jU} be selected such that all points within the bounds are also in R. We can represent these bounds as:*

$$H = \begin{bmatrix} \theta_{1L} & \theta_{2L} & \dots & \theta_{nL} \\ \theta_{1U} & \theta_{2U} & \dots & \theta_{nU} \end{bmatrix}$$

This representation naturally fits user understandings of feature ranges such as \$1,000 < *income* < \$5,000. Further, as all points in H are also in R ($H \subset R$) they are guaranteed to be of homogeneous class (Prop. 1). Thus, a user can still select $x'_i \in H$ as a counterfactual example (Prop. 3) and recover a minimum robustness guarantee (Prop. 2). Representing regions as hyperrectangles also allows the minimum robustness guarantee specified by α to be divided among each feature to provide feature independence while maintaining the region properties from Def. 3. In theory any set of bounds which fits Def. 4 is sufficient, but per Rem. 3 large regions are desirable. Therefore, we can select the maximum H that is fully contained in the hypersphere defined by radius α around x' of R. This creates a hypercube which divides the minimum robustness equally across all axes. As some features may vary more or hold more importance to the user than others, we consider algorithms that generate non-square hyperrectangular regions in Sec. 4.4.

2.3 FACET Explanation Query Language

As motivated in Sec. 1, the selection of the best counterfactual explanation depends on a user's real-world situation. Thus, to assure personalized explanation, FACET generates counterfactual region explanations in an interactive dialog with the user. To do this, we design a query language to express explanation preferences. A FACET example query is shown in Fig. 2.

```
1 WITH R AS COUNTERFACTUAL-REGIONS(F)
2 SELECT TOP-EXPLANATIONS(k) FROM R
3 WHERE R.Class = DesiredClass
4 WITH UNALTERED FEATURES: R.Gender, R.Rent
5 WITH WHAT-IF SPECULATION
6      R.Income > $6,000 AND R.Income < $10,000
7      AND R.Rent = $2,000
8 ROBUST BY R.Widths >= 2v
9 ORDER BY DISTANCE(X) WITH WEIGHT W
```

Fig. 2. Example query for FACET's explanation analytics.

We adopt a SQL-like syntax as it is more precise than natural language, easily understood by users, and allows for powerful chaining of query clauses. We tailor each clause to meet one of the following FACET design criteria for effective explanation analytics.

- **Explanation Relevancy:** For an explanation to be useful, the generated counterfactual should be substantially similar to the user's instance as measured by δ. Given a set of counterfactual regions of the desired class $\mathcal{R}^{c_d}$ and an instance x, the user may want to find the counterfactual region R^* that is closest to x.

$$R^* = \underset{R_i \in \mathcal{R}^{c_d}}{\arg\min}\ \delta(x, R_i) \tag{1}$$

  where $\delta(x, R_i)$ is the distance between x and the nearest point in R_i. As the user may want to compare several regions of similar distance, FACET supports querying for the k nearest regions. These semantics are expressed in Lines 1-3 and 9 of Fig. 2.
- **Unalterable Features:** The user can restrict a set of features whose values should not be altered during explanation as expressed in the `UNALTERED FEATURES` clause in Line 4 of Fig. 2. These could include factors meaningfully not within their control such as race and gender or features such as rent which they may deem too costly to alter. We specify that a counterfactual region meets the condition for not altering a feature when at least one point in the region matches the existing restricted feature value. For example, given an applicant with $rent = \$3{,}000$, a region H with bounds $\$1{,}000 < rent < \$5{,}000$ would not alter their rent.
- **What-If Restrictions.** A user may also want to restrict the value of one or more features to a specific value or range of values. These semantics naturally model "what-if" querying. Such as "what if I got a raise?". Constructing these as "ranged conditional queries" using the `WHAT-IF SPECULATION` clause allows users to account for the non-exact nature of such hypotheticals, e.g., not knowing the exact dollar amount of a possible raise. An example of this is shown in Lines 5-7 in Fig. 2. If a user overly constrains a query, e.g., requiring $income < \$1$, FACET may not find an explanation and will prompt the user to relax their constraints.
- **Feature Prioritization.** As users may have more difficulty (or a preference for) affecting the values of some features over others, FACET's analytics allow the user to assign a weight w_j to each feature to prioritize explanations which alter easily changeable features first. Given this weight vector w, FACET's explanation analytics incorporate this user preference as an optional weighting in the ordering of results as shown in Line 9 of Fig. 2.

- **Explanation Robustness.** To account for feature value fluctuations (e.g., varying savings balance), users may require an explanation be robust to a certain scale of perturbation in one or more features. FACET's analytics allow the user to specify the minimum required robustness for a region. Given hyperrectangular regions, robustness corresponds to the width of the hyperrectangle along a given feature axis. FACET supports this through the `ROBUST BY` clause as in Line 8 of Fig. 2. As a syntactic shortcut, an equal minimum robustness guarantee $p \in \mathbb{R}$ on all dimensions can be expressed by `R.MinWidth >= 2p`. For features with different expected variations, FACET supports the condition `R.Widths >= 2v`, where $v \in \mathbb{R}^n$ is a vector representing the minimum required robustness along each feature axis.

3 FACET EXPLANATION ANALYTICS SYSTEM

Fig. 3 depicts the FACET framework. To ensure near real-time interactivity of explanation exploration in spite of the complexity of the explanation space, FACET adopts a database-inspired approach of precompute-and-index followed by query-by-lookup. *Preprocessing Stage:* First, the

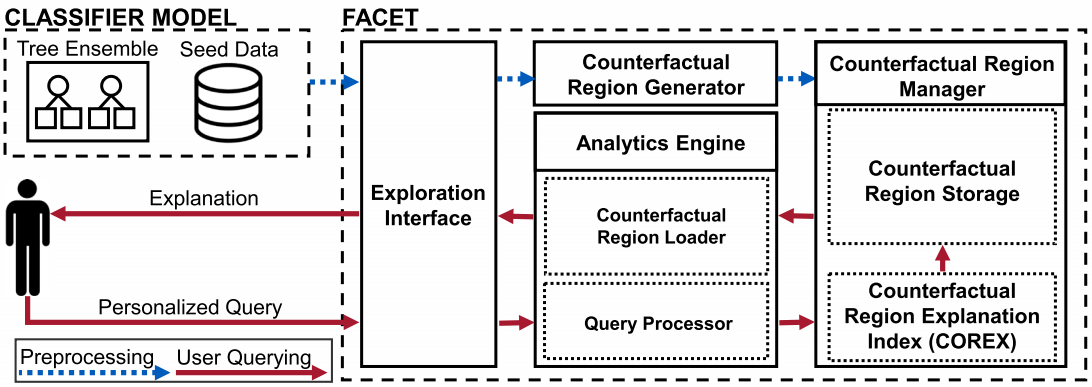

Fig. 3. Overview of FACET explanation system.

model creator provides FACET with the trained tree ensemble and a set of representative domain data, e.g., the model's training data. Then, FACET's *Counterfactual Region Generator* (Sec. 4) intelligently creates a set of regions representative of the model's learned behavior. While multiple approaches are feasible and could be plugged into our FACET framework, FACET's generation algorithm leverages the provided domain data to identify the portions of the feature space which have realistic feature value combinations. FACET prioritizes the generation of regions in these areas and stores them in the *Counterfactual Region Manager*.

Next, we construct FACET's counterfactual region explanation index (*COREX*) on these regions (Sec. 5). COREX is a bitvector-based index for compactly encoding the high dimensional spatial data used to represent counterfactual regions. By encoding information on the location and size of each counterfactual region, COREX is able to efficiently retrieve regions from any arbitrary portion of the input space efficiently, even for large explanation spaces.

Exploration Stage: During online exploration, users submit explanation queries which include their personalized constraints and priorities via FACET's *explanation query language* (Sec. 2.3). These queries are processed by the *FACET Analytics Engine* (Sec. 6). During query execution, the Analytics Engine uses COREX to quickly narrow down on a set of relevant counterfactual regions as candidates for consideration. These candidate regions are then fetched by the FACET region manager and a counterfactual example is selected from each region. A final selection of region-example pairs are returned to the user. Upon examining the returned explanations, the user may choose to adapt their explanation request to incorporate more refined considerations and interactively explore until they find the best personalized explanation for their needs.

4 COUNTERFACTUAL REGION GENERATION

4.1 Counterfactual Regions in Trees

For simplicity, we first describe the process of generating counterfactual region explanations for decisions of binary decision trees.

Remark 5. (Decision Tree Partitioning.) *In a decision tree, each internal node creates a partition of the feature space, i.e., the node splits the set of all possible points between its two children nodes. As these nodes split on a single feature, each partition is aligned with an axis of the feature space. This continues until a leaf node is reached where all points in that partition are assigned the same class.*

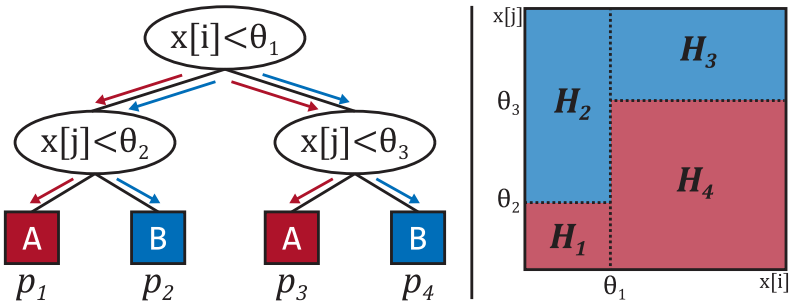

Fig. 4. Example of FACET's tree region generation.

Therefore, by walking each path from the root to a leaf, we can generate a set of rules that describe a portion of the input space of a homogeneous class. For example, in Fig. 4, the path p_1 from the root to the leftmost leaf defines the rule set $\{x[i] < \theta_1 \wedge x[j] < \theta_2\}$ and creates a partition of points assigned class A. Further, as each split of the space is axis-aligned, the resulting partition must be a hyperrectangle in the feature space. We can treat this partition as a counterfactual region of class A as abstracted in Def. 3 as all points falling into this region are guaranteed to be of class A and can represent counterfactual examples for an instance with $y = f(x) = B$.

Remark 6. (Tree Region Extraction.) *For each leaf, the set of decision rules that define its region is a conjunction of all internal nodes leading to that leaf. There will be at least one and at most D such rules distributed across the n features, where D is the depth of the leaf. Through simplification, we can obtain at most $2n$ rules. The resulting rule set can be compactly represented as a matrix H of the form in Def. 4, with the acceptable values for feature j bounded by $\theta_{jL} \leq x[j] < \theta_{jU}$. For features j not constrained by the rule set in question, we set θ_{jU} and θ_{jL} to $\pm\infty$.*

Algorithm 1 FACET Decision Tree Explanation

```
procedure EXPLAIN(t, x)
    Regions = EnumerateLeafRegions(t)
    BestDist = ∞, BestR = None
    for R ∈ Regions do
        if R.Class == c_d ∧ δ(R, x) < BestDist then
            BestDist = δ(R, x), BestR = R
    return BestR
procedure ENUMERATELEAFREGIONS(t)
    Paths = EnumeratePaths(t), Regions = []
    for P ∈ Paths do
        R = SimplifyRules(P) ▷ Extracts Hyperrect. Region
        Regions.append(R)
    return Regions
```

Counterfactual Region Explanation in Trees. Following the observations above, FACET can generate counterfactual region explanations for decision trees. A simple algorithm for the nearest region explanation of a decision tree t is shown in Alg. 1. In practice, the set of all hyperrectangular regions $\mathcal{H}$ can be enumerated once during preprocessing, and be indexed and stored separated by class, ($\mathcal{H} = \mathcal{H}^A \cup \mathcal{H}^B$). This can easily be extended to support an arbitrary number of classes.

Remark 7. (Tree Region Coverage.) *As every point in the feature space falls in exactly one leaf, $\mathcal{H}$ will fully tile the space with a total of 2^D hyperrectangular regions. Any valid counterfactual region will be a subset of the regions in $\mathcal{H}$ as per Prop. 3.*

4.2 Counterfactual Regions in Tree Ensembles

Next, we characterize counterfactual regions in tree ensembles. Here we consider explanation of ensembles aggregated by majority vote and later generalize to other decision procedures (Sec. 7.1).

Remark 8. (Hyperrectangles in Tree Ensembles.) *Given a tree ensemble $\mathcal{T} = \{t_1 ... t_T\}$ aggregated by majority vote, a point x is classified as class c when at least $\lceil T/2 \rceil$ trees predict $t(x) = c$. As the leaves of each tree represent a hyperrectangle of the form in Def. 4 in the feature space (Sec. 4.1), every point of class c must fall into at least $\lceil T/2 \rceil$ hyperrectangles of class c (and at most T such hyperrectangles). Fig. 5 shows an example of this for an ensemble of $T = 3$ trees.*

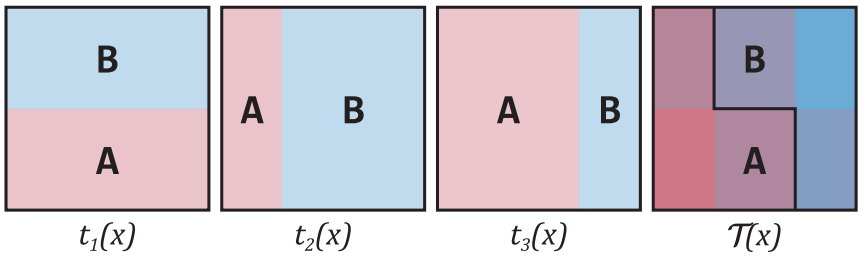

Fig. 5. Example leaf hyperrectangles for an ensemble.

Per Rem. 8, we denote the set of leaf hyperrectangles which x falls into as $\mathcal{H}_x^c$, with $|\mathcal{H}_x^c| = \left[\lceil T/2 \rceil, T\right]$. Then, to generate valid counterfactual regions of a homogeneous class (Prop. 1), we take the intersection of $\lceil T/2 \rceil$ of these hyperrectangles as in Eq. 2.

$$R^c_{(\mathcal{T})} = \bigcap_{i=1}^{\lceil T/2 \rceil} H_i = \begin{bmatrix} \theta_{1L} & \theta_{2L} & \dots & \theta_{nL} \\ \theta_{1U} & \theta_{2U} & \dots & \theta_{nU} \end{bmatrix} \tag{2}$$

Remark 9. (Hyperrectangle Intersections.) *Let $\mathcal{H}^c$ denote a set of overlapping hyperrectangles of class c. We can intersect $\mathcal{H}^c$ by taking the smallest upper bound $\theta_{jU} = \min_i H_{i:jU}$ and largest lower bound $\theta_{jL} = \max_i H_{i:jL}$ from all $H_i \in \mathcal{H}^c$ along each dimension j.*

Rem. 9 rests on the properties that (1) the intersection of two hyperrectangles is also a hyperrectangle, and (2) leaf hyperrectangles are axis-aligned. While intersecting more than $\lceil T/2 \rceil$ leaf hyperrectangles will generate a valid counterfactual region, we note that the intersection of any two hyperrectangles is at most the size of the larger. Therefore, taking additional intersections only reduces the size and robustness of the resulting region (Rem. 3).

Remark 10. (Region Existence.) *From Rem. 8, we saw that every point in the feature space is contained in at least $\lceil T/2 \rceil$ leaf hyperrectangles. Rem. 9 finds that intersecting $\lceil T/2 \rceil$ such hyperrectangles creates a counterfactual region. Thus we can create a counterfactual region of the observed class at any arbitrary point in the feature space.*

In Sec. 4.4, we will exploit Rem. 10 to intelligently select the location and number of counterfactual regions we generate to prioritize realistic counterfactual explanations.

4.3 Complexity of Region Set

As we established in Eq. 2, a counterfactual region for an ensemble of trees is defined as the intersection of $\lceil T/2 \rceil$ leaf hyperrectangles. Here we consider the feasibility of enumerating all possible such regions for an ensemble as we did in Alg. 1 for a single tree.

Remark 11. (Complexity of Region Space.) *For an ensemble with L leaves, there are approximately ${}^{L}C_{\lceil T/2 \rceil}$ intersections of $\lceil T/2 \rceil$ leaf hyperrectangles. This results in a number of possible counterfactual regions which is exponential in the number of leaves, which is itself exponential in the tree depth. With $L \approx 2^D T$ for balanced binary trees of depth D we find the possible number of regions to be:*

$$N_{regions} \propto \frac{2^D T!}{\lceil T/2 \rceil! (2^D T - \lceil T/2 \rceil)!}$$

For an ensemble of $T = 100$, $D = 5$, $N_{regions} > 10^{100}$ which is far too many regions to reasonably check. Additionally, applying branch-and-bound to progressively intersect leaf hyperrectangles is intractable as this requires computing the ${}^{L}C_2$ pairwise intersections of leaves, an even larger value.

4.4 FACET Region Generation

As analyzed in Sec. 4.3 the number of possible counterfactual regions is too large to fully enumerate. However, as our goal is to generate actionable explanations, it is not necessary to generate all intractably many possible counterfactual regions. Instead, we propose a heuristic for selectively generating a large, but tractable, subset of regions of high utility.

Remark 12. (Realistic Region Utility.) *For a region explanation to be actionable for a user, it must contain realistically achievable combinations of feature values, e.g., a loan explanation that requires $\$1 < income < \$100$ and $\$10M < rent < \$50M$ is unlikely to be useful to any real user. An explanation is considered realistic when it lies on or near the data manifold of real instances [11, 12, 37].*

Selective Region Generation. Per Rem. 10, we can generate a counterfactual region of the majority class at any point. Further, Rem. 12 finds that regions are of high utility when they contain realistic feature value combinations as determined by proximity to the data manifold. Therefore, given a set of seed points X which follow the realistic data manifold, we can generate a set of counterfactual regions which contain realistic feature value combinations. While many seed point selection strategies are possible, if explanation is done in collaboration with the model owner (e.g., the bank approving loans) it is reasonable to select X as ensemble training data or other domain-provided representative data. With each seed point used to generate one counterfactual region, we can augment X as in Eq. 3 to generate an arbitrary number of regions.

$$\widetilde{x}_i \overset{\text{iid}}{\sim} \text{Unif}\{X\} + \mathcal{N}(\mu, \sigma^2), \; i = 1, \ldots, N_r \tag{3}$$

Algorithm 2 FACET Ensemble Region Generation

```
procedure GenerateRegions(T, X̃)
    LeafRects = [[] * T], Regions = []
    for i = 1, ..., T do
        LeafRects[i] = EnumerateLeafRegions(t_i)
    for x̃ ∈ X̃ do
        PredLeaves = Predict(T, x̃)
        PredRects = MatchRects(PredLeaves, LeafRects)
        R = IntersectRects(PredictedRects)
        if R not in Regions then Regions.append(R)
    return Regions
```

The data augmentation approach of Eq. 3 incentivizes the generation of a diverse set of regions which fall on or just off the data manifold to provide good coverage of the feature space. The value of σ can also be adjusted for each dataset to control the explanation diversity. Using these augmented seed points $\widetilde{X}$, we selectively generate regions for ensemble $\mathcal{T} = \{t_1 ... t_T\}$ as in shown Alg. 2. Here `Predict` identifies the leaf each tree predicts for $\widetilde{x}$, and `MatchRects` selects the hyperrectangle which corresponds to that leaf. The result of this is a total of T hyperrectangles $\mathcal{H}_{\widetilde{x}_i}$ where at least $\lceil T/2 \rceil$ are of the observed class $\mathcal{T}(\widetilde{x}_i) = c_o$. We denote this set $\mathcal{H}^{c_o}_{\widetilde{x}_i}$. To obtain a counterfactual region of class c_o, we can take the intersection of $\lceil T/2 \rceil$ hyperrectangles from $\mathcal{H}^{c_o}_{\widetilde{x}_i}$ as in Eq. 2.

Intersection Selection. As there are exponentially many (${}^{T}C_{\lceil T/2 \rceil}$) valid selections of intersections to choose from in Alg. 2, we design `IntersectRects` using the following size maximizing heuristic (per Rem. 3). For each hyperrectangle H_i, we apply a selection priority function $\rho(H_i)$ as the total number of unbounded edges, i.e., the number of axis directions along which the hyperrectangle makes no restrictions (of the $2n$ total edges per region).

$$\rho(H_i) = \sum_{j=1}^{n} H_{i:jU} = \infty + \sum_{j=1}^{n} H_{i:jU} = -\infty \tag{4}$$

We then sort $\mathcal{H}^{c_o}_{\widetilde{x}_i}$ by largest ρ and take the intersection of the first $\lceil T/2 \rceil$ hyperrectangles. This minimizes the number of restrictions placed on the region (and therefore the user) and encourages generation of large regions with strong minimum robustness guarantees. The final set of counterfactual regions $\mathcal{H}_{(\mathcal{T})}$ can then be indexed and analyzed using FACET's explanation analytics. We experiment with generating between 10^2 and 10^6 regions in Sec. 7.6.

4.5 Generalizing FACET to Other Ensembles

FACET's region generation in Alg. 2 is designed to be extensible to other ensemble types by simply altering the `Intersect Rects` procedure (Sec. 4.2) to suit the ensemble aggregation technique. This allows FACET's other explanation indexing and querying processes to remain the same.

Soft Voting. If the ensemble from Sec. 4.2 is aggregated by soft voting – i.e., the outcome is the class with largest average probability across predicted leaves – we alter `Intersect Rects` as follows. Rather than intersecting exactly $\lceil T/2 \rceil$ hyperrectangles, hyperrectangles are intersected in order of Eq. 4 until class c reaches a majority probability. The resulting regions will match Def. 3.

Gradient Boosting Trees. Given an ensemble $\mathcal{T}$ trained by gradient boosting (as in XGBoost), each tree t_i is a regression tree predicting the errors of t_{i-1}. If we assume $\mathcal{T}$ is trained with log loss for binary classification, the class $\mathcal{T}(x) = c_o \in \{c_A, c_B\}$ of a point x is determined as follows. Let $\tau(x) = t_0(x) + \sum_{i=1}^{T} \eta t_i(x)$ be the log-odds of x where η is a scalar learning rate, and $t_0(x)$ some initial prediction (typically class priors). Then $\mathcal{T}(x) = c_A$ when $sigmoid(\tau(x)) > 0.5$, c_B otherwise. In this case, many options are possible for `Intersect Rects`, but we propose two strategies. 1) the *Complete Intersection* of the predicted leaf rects, and 2) the *Progressive Relaxation* of this intersection. In the relaxation case we compute the class probabilities from intersecting all leaves and then progressively remove intersections in reverse ensemble order until the probability can no longer guarantee $\mathcal{T}(x) = c_o$. Both produce regions meeting Def. 3. We evaluate these strategies in Sec. 7.3.

5 FACET INDEX: DESIGN AND ALGORITHMS

Next, we describe FACET's counterfactual region index COREX which is designed to support the execution of FACET's analytics queries over a large number of counterfactual regions.

5.1 Index Design Requirements

As outlined in Sec. 2.3, FACET's analytics seek the nearest region subject to a set of personalized constraints and weights. As each region is represented as an n-dimensional hyperrectangle (where n is the dimensionality of the input data), the explanation analytic task is equivalent to a high-dimensional personalized spatial k-nearest neighbor search. An index for this task must be

- *Expressive* to handle complex user considerations such as feature value constraints, unalterable features, and feature prioritization
- *Scalable* in the number of dimensions to support high dimensional decision spaces with multiple restrictions per dimension, yet
- *Lightweight* to work inside memory and service queries in real time per the need for human-in-the-loop interactivity.

As we discuss in Sec. 9, lightweight indexing for high dimensional spatial data is understudied, with off-the-shelf indices failing to meet the above set of requirements. Thus we develop COREX (counterfactual region explanation index) to enable the selection of arbitrary subsets of the generated counterfactual regions based on their location in the feature space. This enables FACET's index-aware queries to identify regions that are proximal to the explained instance and constraints. To achieve this, COREX develops a set of predicates that intelligently partition the feature space and encodes the location of each region with respect to these predicates as a set of bit vectors. As we show in Sec. 7.6, this approach leverages the efficient bitwise operations of modern CPUs to support kNN searching in near real-time.

5.2 FACET's COREX Region Encoding

We design a highly efficient bit vector based encoding of counterfactual region locations using the following strategy.

Definition 5. (Partitioning Predicate P.) *Given a counterfactual region H, let a predicate P partition the feature space on feature j by some threshold θ such that $P(H) = True$ if and only if some point in H meets P.*

$$P(H) \leftrightarrow \exists x_i \in H \mid x_i[j] < \theta$$

Definition 6. (Bit Vector Region Encoding b.) *Given a set of counterfactual regions $\mathcal{H}$ and a predicate P. Let a bit vector $b = bitvector(\mathcal{H}, P)$ be a sequence of bits of length $|\mathcal{H}|$ where $b[i] = 1$ indicates that $P(H_i) = True$ and $b[i] = 0$ indicates $P(H_i) = False$.*

Tab. 1 shows example bit-vector values for the predicate $P_1 = (Income < \$5{,}000)$. As counterfactual regions are represented as bounded axis aligned hyperrectangles, $P = (j < \theta)$ can be evaluated simply with respect to the upper bound of the region $\theta < H_{i:jU}$, while $P = (j > \theta)$ with respect to the lower bound $\theta > H_{i:jL}$. This allows for efficient computation of a predicate's value without explicitly checking every point in the region.

Record	Income, (θ_L, θ_U)	Predicate, P_1 $Income < \$5{,}000$	Bit Vector
H_1	$(0, 1500)$	True	1
H_2	$(1700, 5700)$	True	1
H_3	$(7000, 9000)$	False	0

Table 1. FACET bit encodings for counterfactual regions.

In COREX, we create several predicates for each feature and precompute their values for all regions during preprocessing. This allows COREX to rapidly evaluate the location of regions by performing bitwise operations on the bit vectors for each predicate.

Remark 13. (Bidirectional Predicate Indexing.) *Given two predicates P_1, P_2 with associated bit vectors b_1, b_2 precomputed over a set of counterfactual regions $\mathcal{H}$. If the predicates partition a feature j s.t. $P_1 = (j < \theta_1)$, $P_2 = (j > \theta_2)$ and $\theta_2 < \theta_1$. We can identify the set of counterfactual regions which fall in this area as $P_3 = P_1 \wedge P_2$ which can be efficiently computed as the bitwise AND, $b_3 = b_1$ AND b_2.*

By parsing conditional constraints into bit vector lookups, FACET can search for matching regions substantially faster than via direct evaluation. For example, if $j = income$, $\theta_1 = \$2{,}000$ and $\theta_2 = \$5{,}000$, the constraint $\$2{,}000 < income < \$5{,}000$ can be converted into P_3 and evaluated via index lookup. Evaluating P_3 directly on the records in Tab. 1 requires subtracting θ from each region's bounds and checking the result. This totals four operations per region compared to a single bitwise AND for the lookup. Further, as the bit vector value for each region is a single bit, they can be packed into words. This allows COREX to compute a predicate on up to 64 regions in one operation compared to the 256 operations otherwise needed. By creating multi-word bit vectors, COREX can manage arbitrarily many regions with comparison costs scaling in the number of words. This grows much slower than the number of regions and remains highly efficient due to CPU pipelining of simple operations and the execution of multiple bitwise operations per instruction.

While not every query will precisely match these predicates, we can intelligently select the split values and use precomputed bit vectors to greatly reduce the number of regions we must consider. For example, if the user conditioned $income < \$4{,}000$ rather than $\$5{,}000$ in their query, we could select the nearest larger predicate (still P_1) to compute P_3 as before, and filter the resulting small set of regions to ensure they meet the constraint. We can also check if the defined region is non-empty with a single bit count operation. This is exploited in FACET's query processing to efficiently check if the lookup finds any matching regions. In Sec. 5.3, we detail COREX's strategy for predicate selection, while in Sec. 6 we describe how COREX is leveraged for use in additional analytics.

5.3 FACET's COREX Layout

To enable rapid selection of counterfactual regions based on their location, COREX generates a set of predicates which partition the feature space along each dimension. These predicates are then encoded in bit vectors and stored such that ANDing together bit vectors from multiple axes selects just the regions which fall in a specific portion of the feature space. COREX selects split values along each axis to evenly distribute the regions for optimal efficiency. We also create a separate index for regions of each class.

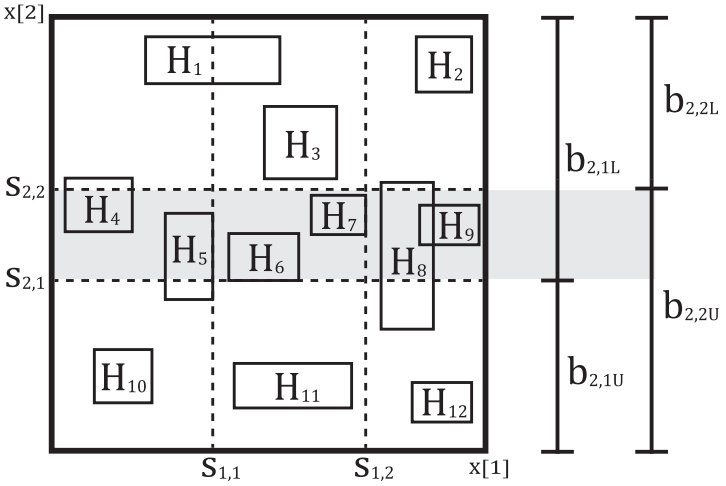

Fig. 6. FACET bit vector index with $|\mathcal{H}^c| = 13$ and $m = 2$.

Definition 7. (Selected Split Values S_j.) *Given a feature space $\mathcal{X} \in \mathbb{R}^n$ and a set of hyperrectangular counterfactual regions of class c, $\mathcal{H}^c$. Let $S_j = s_{j,1} \ldots s_{j,m}$ be a set of m split values along dimension j. Let $\mathcal{B}_j$ be the set of unique upper and lower bounds along dimension j sorted in increasing order ($|\mathcal{B}_j| \leq 2|\mathcal{H}^c|$). Then select a split value $s_{j,k}$ as every $|\mathcal{B}_j|/(m+1)$ bounds, distributing the remaining $|\mathcal{B}_j|\%(m+1)$ bounds evenly among the first intervals.*

Repeating this for each axis results in an n-dimensional grid structure of non-zero width intervals with an approximately equal number of hyperrectangles along each row and column. An example of the partitioned space is shown in Fig. 6 for a set of twelve hyperrectangles. Note how the width of each interval is adjusted to contain approximately two regions in each grid cell.

Bi-directional Predicate Construction. To partition the feature space based on these split values, we construct two predicates per split. For the kth split value of feature j, $s_{j,k}$, we define the predicate bounded below by the split as $P_{j,kL} = (x[j] \geq s_{j,k})$ and the space bounded above as $P_{j,kU} = (x[j] < s_{j,k})$. This corresponds to $2m$ predicates which can be applied to $\mathcal{H}^c$, to generate $2m$ bit vectors each of length $|\mathcal{H}^c|$. As some hyperrectangles span across the split values, they will have points on either side of θ. Thus, the lower bound bit vector $b_{j,kL} = P_{j,kL}(\mathcal{H}^c)$ is not necessarily the inverse of upper bound bit vector $b_{j,kU} = P_{j,kU}(\mathcal{H}^c)$.

COREX Access Operation. The bit vectors COREX constructs can then be used to quickly look up hyperrectangles that fall into a particular portion of the feature space. For example, in Fig. 6, the grey region denotes the portion of the feature space that matches the predicate $P = (s_{2,1} \leq x[2] < s_{2,2})$ selected by the bit vector computation $(b_{2,2U}\ AND\ b_{2,1L})$. Our index returns the hyperrectangles $\{H_4, H_5, H_6, H_7, H_8, H_9\}$, as they match this predicate.

6 FACET EXPLANATION QUERY PROCESSING

We now introduce our analytics execution engine that leverages FACET's COREX to efficiently process user interaction queries to serve explanations in real time.

6.1 KNN Explanation Search

As in Sec. 2.3, the user may want to find relevant counterfactual regions as those near the explained instance. We offer two processing strategies for kNN explanation search. (1) a custom low-overhead strategy for $k = 1$, and (2) a general strategy for kNN with $k > 1$. As computing the distance between x and every region in $\mathcal{H}^{c^d}$ is prohibitively expensive, FACET leverages COREX to look-up regions in the neighborhood of x. Then, we compute the precise distance to these nearby regions and return the k-nearest.

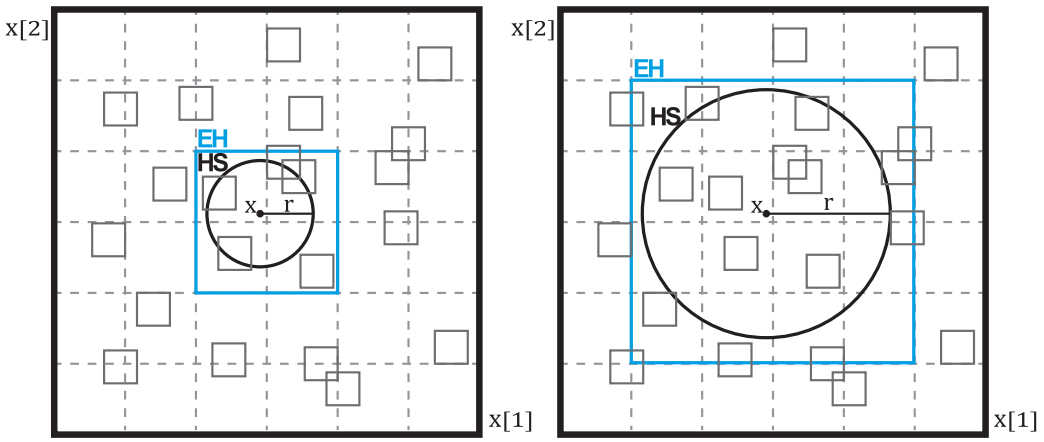

Fig. 7. Example of FACET progressive radius kNN search.

Definition 8. (Neighborhood Hypersphere HS.) *Given an instance x, let the neighborhood of x be defined by a hypersphere HS of radius r centered on x such that $HS = \{x_i \in \mathbb{R}^n \mid \delta(x, x_i) < r\}$.*

To search COREX for counterfactual regions in this neighborhood, we find the set of predicates which minimally enclose HS by choosing the nearest smaller/larger split value along each axis.

Definition 9. (Enclosing Hyperrectangle EH.) *Given a hypersphere HS and a set of m split values S_j along axis j. Select the nearest lower $s_{jL} = \max\{s \in S_j \mid (x[j] - r) - s > 0\}$ and nearest higher split values $s_{jU} = \min\{s \in S_j \mid s - (x[j] + r) > 0\}$. Then let EH be a hyperrectangle where*

$$EH = \begin{bmatrix} s_{1L} & s_{2L} & \dots & s_{nL} \\ s_{1U} & s_{2U} & \dots & s_{nU} \end{bmatrix}$$

We can then select the counterfactual regions which fall into EH by ANDing together the bit vectors from each axis as $b_{EH} = AND_{j=1..n}\ b_{jL}\ AND\ b_{jU}$ where b_{jL} and b_{jU} are the bit vectors corresponding to the predicates $P(j \geq s_{jL})$ and $P(j < s_{jU})$ respectively. Here $b_{EH}[i] = 1$ iff the counterfactual region H_i falls within EH. By performing a bit-count on b_{EH}, we can determine the number of regions in this space. Note that some regions may fall in EH but outside HS and are filtered for exclusion from the neighborhood search. Examples of HS and EH are shown in Fig. 7.

Algorithm 3 FACET Region Neighborhood Search

```
1:  procedure COREXNN(x)
2:      BestDist = ∞, BestR = None
3:      r = rinit, bSearched = BitVector([0 * N_regions])
4:      while BestR is None ∧ BestDist > r do
5:          HS = HyperSphere(x, r)
6:          EH = EnclosingSplits(HS)
7:          bU, bL = COREXVectors(EH)
8:          bMatches = BitVector([0 * N_regions])
9:          for j ∈ 1...n do
10:             bAxisMatch = bL[j] AND bU[j]
11:             bMatches = bMatches AND bAxisMatch
12:         bNew = bMatches XOR bSearched
13:         if CountBits(bNew) > 1 then
14:             Regions = COREXLoad(bNew)
15:             for R ∈ Regions do
16:                 if δ(R, x) < BestDist then
17:                     BestDist = δ(R, x), BestR = R
18:         r = r + rstep
19:     return BestR
```

Neighborhood Search. Using this neighborhood concept, FACET searches for counterfactual region explanations using the progressive radial search shown in Alg. 3. As regions are indexed by class, we search the index for the desired class. By searching with increasing radii, we can identify the nearest counterfactual region to x as shown in Fig. 7. Further, by tracking the bit vector `bSearched`, we can determine which regions have already been checked. This allows FACET to avoid recomputing the distance to the same region twice. To modify Alg. 3 to search for k-nearest regions, we use a priority queue of tuples $(distR, R)$, with the search completing when the queue contains k elements and the k^{th} element has $dist \leq r$. Regions further than the k^{th} element are automatically discarded to minimize memory overhead.

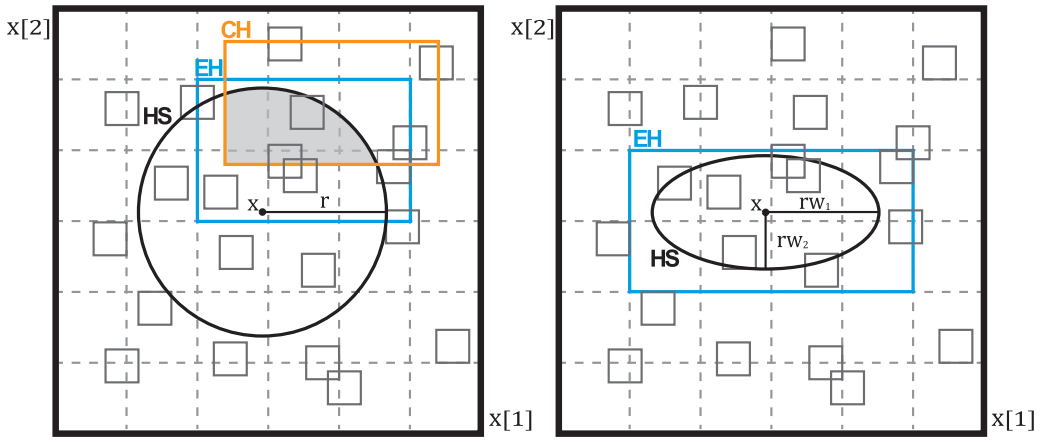

Fig. 8. Constrained and prioritized FACET searches.

6.2 Explanation Refinement Processing

Users may want to constrain the features used in an explanation (Sec. 2.3). Constraints include both unalterable features $\mathcal{U}$ (whose values can't be changed) and restricted feature values when the user provides a range of acceptable values along one or more features.

Definition 10. (Constraint Hyperrectangle CH.) *Let $\mathcal{A}$ be a set of tuples (a_{jL}, a_{jU}) provided by the user where a_{jL} and a_{jU} represent the minimum and maximum allowed values for a feature j. Let $\mathcal{U}$ be a set of user provided unalterable features. Then, for all features $\forall j \in \mathcal{U}$ append a tuple $a_{jL} = a_{jU} = x[j]$ to $\mathcal{A}$. For all features not yet restricted in $\mathcal{A}$, let $a_{jL} = -\infty$, $a_{jU} = \infty$. The result of this is a set of minimum and maximum allowable values for all n features which defines a constrained search area in the feature space. We can represent this as a constraint hyperrectangle CH.*

$$CH = \begin{bmatrix} a_{1L} & a_{2L} & \dots & a_{nL} \\ a_{1U} & a_{2U} & \dots & a_{nU} \end{bmatrix}$$

To consider only this constrained area during the nearest neighbor search, CH can be intersected with the neighborhood hypersphere HS, before the enclosing hyperrectangle EH is constructed. The result is depicted in Fig. 8 for a constrained search from Fig. 7.

6.3 Prioritized Feature Explanations

As changing certain features may have different real-world difficulties for the user, FACET's analytics incorporate a set of feature prioritizations for the explanation search.

Definition 11. (Prioritized Explanation Distance $\delta(x, x', w)$.) *Given an instance x, explanation x', and a weight vector $w \in \mathbb{R}^n$ where w_j is proportional to the ease of altering feature j. Let $\delta(x, x', w)$ be a weighted distance function such that features with $w_j = 1$ have a normal cost, and features with $w_j > 1$ have a decreased cost.*

$$\delta(x, x', w) = \sqrt{\sum_{j=1}^{n} \left(\frac{x_j - x'_j}{w_j} \right)^2}, w_j \geq 1$$

Using this distance function, FACET defines the search area as $\{x_i \in \mathcal{X} \mid \delta(x, x_i, w) \leq r\}$. This alters the search region from a hypersphere to a hyper-ellipsoid whose semi-axes are scaled by the weights vector. Fig. 8 shows this for $w = [1\ 2]$. The hyper-ellipsoid search region allows FACET to seamlessly integrate feature prioritization into the kNN search and empowers users to search with personalized weights without having to construct extra indices.

Global Feature Weighting. If FACET is given non-normalized data, we include a global feature weighting scheme to ensure that no feature is unwantedly favored for alteration. For this "equal weighting" case, we compute a weights vector $w^{(g)}$ proportional to the range of each feature as $w^{(g)} = range(X) \odot min\{range(X)\}$. $w^{(g)}$ is then combined by element-wise multiplication with the user weights during querying ($w = w^{(g)} \odot w^{(u)}$). Model designers also may set their own global weights directly as desired using their knowledge of the specific domain.

6.4 Minimum Robustness Constraint

As shown in Sec. 2.2, counterfactual regions provide critical information about the robustness of an explanation to perturbation. To incorporate users' real-world robustness constraints, FACET provides two levels of abstraction. The first strategy is a minimum radial perturbation requirement. Given a user provided value $p \in \mathbb{R}$, FACET will return only counterfactual regions which include at least one point that falls at least distance p from all of the containing region's edges ($\exists x'_i \in R \mid \forall j \delta(\theta_{jL}, x'_i) \leq p \wedge \delta(\theta_{jU}, x'_i) < p$). In Alg. 3 this constraint is checked against regions which fall within the search hypersphere. For features with different expected variations FACET also supports a minimum robustness constraint along each axis. This takes the form of a vector $v \in \mathbb{R}^n$, where $v[j]$ specifies the minimum perturbation along each feature which the explanation must be robust to. Querying with minimum robustness constraints is thus a straightforward but powerful operation for ensuring that explanations are actionable for the user.

7 EXPERIMENTS

7.1 Experimental Setup and Methodology

Experimental Setup. Experiments are run on a virtual machine with 4 EPYC-7543 CPU cores and 16GB of RAM. All code is developed in Python 3 and is released on GitHub[2] for reproducibility.
Methods. We evaluate FACET against state-of-the-art model agnostic and model transparent counterfactual explanation methods.

- **MACE** [12] is the state-of-the-art model agnostic method, it uses predicted class probabilities to estimate the model's behavior in the neighborhood of the explained instance.
- **OCEAN** [37] is the state-of-the-art model transparent method, it encodes explanation as a mixed integer programming task solved via the Gurobi optimizer.
- **RFOCSE** [16] is a recent technique for explaining tree ensembles which produces sets of counterfactual examples by fusing the ensemble into a single tree during explanation.
- **AFT** [45] is a popular method for explaining tree ensembles which generates an altered instance for each tree leaf and checks to see if any hold as explanations for the ensemble.

AI Models and Datasets. To demonstrate FACET's capacity to generate counterfactual explanations, we train a Random Forest (RF) or Gradient Boosting Classifier (GB) ensemble using the Scikit Learn [39] package on eight benchmark datasets. Results for the five datasets in Tab. 2 are shown here, with an additional three on GitHub.[2] Data is preprocessed as in [12, 37] by one-hot encoding categorical features, numeric encoding discrete features, and normalizing numeric features on [0,1]. Requirements for validity of these encodings are enforced by FACET and applied to all methods except AFT (due to lack of support). For each dataset, we train ensembles of various configurations using an 80/20 train/test split and sample a batch of 20 instances from the test set for explanation. Training data is used as FACET's seed data per Eq. 3 and Scikit parameters are left as default. We set FACET's number of regions and index granularity per the results of Sec. 7.6.

[2]https://github.com/PeterVanNostrand/FACET

Dataset	Abbrv	N	n	n_{OH}
Adult [14]	adult	45222	11	41
Breast Cancer Wisconsin [14]	cancer	699	9	9
Credit Card Default [14]	credit	29623	14	14
MAGIC Gamma Telescope [14]	magic	19020	10	10
Spambase [14]	spambase	4600	57	57

Table 2. Dataset characteristics, N instances and n features before and after one-hot encoding.

Metrics. Counterfactual examples are evaluated on t, the average time in seconds to explain a sample; the average explanation sparsity s, i.e. the number of features changed; proximity δ_1 and δ_2, the average L1 and L2 norm distances; and % validity, the percent of returned explanations which are correctly counterfactual [35]. We report the average of these metrics for each batch and repeat the training and explanation process 10 times, averaging the results.

7.2 Counterfactual Example Quality - Random Forest

We evaluate FACET against state-of-the-art explanation methods by comparing the nearest point in FACET's counterfactual regions to counterfactual examples generated by existing methods. As these methods suffer from slow runtimes we select an ensemble with $T = 10$, $D_{max} = 5$ (RF). The results are shown in Tab. 3 with distances δ_1, δ_2 reported as multiples of FACET's. As MACE and RFOCSE struggle to generate explanations in tractable time, we apply a 300 second max explanation time per instance, after which explanation is terminated. We report the percentage of valid explanations found in that time. For cancer and spambase RFOCSE had near 0% validity within the time limit, so we rerun it for one test iteration with unlimited time. These results are marked with *.

Dataset	**ADULT**					**CANCER***					**CREDIT**					**MAGIC**					**SPAMBASE***				
	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$
FACET	0.108	1.68	1.00	1.00	100	0.094	4.33	1.00	1.00	100	0.128	4.54	1.00	1.00	100	0.003	2.99	1.00	1.00	100	0.201	4.47	1.00	1.00	100
MACE	107.4	5.69	70.5	81.4	68.5	2.222	30.00	21.2	8.88	100	190.2	10.9	9.07	5.77	38.0	2.472	10.0	12.4	7.61	100	19.91	57.0	1261	371	100
OCEAN	0.586	2.44	3.26	2.97	100	0.681	11.41	1.43	1.02	100	1.721	6.15	2.13	1.97	100	1.790	6.30	1.45	1.05	100	0.464	7.50	0.69	0.64	100
RFOCSE	56.59	1.40	0.54	0.58	90.5	1936	4.65	1.29	1.07	100	202.5	2.44	0.16	0.21	35.0	147.0	2.42	0.87	1.04	95.0	2208	3.30	4.25	4.51	100
AFT	0.003	1.46	0.76	0.80	96.0	0.002	1.86	0.92	1.26	37.0	0.002	2.42	1.12	1.22	85.5	0.004	1.65	0.99	1.24	99.0	0.003	1.98	2.00	3.03	76.0

Table 3. Comparison to state-of-the art counterfactual example generation techniques for random forest ($T = 10$, $D_{max} = 5$) in terms time t, sparsity s, L1-Norm δ_1, L2-Norm δ_2, and validity %.

In Tab. 3, we find MACE's local permutation approach generates explanations with very poor distance. Its best case is 6x FACET's δ_2 distance (credit) and as bad as >1200x FACET's δ_1 (spambase). RFOCSE's explanation distance is fair when it finds an explanation, but has intractable runtime for some datasets (spambase, cancer). This likely results from the exponential complexity of fusing the ensemble into one tree. It also frequently fails to find any explanation (failing up to 65% of the time for credit). RFOCE's small distance for credit and adult is due to it preferentially failing to explain instances that lie further from the decision boundary. Thus if methods have % < 100, the δ values are not directly comparable, but are included for completeness. Similarly, AFT quickly returns explanations of adequate distance, but produces many invalid counterfactuals (up to 63% for cancer), rendering it and RFOCSE unfit for practical use. FACET consistently outperforms the state-of-the-art in explanation time, distance, and validity with only OCEAN being competitive. We thus compare to OCEAN in more depth in Sec. 7.7.

7.3 Counterfactual Example Quality - Gradient Boosting

Tab. 4 examines FACET's ability to generalize across different ensemble types including Random Forest (abbreviated FCT-RF), and Scikit's Gradient Boosting Classifier [29] with our Complete Intersection (FCT-GB1), and Relaxed Intersection (FCT-GB2) region generation strategies (Sec. 4.5). Ensembles here are larger than in Tab. 3 as we selected gradient boosting's default of $T = 100$, $D_{max} = 3$ for all three cases. Results for δ_1, δ_2 are reported as multiples of FCT-RF.

Dataset	ADULT					CANCER					CREDIT					MAGIC					SPAMBASE				
	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$	$t\downarrow$	$\delta_0\downarrow$	$\delta_1\downarrow$	$\delta_2\downarrow$	$\%\uparrow$
FC-RF	0.233	3.92	1.00	1.00	100.0	0.110	11.86	1.00	1.00	100.0	0.129	6.31	1.00	1.00	100.0	0.007	4.24	1.00	1.00	100.0	0.036	11.04	1.00	1.00	100.0
FCT-GB1	0.368	2.60	0.11	0.15	100.0	0.059	8.08	0.50	0.60	100.0	0.129	4.15	0.13	0.14	100.0	0.006	4.86	0.78	0.68	100.0	0.056	7.75	0.51	0.61	100.0
FCT-GB2	0.342	2.50	0.08	0.12	100.0	0.048	7.54	0.47	0.58	100.0	0.128	4.10	0.12	0.14	100.0	0.006	4.83	0.78	0.68	100.0	0.056	7.66	0.49	0.59	100.0

Table 4. Evaluation of FACET explanation approaches on gradient boosting and random forest ensembles ($T = 100$, $D_{max} = 3$) in terms time t, sparsity s, L1-Norm δ_1, L2-Norm δ_2, and validity %.

Here we find that FACET generalizes well to other ensemble types with both approaches for the gradient boosting ensemble having similar explanation time and lower explanation distances than the random forest case, with the complete intersection approach being slightly more distant. Thus, if seeking models for low cost explanation, designers may wish to consider using gradient boosting ensembles in place of random forests when appropriate.

7.4 Explanation Robustness

Next, we compare the robustness of counterfactuals provided by FACET's regions to the robustness of counterfactual examples from state-of-the-art methods. To do this, we explain an instance with each method and perturb the resulting counterfactual example 100 times along uniformly random sampled vectors. We repeat this for varying perturbation sizes and record the ratio of perturbed points which are valid explanations. We generate new perturbation vectors for each instance. Counterfactual examples for FACET are selected to be centered in the returned region. The results of this are shown in Fig. 9 averaged over 10 randomly selected instances (RF, $T = 10$, $D_{max} = 5$).

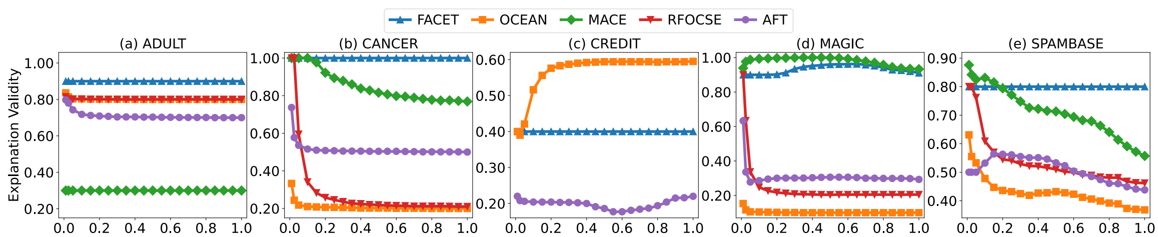

Fig. 9. Evaluation of nearest explanation robustness to varying random perturbation size (percent of space).

Here we find that FACET's explanations are substantially more robust than state-of-the-art methods as seen in the much higher validity of the perturbed explanation set. Notably, only MACE has comparable robustness to FACET. This is a result of MACE's poor explanation distance which causes its counterfactuals to be far from the nearest decision boundary. The same is true for OCEAN on the credit dataset. Values for MACE and RFOCSE could not be generated for credit due to their high failure rate on this data. All methods showed lower validity on credit compared to other datasets, indicating that the ensemble has learned small segmented partitions. Overall, FACET obtains higher robustness on average than its competitors without sacrificing explanation distance.

7.5 FACET Explanation Personalization

To demonstrate that FACET's analytics can be personalized to meet user needs in interactive time, we sample instances as in the batch methodology above (RF, $T = 100$, $D_{max} = None$). For each batch, we query FACET to generate explanations for varying values of k and under varying sets of feature-value and minimum robustness constraints.

Min Robustness. Fig. 10a-c show the effect of user provided minimum robustness constraints on FACET's explanation. We find that a user can easily obtain an explanation robust to perturbations up to 10% of the feature space with negligible impact on explanation distance (10b), runtime (10a) and sparsity (10c). This demonstrates FACET's ability to provide users with the power to obtain their desired outcome on even difficult datasets such as credit.

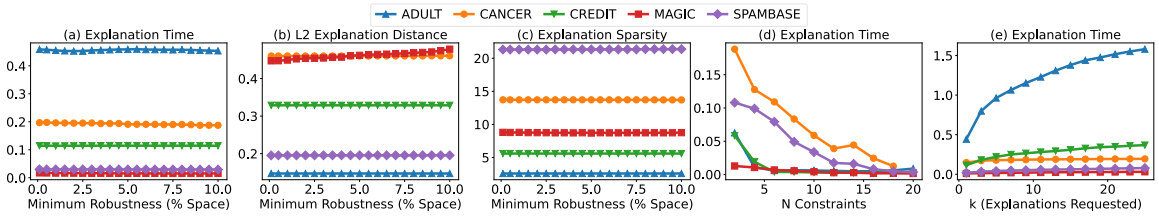

Fig. 10. Evaluation of FACET's explanation analytics with diverse query workloads.

Effect of Constraints. To evaluate user constraints, we vary the number of constraints applied, N_c. We randomly select features to constrain and the applied min and max allowable values. The particular constraint values are selected from a random uniform distribution on $[0.1, 0.49]$ for lower bounds and $[0.51, 0.99]$ for upper bounds for numeric features. This ensures the constrained region has a non-zero volume. Discrete features were randomly constrained to one valid value; while binary features were not constrained due to the limited options. The result of this is shown in Fig. 10d. Here we observe that as N_c increases, the time needed to execute FACET's explanation search decreases. This is because the application of constraints by FACET's constraints hyperrectangle is highly efficient when applied via COREX and the more constraints applied, the smaller the search region. This is beneficial for users as COREX enables their subsequent explanation refinement queries to execute increasingly quickly, allowing for easy fine-tuning.

Effect of k. Fig. 10e shows FACET's query time for different values of k, the number of requested explanations per instance. Here we find that FACET can service users with many counterfactual regions with little additional latency compared to $k = 1$, empowering users to compare many explanations using one query. The higher runtime for adult is a result of enforcing its many one-hot encoded features, a process which is not yet optimized in our code.

7.6 FACET's COREX Index Evaluation

To demonstrate FACET's ability to efficiently generate and index a large number of candidate counterfactual regions with COREX, we evaluate FACET's performance while varying the number of regions generated from $N_r = 100$ to 100,000. We then explain samples according to the batch methodology above. Here we use FACET's kNN query (Sec. 6.1) with $k = 1, m = 4$. (RF, $T = 100$, $D_{max} = None$). Region seed points are generated via the augmentation strategy from Eq. 3.

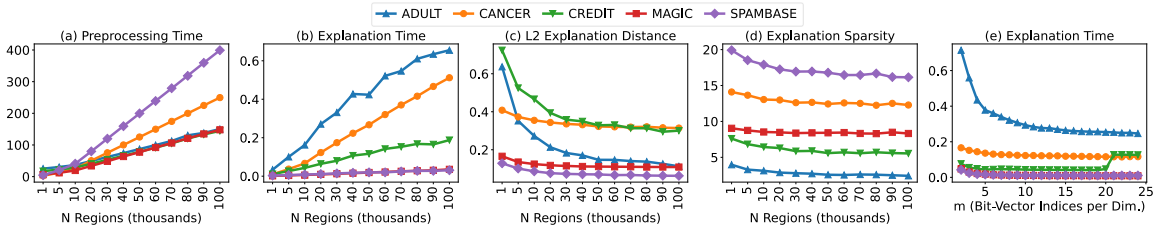

Fig. 11. Evaluation of FACET's explanation analytics using COREX, our counterfactual region index.

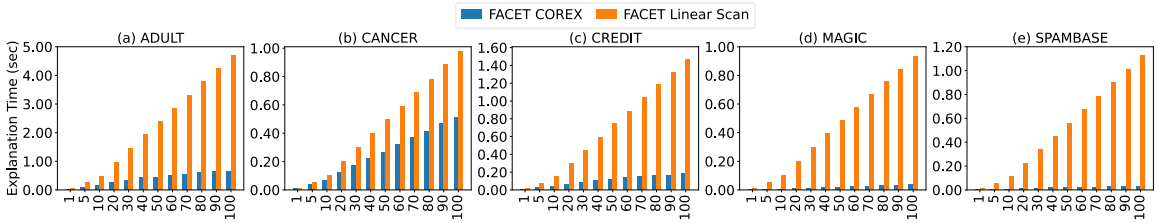

Fig. 12. Evaluation of query response time with and without COREX, FACET's bit-vector based counterfactual region explanation index. Varying N_r, the number of indexed counterfactual regions.

Index Construction. Fig. 11a shows the total generation and indexing time for counterfactual regions during preprocessing. Here we see that FACET can generate and index a very large number of regions in just a few minutes, even on high dimensional data.
Region Distance. Fig. 11c shows the average explanation distance resulting from searching these regions and Fig. 11d the average sparsity. We observe that FACET's region generation algorithm (Sec. 4.4) achieves a consistent explanation distance while selecting only a small subset of the exponentially many possible regions. Based on these results, we select $N_r = 20{,}000$ for cancer, magic, and spambase, and $N_r = 50{,}000$ for adult and credit.
Region Searchtime. Fig. 12 compares the average time needed to process a kNN query using COREX as compared to a linear scan of the regions. Here we observe that COREX greatly accelerates query execution and enables FACET to maintain real-time interaction even while searching up to 100,000 counterfactual regions.
Effect of Index Granularity. As the number of bit-vector intervals generated per feature (m) affects explanation time we analyze this parameter in Fig. 11e. As larger values of m result in a larger index, we progressively increase the value of m until addition of intervals does not benefit the explanation time. We select a final value of $m = 16$ for all datasets which is used in all experiments.

7.7 Model Scalability Evaluation

Next, we evaluate the impact of varying AI model complexities to demonstrate that FACET is capable of generating relevant counterfactuals in interactive time for even complex models. We generate explanations for ensembles of size $T = 10 \ldots 500$ (RF, $D_{max} = 5$). Given the results of Sec. 7.2, we select OCEAN as the closest competitor to FACET due to the poor explanation distance of MACE, the frequent invalid explanations of AFT, and the extremely poor runtime of RFOCSE.

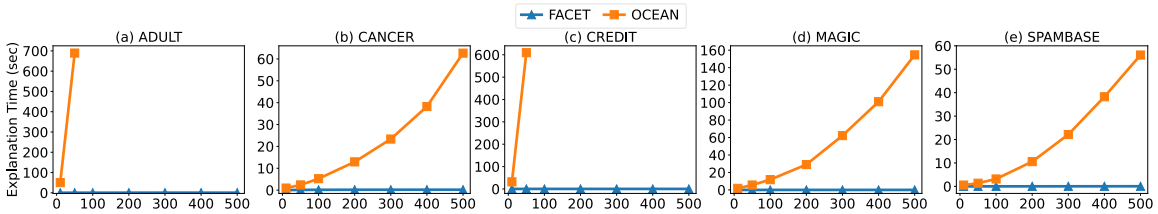

Fig. 13. Explanation time as a function of model complexity. Varying number of trees T.

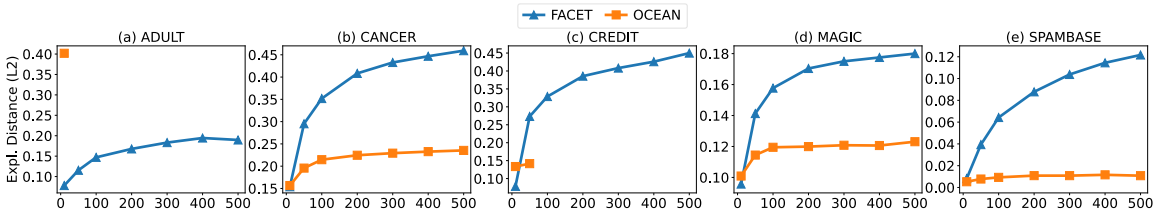

Fig. 14. Explanation distance as a function of model complexity. Varying number of trees T.

Fig. 13 compares the average explanation time for FACET and OCEAN. Here we find that OCEAN's runtime is exponential in the ensemble size, making OCEAN unsuitable for real time explanation and precluding OCEAN from being used as part of an interactive process. Indeed for adult and credit, the runtime grows so rapidly that for $T > 50$ OCEAN hits its internal runtime limit of 15 minutes for every instance and produces no valid explanations. Comparatively, FACET's index-aware queries leveraging COREX can generate explanation in interactive time regardless of ensemble complexity. Further, Fig. 14 compares average δ_2 distance for both methods. Here we observe that FACET's explanations are comparable to those generated by OCEAN in terms of explanation distance, while providing the added guarantee of explanation robustness and the flexibility of personalization via real-time explanation analytics.

8 PRELIMINARY USER STUDY

To examine FACET's explanations in a human context, we conduct a survey of 15 participants examining the between-groups effect of explanation type (Region vs Example) on task performance.
Explanation Generation and Display. For the study, we use a Kaggle dataset [23] of loan approval information and train a random forest model to predict *Loan Approval* (binary reject/accept) from *Applicant Income*, *Coapplicant Income*, *Loan Amount*, and *Loan Term*. We then generate Counterfactual Region explanations for a set of rejected instances using FACET. From each region we also generate a traditional Counterfactual Example matching the state-of-the-art. We present the explanations to users as a table of feature-values. Region explanations were presented as ranges along each feature, e.g., $\$100 - \$500$, and Example explanations as its specific value, e.g., \$300.
Study Design. In the survey, each respondent was introduced to the loan scenario, randomly assigned to a group (7 for Examples, 8 for Regions), trained to read the explanations, and then asked two evaluation questions. Attention checks were used to ensure respondents read the given material and results discarded from those who failed a check, this left 6 respondents per group.
Evaluation of Understanding. To assess the effect of explanation type on understanding, respondents were presented with a Region or an Example explanation for a rejected instance and a new similar instance of unknown outcome. They were then asked to predict the model's classification for the new instance. We asked two such questions and scored respondents compared to the true model behavior for an *accuracy score* ranging 0-2. On this score, respondents on average made more accurate predictions of the model's behavior using Regions (2.0) compared to Examples (1.83).
Overall Takeaway. These results suggest that FACET's novel Counterfactual Region explanations may improve a user's understanding of the model more than state-of-the-art Counterfactual Example explanations. While these initial results are promising, an in-depth evaluation study examining a larger cohort across a wider variety of metrics is needed for a full assessment.

9 RELATED WORK

Non-counterfactual Explanation. Explanation for tree ensembles traditionally identified feature importance, namely, at an ensemble level using metrics like Gini Index [7] and an instance level via model-opaque approximation techniques [32, 41]. However, these techniques cannot produce counterfactuals. Approaches exist for fusing ensembles to single interpretable trees [10, 42], but require approximation and are unfit for counterfactual explanation.

Counterfactual Explanation. A set of model-opaque methods generate counterfactual examples via gradient access [11, 28, 35] but require differentiable models. Later works extend to non-differentiable models using local approximation from predicted class probabilities [12, 18, 40]. Counterfactual generation has also been considered via genetic algorithms [19, 43, 44] similar to adversarial attacks. Of model transparent approaches for tree ensembles, initial work AFT [45] generated explanations via exhaustive enumeration of altered instances and often failed to generate valid counterfactual examples (Tab. 3). A set of methods [9, 25, 37] convert ensemble explanation to an integer programming task solved by optimization. These methods can locate distance-minimal counterfactual examples, but lack suitable personalization and as shown in Fig. 13 have a runtime exponential in the ensemble size. RFOCSE [16] attempts to generate sets of counterfactual examples by merging the entire ensemble to a tree. Yet as shown in Table 3, it also reaches intractable runtimes for some datasets on even small ensembles. Recent work [31] approximates tree ensembles via differentiable neural networks and proceeds similar to gradient-based model-opaque methods. All but [18] generate counterfactual examples and so lack the critical component of explanation robustness. While [18] provides approximate counterfactual rule sets, the approximation process renders it incapable of definitively specifying the class of points in the rule sets. Recent work [15] attempts

to find counterfactuals which hold after model retraining, a different consideration of robustness than FACET. This approach cannot provide the interactive user understanding of FACET's analytics nor guarantee robustness to perturbations as provided by FACET's regions.

Indexing Techniques. Spatial nearest neighbor search has largely been driven by GIS systems. There are a plethora of spatial indices for kNN search: Quad Trees [21], RTree-based methods [4, 5, 24], and others. However, most are designed for lower dimensional spaces and perform poorly at high dimensions [5]. On the other hand, kNN methods for high dimensional data tend to perform approximate nearest neighbors search via hashing [20, 22], partitioning [6, 36], and graph-based [13, 33] approaches. Unfortunately, these works tend to operate on point data rather than spatial data. As explanation region analytics necessitates high-dimensional spatial NN search, an understudied field, with searches subject to personalization constraints, existing standard indices are insufficient for use in FACET and we develop a custom-tailored solution.

10 DISCUSSION AND FUTURE WORK

FACET opens a new area of research in explanation analytics, namely in developing paradigms for human-in-the-loop exploration of the explanation space rather than simply generating static explanations. After all, explanations aim to help humans understand, gain trust, and affect change, all processes which require human input. As such new modalities for communicating, exploring, and interacting with explanations using visual and/or natural language models hold promise. On the database side, FACET pushes the state-of-art in new classes of queries, query processing, indexing, and optimization for realizing these novel explanation paradigms. Further work could also study the generation of FACET's region explanations for other model types; from traditional techniques to more complex neural networks, and in new ways such as dynamic on-the-fly region generation. This would bring FACET's concept of *robust user-centric* explanation to a wider array of systems. While FACET's core counterfactual region abstraction, representation, indexing, and analytic components may remain applicable, new research is needed into generating counterfactual regions in new types of models and to explore the possibilities of explanation indexing.

11 CONCLUSION

We propose FACET, the first system for robust personalized explanation of decisions made by AI models in general and tree ensemble-based models in particular. FACET's explanation analytics, backed by our custom bit-vector based explanation index COREX, empower users to identify actionable explanations through iterative human-in-the-loop explanation refinement. Our novel region explanations are simple to understand and provide guarantees of robustness critical for the real-world application of explanations. Our experiments demonstrate that FACET generates these robust explanations an order of magnitude faster than state-of-the-art XAI systems while maintaining similar explanation quality and leading to improved user understanding.

ACKNOWLEDGMENTS

This research was supported in part by NSF under grants IIS-1910880, CSSI-2103832, CNS-1852498, NRT-HDR-2021871 and the U.S. Department of Education under grant P200A180088. Thanks also to the members of the DAISY research group for their input on this research.

REFERENCES

[1] Equal Credit Opportunities Act. 1974. Public Law, 15 C.F.R § 1691, Regulation B 12 C.F.R. § 1002.

[2] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access* 6 (2018), 52138–52160.

[3] Solon Barocas, Andrew D. Selbst, and Manish Raghavan. 2020. The Hidden Assumptions behind Counterfactual Explanations and Principal Reasons. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 80–89. https://doi.org/10.1145/3351095.3372830

[4] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. 1990. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data* (Atlantic City, New Jersey, USA) *(SIGMOD '90)*. Association for Computing Machinery, New York, NY, USA, 322–331. https://doi.org/10.1145/93597.98741

[5] Stefan Berchtold, Daniel A. Keim, and Hans-Peter Kriegel. 1996. The X-Tree : An Index Structure for High-Dimensional Data. In *Proceedings of the Twenty-second International Conference on Very Large Data-Bases ; Mumbai (Bombay), India 3 - 6 September, 1996*, T. M. Vijayaraman (Ed.). Morgan Kaufmann, San Francisco, 28–39.

[6] Erik Bernhardsson. 2005. Spotify/Annoy: Approximate nearest neighbors in c++/python optimized for memory usage and loading/saving to disk. https://github.com/spotify/annoy.

[7] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[8] Leo Breiman. 2001. Statistical modeling: The two cultures. *Statist. Sci.* 16, 3 (2001), 199–231.

[9] Zhicheng Cui, Wenlin Chen, Yujie He, and Yixin Chen. 2015. Optimal Action Extraction for Random Forests and Boosted Trees. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, NSW, Australia) *(KDD '15)*. Association for Computing Machinery, New York, NY, USA, 179–188. https://doi.org/10.1145/2783258.2783281

[10] Houtao Deng. 2019. Interpreting tree ensembles with intrees. *International Journal of Data Science and Analytics* 7, 4 (2019), 277–287.

[11] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc., Montreal, QC, Canada, 12. https://proceedings.neurips.cc/paper/2018/file/c5ff2543b53f4cc0ad3819a36752467b-Paper.pdf

[12] Amit Dhurandhar, Tejaswini Pedapati, Avinash Balakrishnan, Pin-Yu Chen, Karthikeyan Shanmugam, and Ruchir Puri. 2019. Model Agnostic Contrastive Explanations for Structured Data. *ArXiv preprint* abs/1906.00117 (2019), 12 pages. https://arxiv.org/abs/1906.00117

[13] Wei Dong. 2014. AAALGO/kgraph: A library for K-Nearest Neighbor Search. https://github.com/aaalgo/kgraph.

[14] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[15] Sanghamitra Dutta, Jason Long, Saumitra Mishra, Cecilia Tilli, and Daniele Magazzeni. 2022. Robust Counterfactual Explanations for Tree-Based Ensembles. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, Baltimore, MD, USA, 5742–5756. https://proceedings.mlr.press/v162/dutta22a.html

[16] Rubén R. Fernández, Isaac Martín de Diego, Víctor Aceña, Alberto Fernández-Isabel, and Javier M. Moguerza. 2020. Random forest explainability using counterfactual sets. *Information Fusion* 63 (2020), 196–207. https://doi.org/10.1016/j.inffus.2020.07.001

[17] Sainyam Galhotra, Romila Pradhan, and Babak Salimi. 2021. Explaining Black-Box Algorithms Using Probabilistic Contrastive Counterfactuals. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 577–590. https://doi.org/10.1145/3448016.3458455

[18] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local Rule-Based Explanations of Black Box Decision Systems. *CoRR* abs/1805.10820 (2018), 10 pages. arXiv:1805.10820 http://arxiv.org/abs/1805.10820

[19] Masoud Hashemi and Ali Fathi. 2020. PermuteAttack: Counterfactual Explanation of Machine Learning Credit Scorecards. https://doi.org/10.48550/ARXIV.2008.10138

[20] Qiang Huang, Jianlin Feng, Yikai Zhang, Qiong Fang, and Wilfred Ng. 2015. Query-aware locality-sensitive hashing for approximate nearest neighbor search. *Proceedings of the VLDB Endowment* 9, 1 (2015), 1–12.

[21] Gregory M. Hunter and Kenneth Steiglitz. 1979. Operations on Images Using Quad Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1, 2 (1979), 145–153. https://doi.org/10.1109/TPAMI.1979.4766900

[22] Qing-Yuan Jiang and Wu-Jun Li. 2015. Scalable graph hashing with feature transformation. In *Twenty-fourth International Joint Conference on Artificial Intelligence*. AAAI, Buenos Aires, Argentina, 2248–2254.
[23] Kaggle. 2008. Loan Predication. https://www.kaggle.com/datasets/ninzaami/loan-predication,.
[24] Ibrahim Kamel and Christos Faloutsos. 1993. *Hilbert R-tree: An improved R-tree using fractals.* Technical Report. University of Maryland, Institute for Systems Research.
[25] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. 2020. DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, Christian Bessiere (Ed.). International Joint Conferences on Artificial Intelligence Organization, Yokohama, Kanto, Japan, 2855–2862. https://doi.org/10.24963/ijcai.2020/395 Main track.
[26] Amir-Hossein Karimi, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera. 2022. A Survey of Algorithmic Recourse: Contrastive Explanations and Consequential Recommendations. *ACM Comput. Surv.* 55, 5, Article 95 (dec 2022), 29 pages. https://doi.org/10.1145/3527848
[27] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. 2021. Algorithmic Recourse: From Counterfactual Explanations to Interventions. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 353–362. https://doi.org/10.1145/3442188.3445899
[28] Thai Le, Suhang Wang, and Dongwon Lee. 2020. GRACE: Generating Concise and Informative Contrastive Sample to Explain Neural Network Model's Prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 238–248. https://doi.org/10.1145/3394486.3403066
[29] Scikit Learn. 2023. Gradient Boosting Classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html,.
[30] Zhicheng Liu and Jeffrey Heer. 2014. The Effects of Interactive Latency on Exploratory Visual Analysis. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 2122–2131. https://doi.org/10.1109/TVCG.2014.2346452
[31] Ana Lucic, Harrie Oosterhuis, Hinda Haned, and Maarten de Rijke. 2022. FOCUS: Flexible Optimizable Counterfactual Explanations for Tree Ensembles. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 5 (Jun. 2022), 5313–5322. https://doi.org/10.1609/aaai.v36i5.20468
[32] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, CA, USA. https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf
[33] Yury Malkov, Alexander Ponomarenko, Andrey Logvinov, and Vladimir Krylov. 2014. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems* 45 (2014), 61–68.
[34] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1–38. https://doi.org/10.1016/j.artint.2018.07.007
[35] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 607–617. https://doi.org/10.1145/3351095.3372850
[36] Marius Muja and David G Lowe. 2014. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 11 (2014), 2227–2240.
[37] Axel Parmentier and Thibaut Vidal. 2021. Optimal Counterfactual Explanations in Tree Ensembles. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, Virtual, 8422–8431. https://proceedings.mlr.press/v139/parmentier21a.html
[38] Article 29 Data Protection Working Party. 2016. Guidelines on Automated Individual Decision-Making and Profiling for the Purposes of Regulation 2016/679. https://ec.europa.eu/newsroom/article29/items/612053
[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[40] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. FACE: Feasible and Actionable Counterfactual Explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (New York, NY, USA) *(AIES '20)*. Association for Computing Machinery, New York, NY, USA, 344–350. https://doi.org/10.1145/3375627.3375850
[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[42] Omer Sagi and Lior Rokach. 2020. Explainable decision forest: Transforming a decision forest into an interpretable tree. *Information Fusion* 61 (2020), 124–138.

[43] Maximilian Schleich, Zixuan Geng, Yihong Zhang, and Dan Suciu. 2021. GeCo: Quality Counterfactual Explanations in Real Time. *Proc. VLDB Endow.* 14, 9 (oct 2021), 1681–1693. https://doi.org/10.14778/3461535.3461555

[44] Shubham Sharma, Jette Henderson, and Joydeep Ghosh. 2020. CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-Box Models. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (New York, NY, USA) *(AIES '20)*. Association for Computing Machinery, New York, NY, USA, 166–172. https://doi.org/10.1145/3375627.3375812

[45] Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. 2017. Interpretable Predictions of Tree-Based Ensembles via Actionable Feature Tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 465–474. https://doi.org/10.1145/3097983.3098039

[46] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E. Hines, John P. Dickerson, and Chirag Shah. 2020. Counterfactual Explanations and Algorithmic Recourses for Machine Learning: A Review. https://doi.org/10.48550/ARXIV.2010.10596

[47] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *Harvard journal of law & technology* 31, 2 (2017), 841–.

Received April 2023; revised July 2023; accepted August 2023