Structural Drift Repair in Decision Trees via Bayesian Model-Based Diagnosis

Yoav Zelinger[1] **Meir Kalech**[1]

[1]The Stein Faculty of Computer Science and Information, Ben-Gurion University of the Negev, Be'er Sheva, Israel

Abstract

Decision trees are widely used in machine learning due to their interpretability. However, when the underlying data distribution changes—a phenomenon known as *concept drift*—their performance can degrade significantly. Retraining the model from scratch discards the learned structure and offers no insight into which nodes were actually affected. Prior work, APPETITE, used spectrum-based fault localization to identify and modify a single faulty node, but cannot handle drift that affects multiple nodes simultaneously or that demands structural changes to the tree. We propose BTAD (*Bayesian Tree Adaptation for Drift*), which formalizes the Decision Tree Concept Drift Problem as a Model-Based Diagnosis task and employs BARINEL, a Bayesian algorithm that ranks multi-node fault hypotheses across the entire tree. For repair, BTAD applies Subtree Re-fitting, replacing affected subtrees with ones fitted to the post-drift distribution. BTAD relies on lightweight node-level statistics and requires no access to the original training data. Experiments show that BTAD outperforms APPETITE in both recovery and localization, achieving accuracy comparable to full retraining—which provides no drift insight and requires the original data.

1 INTRODUCTION

Decision trees are among the most interpretable machine learning models for tabular data (Gao et al. [2021], Lee et al. [2022]), as predictions follow explicit splitting constraints that experts can trace. However, real-world distributions may shift over time—a phenomenon known as *concept drift* (Gepperth and Hammer [2016])—degrading predictive performance. The standard mitigation is to retrain from scratch (Lu et al. [2018]), yet this provides no insight into *which* nodes were affected, and may be impractical when post-drift data is scarce or the original data is unavailable.

To address explainable drift repair, Almog and Kalech [2023] presented APPETITE, a spectrum-based fault localization (SFL) approach, which pinpoints a faulty tree node that can explain the drift, and repairs it. However, APPETITE is limited to a *single* faulty node and narrow node-level parameter corrections, which may be inadequate when drift impacts multiple tree regions or demands structural changes.

We propose BTAD (*Bayesian Tree Adaptation for Drift*), a diagnosis-and-repair framework for concept drift in decision trees that makes two contributions. First, we formalize the Decision Tree Concept Drift Problem (DTCDP) as an Model-Based Diagnosis (MBD) problem (de Kleer and Williams [1987]), enabling the use of BARINEL (Abreu et al. [2009b]), a Bayesian MBD algorithm that produces ranked *multi-node* diagnoses covering all drift-affected regions. Second, we introduce Subtree Re-fitting, a repair strategy that replaces entire subtrees at diagnosed nodes with newly induced ones, enabling structural modifications beyond local parameter adjustments. By repairing only the affected nodes, BTAD preserves healthy regions and identifies *where* in the tree drift manifested. Crucially, BTAD requires no access to the original training data: during training, each node stores compact statistics from which synthetic samples are generated to augment the post-drift data, improving the robustness of the replacement subtrees. This enables a *pre-trained tree distribution* paradigm, in which a model can be deployed and subsequently adapted in place without accessing the raw data—highly appealing under data privacy regulations (e.g., GDPR), edge deployment and federated settings where raw data cannot be retained.

We evaluate BTAD on benchmarks from the TRIO collection (Cohen-Shapira and Rokach [2021]) under feature-shift and distribution-edge drift scenarios. Our results show that each contribution independently improves performance: replacing the local repair operator with Subtree Re-fitting yields consistent gains, particularly under broad distribu-

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:7992–8005.

tional shifts. In addition, BARINEL presents more precise fault localization than SFL, underscoring its ability to pinpoint drift-affected nodes. Moreover, BTAD shows accuracy recovery that nearly matches NEWALL, a baseline exhibiting two key limitations: (1) no diagnostic insight into which parts of the model were affected by the drift; (2) It requires continued access to the original training set, which may be unavailable due to storage, privacy, or regulatory constraints.

2 RELATED WORK

Concept Drift. Methods for handling concept drift (Gepperth and Hammer [2016]) are generally divided into active approaches, which detect drift and then retrain (Singhal et al. [2020]), and passive approaches, which adapt the model continuously via incremental or online learning (Li et al. [2020], Wang et al. [2003], Scholz and Klinkenberg [2005], Cabrera et al. [2021]). Tree-based stream algorithms such as the Hoeffding Tree (Domingos and Hulten [2000]) and the Mondrian Tree (Roy and Teh [2008]) assume stationary distributions; several extensions incorporate drift handling through tree replacement or ensemble-level mechanisms (Hoeglinger and Pears [2007], Rad and Haeri [2019], Khannouz and Glatard [2024]), most notably the Adaptive Random Forest (Gomes et al. [2017]). Li et al. [2024] further refine this by classifying the drift type before selectively retraining. All these methods operate at the model level—discarding and rebuilding entire trees—without identifying which internal components caused the degradation, a gap also noted in a recent survey on explainability in concept drift Pelosi et al. [2025]. Our work instead diagnoses individual nodes, enabling targeted, structure-preserving repair.

Formal Explainability. Many explainability techniques lack formal guarantees (Marques-Silva [2023]); formal explainability (Marques-Silva and Ignatiev [2022]) addresses this through rigorous, provably minimal explanations developed for both model-agnostic (Cooper and Marques-Silva [2023], Huang et al. [2022]) and model-specific settings, including neural networks (Bassan and Katz [2023], Jiang et al. [2023]) and tree-based classifiers (Ignatiev et al. [2022], Huang and Marques-Silva [2023], Carbonnel et al. [2026], Izza et al. [2025], Bounia and Agoun [2025]). Marques-Silva [2025], Marques-Silva and Huang [2024] survey logic-based approaches that formalize sufficient and contrastive explanations via minimal feature sets, contrasting them with heuristic methods like SHAP (Lundberg and Lee [2017]) and LIME (Ribeiro et al. [2016]). These works focus on individual predictions; our goal is complementary—we localize *nodes implicated in the degradation* under concept drift by diagnosing the tree nodes whose constraints conflict with the shifted distribution.

3 PROBLEM DEFINITION

In this section, we formalize the *Decision Tree Concept Drift Problem* (DTCDP). We begin with the notion of a **classification model**:

Definition 1 (Classification Model)**.** *Let X be a feature space and Y a class space, and let $F \subseteq X$ and $C \subseteq Y$ be sets of features and classes, respectively. Given $S_D = \{(x^j, y^j)\}_{j=1}^m \subseteq X \times C$ a dataset of m samples drawn from distribution D, a classification model is a learned function $\phi : X \to C$ that maps instances to a class label.*

A well-known family of classification models is decision trees. A widely used variant — and the focus of this work — is the **binary decision tree**.

Definition 2 (Binary Decision Tree)**.** *A binary decision tree $\mathcal{T} = (V_\mathcal{T}, E_\mathcal{T})$ is a classification model structured as a rooted, directed acyclic graph. $V_\mathcal{T}$ is partitioned into* internal *(non-terminal) nodes $V_{\mathcal{T}NT}$ and* leaf *(terminal) nodes $V_{\mathcal{T}T}$. Each leaf carries a class $y \in C$. Each internal node $n_i \in V_{\mathcal{T}NT}$ carries a splitting constraint $cnst_{n_i}$ defined over a feature $f_j \in F$, denoted $\langle f_j, cnst_{n_i} \rangle$, and has exactly two children: the right child is reached when $cnst_{n_i}$ is satisfied, and the left child when it is violated. Classification proceeds by traversing from the root along the edges dictated by successive constraints until a leaf is reached; its label is the predicted class. The ordered sequence of nodes visited during this traversal is called a classification path.*

The quality of a decision tree $\mathcal{T}$ on a dataset S is measured by a performance metric $\rho(\mathcal{T}, S) \in \mathbb{R}$ (e.g., accuracy, precision, or recall). When the data distribution shifts from D to a new distribution D' — a phenomenon known as *concept drift* — constraints of $\mathcal{T}$ may no longer reflect the true data structure, causing a measurable drop in ρ. We formalize this as follows:

Definition 3 (Decision Tree Concept Drift Problem)**.** *Let $\mathcal{T}$ be a binary decision tree trained on the sample set S_D drawn from a distribution D, and let $S_{D'}$ be a new sample set drawn from a shifted distribution D'. A Decision Tree Concept Drift Problem (*DTCDP*) is said to occur whenever $\rho(\mathcal{T}, S_D) - \rho(\mathcal{T}, S_{D'}) > \theta_\mathcal{T}$, with $\theta_\mathcal{T} > 0$ a significant degradation threshold.*

4 METHODOLOGY

We address DTCDP by adopting diAgnosis-based aPproach for Post-concEpt drifT decIsion Trees rEpair (APPETITE) suggested by Almog and Kalech [2023], which frames concept drift handling as a two-phase process: (1) **Diagnosis** — identifying $\Delta_\mathcal{T}$, the set of *faulty nodes* responsible for the degradation; (2) **Repair** — modifying $\mathcal{T}$ based on $\Delta_\mathcal{T}$ to realign its predictions with D'.

A key practical consideration for this approach is eliminating the need to access the original training set S_D at repair time. Instead, each phase relies only on compact, constant-memory statistics that are pre-computed during training and stored within the tree's nodes. We detail these stored quantities at each point of use.

We first review the methodology of APPETITE (Almog and Kalech [2023]) (Section 4.1); then we introduce our novel approach BTAD (Section 4.2).

4.1 BACKGROUND: APPETITE (Almog and Kalech [2023])

APPETITE was originally proposed to adapt a decision tree to concept drift. Its diagnosis step uses Spectrum-Based Fault Localization (SFL) to identify a single faulty node, and its repair step applies a local Node Adaptation operator to that node.

4.1.1 Spectrum-Based Diagnosis

Central to the diagnosis is a *spectrum-based representation*. We first introduce the auxiliary notions of *node path* and *misclassification*, each defined with respect to the decision tree $\mathcal{T}$.

Definition 4 (Node Path)**.** *Given a tree $\mathcal{T}$ and a node $n_i \in V_\mathcal{T}$, the **node path** $Path_\mathcal{T}(n_i)$ is the set of splitting constraints encountered along the unique path from the root to n_i. Let p_i be the parent of n_i, the node path of n_i is recursively defined as:*

$$Path_\mathcal{T}(n_i) = \begin{cases} \emptyset & \text{if } n_i \text{ is the root node} \\ Path_\mathcal{T}(p_i) \cup \{cnst_{p_i}\} & \text{if } n_i \text{ is a right child} \\ Path_\mathcal{T}(p_i) \cup \{\neg cnst_{p_i}\} & \text{if } n_i \text{ is a left child} \end{cases}$$

For a sample $s_j \in S_{D'}$, $Path_\mathcal{T}(n_i)(s_j)$ denotes the Boolean result of evaluating all constraints in $Path_\mathcal{T}(n_i)$ against s_j.

Definition 5 (Misclassification)**.** *Given a tree $\mathcal{T}$ and a sample s_j, the **misclassification** predicate $mis_\mathcal{T}(s_j)$ returns $True$ whenever the label assigned by $\mathcal{T}$ does not coincide with the ground-truth class y_j.*

Using those definitions, we can summarize the diagnostic signal of the tree in a binary matrix and an error vector.

Definition 6 (Node Spectrum)**.** *Given $S_{D'}$ — a sample set containing A samples and $\mathcal{T}$ — a decision tree with B nodes. The **node spectrum** $SPCT_\mathcal{T} \in \{0,1\}^{A \times B}$ is a binary matrix whose entry $spct_{\mathcal{T}_{j,i}}$ equals 1 when $Path_\mathcal{T}(n_i)(s_j) = True$, and 0 otherwise.*

Definition 7 (Error Vector)**.** *The **error vector** $E_\mathcal{T} \in \{0,1\}^A$ has its j-th component $e_{\mathcal{T}_j}$ set to 1 when $mis_\mathcal{T}(s_j) = True$, and to 0 otherwise.*

In effect, every row of $SPCT_\mathcal{T}$ encodes the nodes participated by a given sample, while the matching entry in $E_\mathcal{T}$ indicates whether that sample was incorrectly classified.

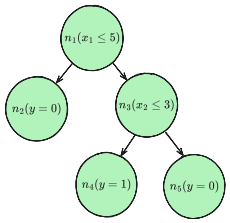


Figure 1: An example decision tree $\mathcal{T}$.

Sample	x_1	x_2	Class
s_1	3	4	0
s_2	4	2	1
s_3	7	1	0
s_4	8	3	0

Table 1: Sample dataset after a data drift.

	n_1	n_2	n_3	n_4	n_5	E
s_1	1	0	1	1	0	1
s_2	1	0	1	0	1	1
s_3	1	1	0	0	0	0
s_4	1	1	0	0	0	0

Table 2: Node spectrum $SPCT$ and error vector E for the tree in Figure 1 on the dataset in Table 1.

To illustrate, take sample $s_1 = ((3,4), y{=}0)$ from Table 1, classified using the tree $\mathcal{T}$ depicted in Figure 1. At the root n_1 the constraint $x_1 \leq 5$ holds, directing s_1 to the right child n_3. There, $x_2 \leq 3$ is not satisfied, so s_1 descends to the left child n_4, whose label is class 1. Because $y = 0 \neq 1$, we say that a misclassification occurs. The resulting spectrum row is $[1,0,1,1,0]$ with $e_1 = 1$ (only n_1, n_3, and n_4 participate). Repeating for all samples yields the full spectrum in Table 2.

APPETITE uses SFL (Abreu et al. [2009a]) to measure and rank how well each node's participation pattern in $SPCT_\mathcal{T}$ correlates with the observed misclassifications captured in $E_\mathcal{T}$. Intuitively, a node that has been participated mostly in misclassifications is a primary candidate for being faulty.

Let $\sigma(SPCT_{\mathcal{T}_{\cdot,i}}, E_\mathcal{T}) \in \mathbb{R}$ be a similarity coefficient (e.g., Ochiai (Abreu et al. [2007]) or Faith (Faith [1983])) evaluated between the i-th column of $SPCT_\mathcal{T}$ (denoted *participation column*) and the error vector $E_\mathcal{T}$. Formally, SFL rank is then defined as: $rank_{\text{SFL},\mathcal{T}}(n_i) = \sigma(SPCT_{\mathcal{T}_{\cdot,i}},\ E_\mathcal{T})$.

To illustrate, consider again Table 2. The participation column of n_3 coincides exactly with E, thereby attaining a perfect SFL score of 1.0 (e.g., under the Ochiai coefficient) — the highest among all nodes.

To refine the SFL ranking, APPETITE incorporates a statistical measure that quantifies how much each node's constraint has shifted between the pre-drift and post-drift distributions.

Definition 8 (Constraint Violation Rate)**.** *Given a tree $\mathcal{T}$, a node $n_i \in V_\mathcal{T}$, and a dataset S, the **constraint violation rate** describes the percentage of sam-*

ples that violate the node's constraint: $viol_{\mathcal{T}}(n_i, S) = \frac{\left|\{s_j | \neg cnst_{n_i}(s_j) \wedge Path_{\mathcal{T}}(n_i)(s_j),\, s_j \in S\}\right|}{|\{s_j | Path_{\mathcal{T}}(n_i)(s_j),\, s_j \in S\}|}$

Definition 9 (STAT Rank)**.** *The Statistical-Analysis rank of a node* $n_i \in V_{\mathcal{T}}$ *(***STAT *rank****) measures the shift in its constraint violation rate between* S_D *and* $S_{D'}$*:* $rank_{\text{STAT},\mathcal{T}}(n_i) = |viol_{\mathcal{T}}(n_i, S_D) - viol_{\mathcal{T}}(n_i, S_{D'})|$.

A high $rank_{\text{STAT},\mathcal{T}}$ reflects a substantial post-drift change in traversal behavior. Notably, $viol_{\mathcal{T}}(n_i, S_D)$ is computed once at training time as a scalar $\hat{v}_{n_i}$ inside each node, eliminating any dependence on S_D during diagnosis.

APPETITE produces a final *suspiciousness score* by taking the product of the SFL and STAT ranks for every node. The node that maximizes this product is returned as the single-element diagnosis $\Delta_{\mathcal{T}}$.

Returning to the running example, suppose the stored pre-drift violation rate for n_3 is $\hat{v}_{n_3} = 0.8$ while the post-drift rate is $viol_{\mathcal{T}}(n_3, S_{example}) = 0.5$; then $rank_{\text{STAT},\mathcal{T}}(n_3) = |0.8 - 0.5| = 0.3$. Combined with the SFL score computed earlier, the final suspiciousness score is $1.0 \times 0.3 = 0.3$.

4.1.2 Node Adaptation

Given the single faulty node identified by SFL, APPETITE applies a targeted modification — denoted as **Node Adaptation** — that adjusts the node to better align with the drifted distribution D'. The modification depends of the nature of the node. A terminal node has its class label reassigned to the majority class among the drifted samples that reach it; a non-terminal node with a binary or categorical feature has its constraint negated; and a non-terminal node with a numeric split has its threshold shifted by the change in the feature mean between S_D and $S_{D'}$ (the pre-drift mean is stored in the node).

4.2 BTAD: BAYESIAN TREE ADAPTATION FOR DRIFT

While APPETITE assumes and effectively handles concept drift that can be attributed to a single node, real-world distributional shifts may affect multiple regions of the tree simultaneously. Moreover, Node Adaptation operator applies narrow, parameter-level adjustments, which may be insufficient when the distributional shift demands deeper structural changes. To address those limitations, we propose BTAD (*Bayesian Tree Adaptation for Drift*), a new approach built upon the APPETITE framework that introduces two key contributions:

(1) Modeling the DTCDP in terms of Model-Based Diagnosis (MBD) problem, a well-known framework for automated diagnosis. This allows us to employ the BARINEL algorithm—an MBD-based Bayesian approach capable of producing multi-node diagnoses.

(2) A novel repair strategy based on Subtree Re-fitting, which replaces subtrees rather than tweaking parameters.

4.2.1 BTAD Diagnosis

Given DTCDP, our goal in the diagnosis phase is to find $\Delta_{\mathcal{T}}$, a set of faulty nodes which can explain the degradation of performance. We formulate this as an instance of consistency-based Model-Based Diagnosis (MBD) (de Kleer and Williams [1987], Stern et al. [2012]). In MBD, a diagnosis problem is characterized by the tuple $\langle SD, \mathcal{C}, \mathcal{O} \rangle$, representing the system description, the component set, and the observation, respectively. Such a problem emerges when *SD* becomes inconsistent with $\mathcal{O}$, under the assumption that every component in $\mathcal{C}$ functions correctly. A diagnosis is then a subset of components whose assumed faulty, suffices to restore consistency between the system description and the observation.

Let $\mathcal{T} = (V_{\mathcal{T}}, E_{\mathcal{T}})$ be a binary decision tree trained over samples S_D drawn from a distribution D. Each node of $\mathcal{T}$ is treated as a diagnosable component that may have been compromised by the distributional shift.

Definition 10 (Components)**.** *Given a decision tree* $\mathcal{T} = (V_{\mathcal{T}}, E_{\mathcal{T}})$*, the component set* $\mathcal{C} = V_{\mathcal{T}}$ *is defined as* $V_{\mathcal{T}}$*, the set of all nodes in* $\mathcal{T}$*.*

We associate with each node $n \in \mathcal{C}$ a Boolean healthy predicate $h(n)$. A node is deemed abnormal ($\neg h(n)$) with respect to $S_{D'}$ when concept drift has rendered its constraint or label inconsistent with the samples that reach it, thereby resulting in misclassifications.

Definition 11 (System Description)**.** *The system description* SD *of* $\mathcal{T}$ *comprises: (i) the tree topology (parent–child relationships and edge semantics), (ii) the splitting constraints at internal nodes, and the class assignments at leaves.*

Definition 12 (Observation)**.** *Let* ρ *be a performance metric and* $\theta_{\mathcal{T}}$ *be a degradation threshold. Given a post-drift sample set* $S_{D'}$ *drawn from* D'*, the observation* $\mathcal{O}$ *is the observed performance degradation:* $\rho(\mathcal{T}, S_D) - \rho(\mathcal{T}, S_{D'}) > \theta_{\mathcal{T}}$*.*

Definition 13 (Tree Diagnosis)**.** *Let* $\mathcal{SD}$ *be the system description of a tree* $\mathcal{T}$ *and let* $\mathcal{O}$ *be the observation of performance degradation. A set of nodes* $\Delta_{\mathcal{T}} \subseteq \mathcal{C}$ *is a tree diagnosis if assuming that* $\forall n \in \Delta_{\mathcal{T}}\ \neg h(n)$ *and* $\forall n \in \mathcal{C} \setminus \Delta_{\mathcal{T}}\ h(n)$*, makes* $\mathcal{SD}$ *consistent with* $\mathcal{O}$*.*

Consistency here is understood in a statistical sense: assuming that the nodes in $\Delta_{\mathcal{T}}$ are abnormal suffices to explain the observed performance gap $\rho(\mathcal{T}, S_D) - \rho(\mathcal{T}, S_{D'})$.

In principle, one could directly apply classical MBD solvers (de Kleer and Williams [1987], Metodi et al. [2014], Wang and Shen [2023]) to enumerate diagnoses. Yet, those solvers presuppose noise-free observations—every discrepancy is attributed to an actual fault. Decision trees, like any learned classifiers, inherently admit misclassification even under the training distribution. Thus, treating every error as evidence of a fault would inflate the diagnosis with healthy nodes. To overcome that, we adopt *Bayesian AppRoach to dIagnose iNtErmittent fauLts* (hereinafter BARINEL) (Abreu et al. [2009b]), a Bayesian MBD approach that accommodates noisy observations by ranking candidate diagnoses according to their posterior probabilities, down-weighting those that include nodes whose participation patterns do not strongly correlate with the observed misclassifications.

Candidates Selection: Starting from $SPCT_{\mathcal{T}}$ and $E_{\mathcal{T}}$, we first enumerate *minimal hitting sets* over the tree's nodes. A subset $\Delta_q \subseteq V_{\mathcal{T}}$ qualifies as a hitting set when every misclassified sample s_j (i.e., $e_{\mathcal{T}j} = 1$) has at least one node from Δ_q along its classification path:
$\forall s_j$ with $e_{\mathcal{T}j} = 1 : \quad \Delta_q \cap \{n_i \mid spct_{\mathcal{T}j,i} = 1\} \neq \emptyset$.
A hitting set is *minimal* when none of its proper subsets is itself a hitting set. The collection of all such minimal hitting sets forms the set of candidate diagnoses.

Candidates Ranking: Every candidate diagnosis Δ_q is ranked by its posterior probability of explaining the degradation, conditioned on the observed spectrum and error vector, following Bayes' theorem: $Pr(\Delta_q \mid SPCT_{\mathcal{T}}, E_{\mathcal{T}}) = \frac{Pr(SPCT_{\mathcal{T}}, E_{\mathcal{T}} | \Delta_q) \cdot Pr(\Delta_q)}{Pr(SPCT_{\mathcal{T}}, E_{\mathcal{T}})}$.
Because the marginal $Pr(SPCT_{\mathcal{T}}, E_{\mathcal{T}})$ is shared across candidates (normalization factor), ranking requires only the numerator. We assign the **prior** $Pr(\Delta_q)$ based on the STAT ranks (Definition 9), i.e., $p_i = rank_{STAT,\mathcal{T}}(n_i)$ is the prior fault probability of node n_i. Assuming independence among faults: $Pr(\Delta_q) = \prod_{n_i \in \Delta_q} p_i \cdot \prod_{n_i \in V_{\mathcal{T}} \setminus \Delta_q} (1 - p_i)$. The **likelihood** $Pr(SPCT_{\mathcal{T}}, E_{\mathcal{T}} \mid \Delta_q)$ is evaluated according to the BARINEL framework (Abreu et al. [2009b]). Candidates with larger posterior probabilities provide a more credible explanation for the observed misclassification pattern under the shifted distribution. The candidate that maximizes the posterior is chosen as the diagnosis of tree $\mathcal{T}$.

In contrast to SFL, which outputs a single node, BARINEL is capable of producing *multi-node* diagnoses, allowing the repair phase to target multiple drift-affected regions.

Returning to our running example (Table 2), the minimal hitting sets (candidates) are $\Delta_1 = \{n_1\}$, $\Delta_2 = \{n_3\}$, and $\Delta_3 = \{n_4, n_5\}$. Among these, BARINEL selects $\Delta_2 = \{n_3\}$. Recall that $p_{n_3} = 0.3$ (from the STAT rank computed earlier in Section 4.1.1); the prior of Δ_2 is $Pr(\Delta_2) = 0.3 \cdot \prod_{n_i \notin \{n_3\}} (1 - p_i)$, which incorporates both the fault probability of n_3 and the health probabilities of all remaining nodes. Its likelihood equals 1.0, since the participation column of n_3 perfectly discriminates misclassified from correctly classified samples.

4.2.2 BTAD Repair

Upon completion of the diagnosis phase, the set of faulty nodes $\Delta_{\mathcal{T}}$ is forwarded to the repair phase. For this target we introduce Subtree Re-fitting approach. Rather than performing localized parametric adjustments at individual nodes, this approach substitutes the entire decision subtree rooted at each diagnosed faulty node with a newly trained decision subtree, thereby enabling comprehensive structural reconfiguration. This strategy is advantageous when the distributional shift is too severe for a simple threshold adjustment or label reassignment to capture, as it allows the new subtree to learn an entirely new feature space partitioning.

The natural re-fitting training set for the replacement subtree rooted at the faulty node n_i is the collection of post-drift samples that reach it, $S_{D'}(n_i) = \{s_j \in S_{D'} \mid Path_{\mathcal{T}}(n_i)(s_j)\}$. However, training exclusively on $S_{D'}(n_i)$ risks overfitting the new subtree to a potentially small post-drift observation set. Augmenting it with samples from the original distribution D yields a more robust partitioning of the feature space. Since we assume that the raw pre-drift samples S_D are not available at repair time, we instead *synthesize* samples that approximate the pre-drift data reaching each faulty node. To this end, per-node generation parameters (as listed in Definition 14) are pre-computed during training and stored as a compact descriptor in each node in constant memory, enabling the generation of a synthetic dataset $\widehat{S_D(n_i)}$.

Definition 14 (Synthetic Sample Generation)**.** *Consider a tree $\mathcal{T}$, a diagnosed node $n_i \in \Delta_{\mathcal{T}}$, and the original training set S_D. Denote by $S_D(n_i) = \{s_j \in S_D \mid Path_{\mathcal{T}}(n_i)(s_j)\}$ the subset of original samples that traverse n_i. We construct a* ***synthetic dataset*** *$\widehat{S_D(n_i)}$ of the same cardinality $|S_D(n_i)|$ by independently drawing each sample according to the following per-feature scheme:*

- *Every* ***numeric feature*** *f is drawn from a Gaussian $\mathcal{N}(\mu_f, \sigma_f)$ and subsequently clipped to the interval $[\min_f, \max_f]$; the statistics μ_f, σ_f, $\min_f$, $\max_f$ are all derived from $S_D(n_i)$.*
- *Every* ***categorical or binary feature*** *f is drawn according to the observed frequency distribution:*
  $Pr(f = v) = \frac{|\{s_j \in S_D(n_i) | s_j[f] = v\}|}{|S_D(n_i)|}$.
- ***Class labels*** *are assigned by sampling from the empirical label distribution:*
  $Pr(y = c) = \frac{|\{s_j \in S_D(n_i) | y_j = c\}|}{|S_D(n_i)|}$.

Each replacement subtree is learned from the combined set $S_{D'}(n_i) \cup \widehat{S_D(n_i)}$, employing the same induction algorithm and hyperparameter configuration used to build the original

tree. Once trained, this new subtree is inserted into $\mathcal{T}$, taking the place of the subtree that was previously rooted at n_i.

When the diagnosis comprises several faulty nodes, some of them may stand in an ancestor–descendant relation within the tree. In such situations, a *dependency resolution* procedure is carried out **prior to** any subtree re-fitting.

Definition 15 (Dependency Resolution)**.** *Suppose $n_A, n_C \in \Delta_{\mathcal{T}}$ with n_A being an ancestor of n_C. Let $n_{A\to C}$ be the child of n_A on the path toward n_C, and $n_{A,other}$ be the remaining child of n_A (not on the same path). The resolution performs by removing n_A from $\Delta_{\mathcal{T}}$ and replacing it with $n_{A,other}$: $\Delta_{\mathcal{T}} \leftarrow \Delta_{\mathcal{T}} \setminus \{n_A\} \cup \{n_{A,other}\}$. This is justified because n_C already accounts for the degradation originating in the $n_{A\to C}$ subtree, making additional structural changes along that branch unnecessary.*

The underlying intuition is that when an ancestor and one of its descendants are both flagged as faulty, the ancestor's splitting criterion becomes redundant: the branch leading to the descendant will undergo its own dedicated repair, so the ancestor essentially reduces to its other branch. The resolution procedure iterates over the nodes in $\Delta_{\mathcal{T}}$ from shallowest to deepest: for each candidate n_A, it checks whether any deeper node $n_C \in \Delta_{\mathcal{T}}$ is a descendant of n_A (again scanning in top-to-bottom order). Upon finding such a pair, n_A is resolved as above and discarded from $\Delta_{\mathcal{T}}$. Processing in this breadth-first manner guarantees that shallower conflicts are addressed first, potentially eliminating deeper ones. Once all ancestor–descendant dependencies have been resolved, the retained nodes in $\Delta_{\mathcal{T}}$ are re-fitted independently.

A detailed complexity analysis is provided in Appendix A and at GitHub[1].

5 EVALUATION

5.1 EXPERIMENTAL SETTINGS

Datasets. A well-known difficulty in concept drift research is the scarcity of evaluation benchmarks that come with ground-truth drift annotations (Li et al. [2020], Brzezinski and Stefanowski [2013]). Most publicly available datasets are collected under stationary assumptions, while studies that do employ real-world drifting data often rely on proprietary sources (Duckworth et al. [2021], Wang et al. [2003]), preventing reproducibility. Consequently, researchers frequently turn to fully synthetic data with MOA framework (Bifet et al. [2010]), that can generate data streams with controlled distributional changes. However, MOA transforms the entire feature space at once, which does not suit our setting, where drift must be attributable to specific features for diagnosis evaluation. Following Tahmasbi et al. [2021], who proposed injecting drift into static datasets, and Almog and Kalech [2023], who applied a similar semi-synthetic strategy to decision-tree repair, we adopt a drift-injection approach over real classification benchmarks.

We draw our benchmarks from the publicly available TRIO collection[2] (Cohen-Shapira and Rokach [2021]), which comprises 138 classification datasets. Each dataset was then partitioned as follows: the first 70% of the instances serve as the pre-drift training set (S_D) used to build the decision tree. The remaining 30% is reserved for post-drift use ($S_{D'}$) and is further divided (randomly) into an *adaptation set*, whose size ranges from 0.5% to 10% of the full dataset, and a fixed 20% of *test set* used exclusively for evaluation after repair.

We applied two filtering criteria to retain only meaningful experimental cases: (1) datasets on which the trained tree achieved less than 75% accuracy on S_D were excluded, since the baseline model was already unreliable; and (2) datasets where the accuracy drop between S_D and $S_{D'}$ fell below 10% ($\theta_{\mathcal{T}}$, see Definition 3) were removed, as the drift was too mild to warrant repair.

Concept Drift Synthesis. We evaluate under two complementary drift-generation procedures, each producing a distinct experimental setting[2].

Feature-Shift Drift (Experiment 1). This protocol is designed to examine the effect of concept drift in scenarios where the change in each feature is applied in isolation from the others. Following the partitioning described in Section 5.1, 30% of the data is randomly selected to serve as the post-drift portion, and drift is injected into it by perturbing individual features. For each numeric feature, a shift $k \in \{-2\sigma, -\sigma, -0.5\sigma, 0.5\sigma, \sigma, 2\sigma\}$ (where σ is the feature's standard deviation) is added to every value. For categorical and binary features, a target value is fixed and a proportion $p \in \{0.3, 0.5, 0.7, 0.9\}$ of the remaining samples is overwritten with that value. This procedure yielded 3667 drift scenarios across 53 datasets.

Distribution-Edge Drift (Experiment 2). While feature-shift drift provides fine-grained control over the perturbation magnitude, modifying features in isolation can inadvertently disrupt existing inter-feature dependencies and correlations present in the data. This motivates a second, complementary protocol that preserves all inter-feature dependencies by partitioning *actual* data points rather than altering their values. For each target feature, instances are ranked by that feature's value, and either the top or bottom 30% constitute the post-drift portion; the remaining 70% serve as the pre-drift data. The focus on extreme values simulates a realistic distributional shift—the "edge" of the distribution emphasizes

[1]Complexity analysis is available at `https://github.com/yoavzelinger/BTAD/tree/main/appendix/complexity.pdf`.

[2]Datasets and resulted scenarios are available at `https://github.com/yoavzelinger/BTAD/tree/main/appendix/`.

that the underlying distribution has moved—while every other statistical relationship in the data remains intact. As in Experiment 1, the 30% post-drift portion is further split: one-third for adaptation and two-thirds for testing. This procedure yielded 2179 drift scenarios across 54 datasets.

Compared Algorithms. We compare five algorithms: three baselines, an ablation variant, and our proposed BTAD. The first two baselines retrain a decision tree from scratch. NEWDRIFT trains exclusively on the post-drift adaptation set $S_{D'}$, while NEWALL trains on the union $S_D \cup S_{D'}$. Note that NEWALL assumes continued access to the original training data, which may not always be available in practice. The third baseline is the APPETITE method (Almog and Kalech [2023]), which pairs the SFL diagnoser with the Node Adaptation repair operator (Section 4.1).

The ablation variant, replaces APPETITE's repair component with Subtree Re-fitting (SUBFIT) while retaining the SFL diagnosis, thereby isolating the contribution of our new repair strategy (denoted , SFL+SUBFIT) (Section 4.2.2). Finally, BTAD combines the BARINEL diagnoser with the SUBFIT repair operator (Section 4.2), providing insights into the diagnosis contribution and overall performance.

All algorithms' decision trees were trained using scikit-learn's `DecisionTreeClassifier` (Pedregosa et al. [2012]). Hyperparameters — the splitting criterion (Gini or Entropy) (Stoffel and Raileanu [2001]) and the maximum number of leaf nodes (10, 20, or 30) — were selected via Grid Search(Larochelle et al. [2007]) with 5-cross-validation on the training set. All procedures involving randomness used a fixed random seed of 7 to ensure reproducibility. The code available at Github [3]. Statistical significance is assessed via paired t-tests at $\alpha = 0.05$.

Evaluation Metrics. To assess **repair quality**, we measure the **Accuracy Diff**: the increase in classification accuracy of the repaired tree relative to the original (pre-repair) tree, both evaluated on the post-drift test set.

To assess **diagnosis quality**, we employ two metrics. First, **Wasted Effort** (Elmishali et al. [2018]) counts how many healthy nodes are inspected before all truly faulty ones, when following the diagnosis ranking order. A node is deemed *truly faulty* if its splitting feature is the feature into which drift was injected. When the ranked candidate list does not cover all faulty nodes, the expected wasted effort for the missing nodes is computed as $\mathbb{E}[WE_{missing}] = \frac{xm-x^2}{x+1}$, where m is the total number of unranked nodes and x is the number of truly faulty nodes among them. Second, **Top-1** measures the percentage of drift scenarios in which the drifted feature is ranked first in the diagnosis. Concretely, the ranked list of diagnosed nodes induces a ranked list of features; Top-1 is satisfied when the first distinct feature in that list matches the true drifted feature.

Both diagnosis quality metrics are reported exclusively for Experiment 1 (Feature-Shift). In Experiment 2 (Distribution-Edge), altering the partition boundary of a single feature can indirectly shift the conditional distributions of correlated features; consequently, there is no unambiguous faulty-features ground truth against which to measure fault localization. Therefore, we restrict diagnosis evaluation to Experiment 1.

Research Questions. We structure our evaluation around three research questions:

RQ1 How does BTAD compare to the baselines and to APPETITE in restoring accuracy after concept drift?

RQ2 How does the proportion of available post-drift data influence the effectiveness of each algorithm?

RQ3 How does the diagnostic quality of BARINEL compare to that of SFL? (only Experiments 1)

5.2 RESULTS

Algorithm	Experiment 1 Accuracy Diff	Experiment 2 Accuracy Diff
NEWALL	20.46 ± 20.09	19.77 ± 18.48
NEWDRIFT	9.10 ± 27.26	16.00 ± 22.82
APPETITE (Almog and Kalech [2023])	14.31 ± 21.82	1.88 ± 15.44
SFL+SUBFIT	15.59 ± 19.44	13.36 ± 17.71
BTAD	18.67 ± 20.16	19.07 ± 19.20

Table 3: Average Accuracy Difference ($\pm$ std). Higher is better. Variance is similar across methods except NEWDRIFT.

RQ1: BTAD Performance. Table 3 presents the average Accuracy Difference for both experimental settings. The results reveal several key observations. First, the new repair mechanism has a significant impact: the move from APPETITE's Node Adaptation operator to SUBFIT—while keeping the same SFL diagnosis—yields a consistent improvement: the SFL+SUBFIT variant raises the Accuracy Difference from 14.31 to 15.59 in Experiment 1 ($p < 10^{-6}$) and, most strikingly, from 1.88 to 13.36 in Experiment 2.

Second, the near-zero recovery of APPETITE in Experiment 2 underscores that local parameter modifications are fragile under wide-impact drift, whereas subtree re-fitting provides a structurally richer repair.

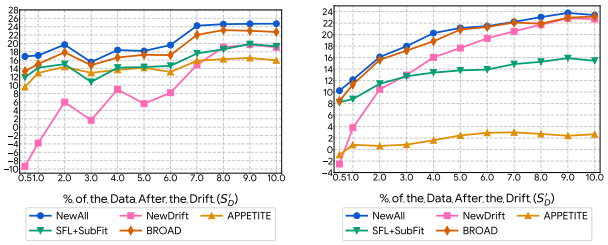
Third, upgrading the diagnosis from SFL to BARINEL—while keeping the same SUBFIT repair—further boosts performance: BTAD improves upon the SFL+SUBFIT variant from 15.59 to 18.67 in Experiment 1 ($p < 10^{-56}$) and from

[3]Code is available at `https://github.com/yoavzelinger/BTAD/tree/main/code/`

13.36 to 19.07 in Experiment 2. This underscores the importance of accurate diagnosis: BARINEL's Bayesian multi-node formulation identifies drift-affected regions more precisely than SFL, allowing the repair phase to focus on the right nodes, leading to more effective recovery.

The two contributions are complementary and cumulative: BTAD achieves 18.67 and 19.07 in Experiments 1 and 2 respectively, making it highly competitive with NEWALL (20.46 and 19.77)—a baseline that assumes continued access to the original pre-drift training data S_D, an assumption that may not hold in practical deployment. Meanwhile, NEWDRIFT, which retrains from scratch on post-drift data alone, recovers considerably less accuracy (9.10 and 16.00).

In Experiment 2, the drift's impact extends beyond a single node. This explains APPETITE's collapse: repairing one node (via a local parameter adaptation) is insufficient when multiple tree regions are affected. Similarly, SFL+SUBFIT's accuracy drops because SFL's single-node diagnosis misses the additional affected regions, even though SUBFIT itself provides sufficient repair capacity. BTAD, by contrast, remains more stable, as BARINEL's multi-node diagnosis can identify and repair all drift-affected regions simultaneously.



(a) Feature-Shift Drift (b) Distribution-Edge Drift

Figure 2: Accuracy Difference as a function of the proportion of post-drift data used for adaptation.

RQ2: Influence of Data Window Size. Figure 2 shows how the adaptation-set size affects each algorithm's Accuracy Difference. In Experiment 1 (Figure 2a), all algorithms improve as the adaptation set grows. NEWDRIFT is severely hampered by small adaptation sets, scoring -9.35 at 0.5% before gradually recovering to $\sim$ 19 at 8–10%. In contrast, the diagnosis-and-repair methods deliver positive gains even at the smallest window: APPETITE achieves 9.58, SFL+SUBFIT 11.89, and BTAD 13.34 at 0.5%. BTAD consistently outperforms both APPETITE and SFL+SUBFIT across all window sizes, and from 7% onward it closely tracks NEWALL that retains access to the original training data. The gap between BTAD and SFL+SUBFIT widens as more data becomes available, indicating that while BARINEL's multi-node diagnosis consistently performs better, it benefits more from richer evidence.

In Experiment 2 (Figure 2b), APPETITE remains largely ineffective (-0.91 to 2.99) regardless of adaptation-set size, confirming that local parameter adjustments fail to mitigate distribution-edge drift. SFL+SUBFIT provides moderate recovery (8.23 to 15.44), yet it plateaus early, reflecting the limitation of a single-node SFL diagnosis under multi-region drift. BTAD, by contrast, climbs steeply at small windows, rivaling NEWALL and outperforming NEWDRIFT.

Diagnoser	Wasted Effort	Top-1 (%)
SFL	11.19	53.83
BARINEL	5.35	73.06

Table 4: Average Wasted Effort (lower is better) and Top-1 accuracy (higher is better) for diagnosers on Experiment 1.

RQ3: Diagnosis Quality. Table 4 reports the Wasted Effort and Top-1 accuracy for SFL and BARINEL since only these algorithms provide a diagnostic ranking. Also, diagnosis metrics are reported only for Experiment 1, where known drifted features provide a clear ground truth. BARINEL substantially outperforms SFL on both metrics: it roughly halves the Wasted Effort (fewer healthy nodes are inspected before all truly faulty nodes are found) and raises Top-1 accuracy. Both improvements are statistically significant ($p \leq 5.95 \times 10^{-93}$ for both metrics). These results confirm that the Bayesian multi-node formulation underlying BARINEL provides a markedly more precise localization of drift-affected nodes than the correlation-based SFL ranking, explaining the accuracy gains observed in **RQ1**. Beyond better repairs, BARINEL improves root cause analysis by more accurately isolating the nodes responsible for drift.

6 CONCLUSION

We presented BTAD, a diagnosis-and-repair framework that adapts decision trees to concept drift in place—enabling a pre-trained tree distribution paradigm. BTAD contributes two complementary advances over the prior APPETITE approach: (i) formulating the DTCDP as a Model-Based Diagnosis problem solved via BARINEL, which produces Bayesian-ranked multi-node diagnoses, and (ii) Subtree Refitting, which replaces entire subtrees rooted at diagnosed nodes rather than applying local parameter adjustments. Experiments under two complementary drift scenarios show that both contributions are cumulative: (1) upgrading the repair from local adjustment to subtree re-fitting yields consistent gains, especially under broad distributional shifts where local tweaks are insufficient, and (2) upgrading the diagnosis from single-node SFL to multi-node BARINEL further improves both repair quality and fault localization.

Future work could explore replacing the per-feature independent synthetic sampling scheme (Definition 14) with modern tabular synthesis methods—such as conditional GANs (Xu et al. [2019]) or diffusion models (Kotelnikov et al. [2023]).

References

Rui Abreu, Peter Zoeteweij, and Arjan Gemund. On the accuracy of spectrum-based fault localization. In *Proceedings - Testing: Academic and Industrial Conference Practice and Research Techniques, TAIC PART-Mutation 2007*, pages 89–98, 10 2007. ISBN 978-0-7695-2984-4. doi: 10.1109/TAIC.PART.2007.13.

Rui Abreu, Peter Zoeteweij, Rob Golsteijn, and Arjan J.C. van Gemund. A practical evaluation of spectrum-based fault localization. *Journal of Systems and Software*, 82(11):1780–1792, 2009a. ISSN 0164-1212. doi: https://doi.org/10.1016/j.jss.2009.06.035. URL https://www.sciencedirect.com/science/article/pii/S0164121209001319.

Rui Abreu, Peter Zoeteweij, and Arjan J.C. van Gemund. Spectrum-based multiple fault localization. In *2009 IEEE/ACM International Conference on Automated Software Engineering*, pages 88–99, 2009b. doi: 10.1109/ASE.2009.25.

Shaked Almog and Meir Kalech. Diagnosis for post concept drift decision trees repair. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*, KR '23, 2023. ISBN 978-1-956792-02-7. doi: 10.24963/kr.2023/3. URL https://doi.org/10.24963/kr.2023/3.

Shahaf Bassan and Guy Katz. Towards formal xai: Formally approximate minimal explanations of neural networks, 2023.

Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, and Thomas Seidl. Moa: Massive online analysis, a framework for stream classification and clustering. In *Proceedings of the first workshop on applications of pattern analysis*, pages 44–50. PMLR, 2010.

Lyes Bounia and Jamal Agoun. Approximating user-preferred abductive explanations via supermodular optimization: Application to tree-based classifiers. In *Proceedings of the 19th European Conference on Logics in Artificial Intelligence (JELIA)*, 2025.

Dariusz Brzezinski and Jerzy Stefanowski. Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):81–94, 2013.

Paola Cabrera, German Castellanos-Domínguez, Carlos Mera, Luis E. Franco-Marín, and Mauricio Orozco-Alzate. Adaptive classification using incremental learning for seismic-volcanic signals with concept drift. *Journal of Volcanology and Geothermal Research*, 413:107211, 2021. doi: 10.1016/j.jvolgeores.2021.107211.

Clément Carbonnel, Martin C. Cooper, Emmanuel Hebrard, Dany Morales, and João Marques-Silva. Explaining multivariate decision trees: Characterising tractable languages. *Journal of Artificial Intelligence Research*, 0:1–26, 2026.

Noy Cohen-Shapira and Lior Rokach. TRIO: Task-agnostic dataset representation optimized for automatic algorithm selection. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 81–90. IEEE, 2021.

Martin C. Cooper and João Marques-Silva. Tractability of explaining classifier decisions. *Artificial Intelligence*, 316 (C), 2023. doi: 10.1016/j.artint.2022.103841.

Johan de Kleer and Brian C. Williams. Diagnosing multiple faults. *Artif. Intell.*, 32(1):97–130, 1987.

Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '00, pages 71–80, 2000. doi: 10.1145/347090.347107.

Christopher Duckworth, Francis P Chmiel, Dan K Burns, Zlatko D Zlatev, Neil M White, Thomas WV Daniels, Michael Kiuber, and Michael J Boniface. Using explainable machine learning to characterise data drift and detect emergent health risks for emergency department admissions during COVID-19. *Scientific reports*, 11(1):1–10, 2021.

Amir Elmishali, Roni Stern, and Meir Kalech. An artificial intelligence paradigm for troubleshooting software bugs. *Engineering Applications of Artificial Intelligence*, 69: 147–156, 2018.

Daniel P. Faith. Asymmetric binary similarity measures. *Oecologia*, 57:287–290, 1983.

Wenrui Gao, Zhonghao Bai, Feng Zhu, Clifford C Chou, and Binhui Jiang. A study on the cyclist head kinematic responses in electric-bicycle-to-car accidents using decision-tree model. *Accident Analysis & Prevention*, 160:106305, 2021.

Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. In *European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 2016.

Heitor Murilo Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfahringer, Geoff Holmes, and Talel Abdessalem. Adaptive random forests for evolving data stream classification. *Machine Learning*, 106:1–27, 2017. doi: 10.1007/s10994-017-5642-8.

Stefan Hoeglinger and Russel Pears. Use of Hoeffding trees in concept based data stream mining. In *2007 Third International Conference on Information and Automation for Sustainability*, pages 57–62. IEEE, 2007.

Xuanxiang Huang and Joao Marques-Silva. *From Decision Trees to Explained Decision Sets*, pages 1100–1108. IOS Press, 2023. ISBN 9781643684369. doi: 10.3233/FAIA230384.

Xuanxiang Huang, Yacine Izza, Alexey Ignatiev, Martin Cooper, Nicholas Asher, and Joao Marques-Silva. Tractable explanations for d-dnnf classifiers. In *AAAI Conference on Artificial Intelligence*, 2022.

Alexey Ignatiev, Yacine Izza, Peter Stuckey, and Joao Marques-Silva. Using maxsat for efficient explanations of tree ensembles. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:3776–3785, 2022. doi: 10.1609/aaai.v36i4.20292.

Yacine Izza, Alexey Ignatiev, Sasha Rubin, Joao Marques-Silva, and Peter J. Stuckey. Most general explanations of tree ensembles. 2025. doi: 10.24963/ijcai.2025/608. URL https://doi.org/10.24963/ijcai.2025/608.

Junqi Jiang, Francesco Leofante, Antonio Rago, and Francesca Toni. Formalising the robustness of counterfactual explanations for neural networks, 2023. URL https://doi.org/10.1609/aaai.v37i12.26740.

Martin Khannouz and Tristan Glatard. Mondrian forest for data stream classification under memory constraints. *Data Mining and Knowledge Discovery*, 38(2):569–596, 2024. doi: 10.1007/s10618-023-00970-4.

Akim Kotelnikov, Dmitry Barber, Ivan Novitskiy, and Dmitry Kiselev. TabDDPM: Modelling tabular data with diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 17564–17579, 2023.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th International Conference on Machine Learning*, pages 473–480, 2007.

Chee Sun Lee, Peck Yeng Sharon Cheang, and Massoud Moslehpour. Predictive analytics in business analytics: decision tree. *Advances in Decision Sciences*, 26(1):1–29, 2022.

Jinpeng Li, Hang Yu, Zhenyu Zhang, Xiangfeng Luo, and Shaorong Xie. Concept drift adaptation by exploiting drift type. *ACM Transactions on Knowledge Discovery from Data*, 18, 2024. doi: 10.1145/3638777.

Zeng Li, Wenchao Huang, Yan Xiong, Siqi Ren, and Tuanfei Zhu. Incremental learning imbalanced data streams with concept drift: The dynamic updated ensemble algorithm. *Knowledge-Based Systems*, 195:105694, 2020.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.

Joao Marques-Silva. Logic-based explainability in machine learning, 2023. URL https://doi.org/10.1007/978-3-031-31414-8_2.

Joao Marques-Silva. Logic-based explainability: Past, present and future. In *Leveraging Applications of Formal Methods, Verification and Validation. Software Engineering Methodologies*, pages 181–204, Cham, 2025. Springer Nature Switzerland.

Joao Marques-Silva and Xuanxiang Huang. Explainability is not a game. *Communications of the ACM*, 67(7):66–75, 2024.

Joao Marques-Silva and Alexey Ignatiev. Delivering trustworthy ai through formal xai. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:12342–12350, 2022. doi: 10.1609/aaai.v36i11.21499.

Amit Metodi, Roni Stern, Meir Kalech, and Michael Codish. A novel sat-based approach to model based diagnosis. *Journal of Artificial Intelligence Research*, 51:377–411, 2014.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Edouard Duchesnay, and Gilles Louppe. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12, 2012.

Giacomo Pelosi et al. Explainability and interpretability in concept and data drift: A systematic literature review. *Algorithms*, 18(2):98, 2025.

Radin Hamidi Rad and Maryam Amir Haeri. Hybrid forest: A concept drift aware data stream mining algorithm. *arXiv preprint arXiv:1902.03609*, 2019.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "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*, pages 1135–1144, 2016.

Daniel M. Roy and Yee Whye Teh. The mondrian process. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, pages 1377–1384, 2008.

Martin Scholz and Ralf Klinkenberg. An ensemble classifier for drifting concepts. In *Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams*, volume 6, pages 53–64, Porto, Portugal, 2005.

Siddharth Singhal, Utkarsh Chawla, and Rajeev Shorey. Machine learning and concept drift based approach for malicious website detection. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 582–585, 2020. doi: 10.1109/COMSNETS48256.2020.9027485.

Roni Stern, Meir Kalech, Alexander Feldman, and Gregory Provan. Exploring the duality in conflict-directed model-based diagnosis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 828–834, 2012.

Kilian Stoffel and Laura Elena Raileanu. Selecting optimal split-functions for large datasets. *Research Letters in the Information and Mathematical Sciences*, 2:73–77, 2001.

Ashraf Tahmasbi, Ellango Jothimurugesan, Srikanta Tirthapura, and Phillip B Gibbons. Driftsurf: Stable-state/reactive-state learning under concept drift. In *International Conference on Machine Learning*, pages 10054–10064. PMLR, 2021.

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235, 2003.

Zhenhua Wang and Yi Shen. Model-based fault diagnosis. *Cham, Switzerland: Springer*, 2023.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Structural Drift Repair in Decision Trees via Bayesian Model-Based Diagnosis (Supplementary Material)

Yoav Zelinger[1] **Meir Kalech**[1]

[1]The Stein Faculty of Computer Science and Information, Ben-Gurion University of the Negev, Be'er Sheva, Israel

A COMPLEXITY ANALYSIS

Let $S_D, S_{D'}$ denote the number of pre-drift and post-drift samples, F the number of features, V the number of nodes in the tree, and $\mathcal{H}$ the number of candidate diagnoses (minimal hitting sets) enumerated by BARINEL.

A.1 DIAGNOSIS COMPLEXITY

A.1.1 SFL Ranking.

STAT Computation The STAT rank (Definition 9) requires computing the pre-drift and post-drift constraint violation rates for every node in the tree. Each violation rate is obtained by passing each sample through the tree and checking constraint satisfaction at every visited node, with each sample visiting at most V nodes. Since we operate under the no-prior-data constraint, the pre-drift violation rates are pre-computed during training and stored as a scalar in each node. Thus, the STAT computation takes $\boldsymbol{O(S_{D'} \cdot V)}$ **time** and $\boldsymbol{O(V)}$ **space**.

Spectrum Computation Both SFL and BARINEL operate on the $SPCT_{\mathcal{T}} \in \{0,1\}^{S_{D'} \times V}$ node-sample spectrum and the $E_{\mathcal{T}} \in \{0,1\}^{S_{D'}}$ error vector. Constructing these requires passing each sample through the tree, recording participation at every visited node and the classification result, taking $\boldsymbol{O(S_{D'} \cdot V)}$ **time and space**.

Given the spectrum, the SFL score of each node is computed from its participation column and $E_{\mathcal{T}}$ using a similarity coefficient. For a typical coefficient such as Ochiai, this requires accumulating four counters per node by comparing each entry of the participation column against the corresponding entry in $E_{\mathcal{T}}$. These counters can be accumulated during spectrum construction and stored per node, resulting in additional constant time and space per node. Thus, the SFL ranking adds only $\boldsymbol{O(V)}$ **time and space**.

The final APPETITE score multiplies the two scalar ranks (SFL and STAT) for each node; the node maximizing this product is returned as the diagnosis (only the highest product needs to be stored). Altogether (STAT + Spectrum + SFL), the full APPETITE diagnosis requires:

- **Time:** $\boldsymbol{O(S_{D'} \cdot V)}$.
- **Space:** $\boldsymbol{O(S_{D'} \cdot V)}$.

A.1.2 BARINEL Ranking.

BARINEL reuses the same spectrum and STAT ranks (calculations in Section A.1.1). Its additional computation consists of two steps: candidate generation and candidate ranking. Since only the top-ranked diagnosis is needed for repair, candidates are scored on-the-fly and only the best is retained.

- **Candidate generation (MHS).** Following Abreu et al. [2009b], candidates are generated by computing minimal hitting sets over the classification paths of misclassified samples using the STACCATO algorithm, which takes $O(S_{D'} \cdot V)$ time and space (operates on the existing spectrum). This step produces a bounded number of candidates $\mathcal{H}$, capped at 100 in practice.
- **Candidate ranking.** For each candidate Δ_q, the posterior is computed from:
  - **Prior:** The product $\prod_{n_i \in \Delta_q} p_i \cdot \prod_{n_i \notin \Delta_q} (1 - p_i)$ using the STAT ranks as per-node fault probabilities. Besides the previous STAT cost, this calculation adds an additional $O(V)$ time and $O(1)$ space.
  - **Likelihood:** Computed via the maximum likelihood estimation procedure of Abreu et al. [2009b], which estimates per-component health parameters using gradient ascent with a constant number of iterations. This procedure takes $O(S_{D'} \cdot |\Delta_q|)$ time and $O(|\Delta_q|)$ space. However, since $|\Delta_q|$ is bounded in practice, this reduces the complexity to $O(S_{D'})$ time and $O(1)$ space per diagnosis.

  Since we only retain the top diagnosis, the total ranking cost is $O\big(\mathcal{H} \cdot (S_{D'} + V)\big)$ time and $O(1)$ space. However, as $\mathcal{H}$ is capped in practice, the time reduces to $O(S_{D'} + V)$.

Altogether (STAT + Spectrum + BARINEL), the full BTAD diagnosis requires:

- **Time:** $O\big(S_{D'} \cdot V + \mathcal{H} \cdot (S_{D'} + V)\big)$, where $\mathcal{H}$ is bounded in practice.
- **Space:** $O(S_{D'} \cdot V)$ (dominated by the spectrum; BARINEL adds only $O(1)$).

A.2 REPAIR COMPLEXITY

A.2.1 Retraining (NEWALL/ NEWDRIFT)

Both baselines train a CART decision tree from scratch.

- **NEWALL:** Trains on $S_D \cup S_{D'}$.
  - **Time:** $O\big((S_D + S_{D'}) \cdot F \cdot \log(S_D + S_{D'})\big)$.
  - **Space:** $O\big((S_D + S_{D'}) \cdot F\big)$.
- **NEWDRIFT:** Trains on $S_{D'}$ only.
  - **Time:** $O(S_{D'} \cdot F \cdot \log S_{D'})$.
  - **Space:** $O(S_{D'} \cdot F)$.

A.2.2 APPETITE Fix (Node Adaptation)

The repair modifies a single diagnosed node: a terminal node has its label reassigned to the majority class among its reaching samples ($O(S_{D'})$); a non-terminal numeric node has its threshold shifted by the change in mean ($O(S_{D'})$, using the pre-stored pre-drift mean and the post-drift mean computed from reaching samples); a non-terminal categorical node has its constraint negated ($O(1)$).

- **Time:** $O(S_{D'})$ - identifying reaching samples dominates.
- **Space:** $O(V)$ for non-terminal numeric nodes; $O(1)$ otherwise.

A.2.3 BTAD Fix (Subtree Re-fitting)

Our repair algorithm operates in three steps:

1. **Dependency Resolution.** Ancestor–descendant pairs in $\Delta_{\mathcal{T}}$ can be resolved (Definition 15) by a single top-down pass over the tree: each node is visited once, propagating ancestry information downward. Also, nodes are replaced in $\Delta_{\mathcal{T}}$ in-place, requiring no additional space. Thus, this process results with $O(V)$ time and $O(1)$ space.

2. **Synthetic Sample Generation.** For each faulty node $n_i \in \Delta_{\mathcal{T}}$, we generate $|S_D(n_i)|$ synthetic samples, each with F features (Definition 14). Per sample per feature the cost is $\boldsymbol{O(1)}$ (a Gaussian draw with clip, or a categorical draw). Generation parameters are pre-stored in each node in $\boldsymbol{O(F)}$ space. Since the retained nodes are pairwise non-ancestral, their reaching sets from S_D are disjoint, giving $\sum_{n_i \in \Delta_{\mathcal{T}}} |S_D(n_i)| \leq S_D$, leading to $\boldsymbol{O(S_D \cdot F)}$ time and $\boldsymbol{O\big(F \cdot (S_D + V)\big)}$ space.
3. **The Subtree Re-fitting.** Each node n_i is repaired by training a CART subtree on $|S_{D'}(n_i)| + |S_D(n_i)|$ samples with F features. By the same disjointness argument, the combined training data across all nodes is therefore bounded by $S_D + S_{D'}$. This leads the repair itself to take $\boldsymbol{O\big((S_D + S_{D'}) \cdot F \cdot \log(S_D + S_{D'})\big)}$ time and $\boldsymbol{O\big((S_D + S_{D'}) \cdot F\big)}$ space. The repair cost is at most that of full retraining (NEWALL), since the disjoint reaching sets partition the data.

Altogether (Dependency Resolution + Synthetic Sample Generation + Subtree Refitting), the full BTAD repair requires:

- **Time:** $\boldsymbol{O\big((S_D + S_{D'}) \cdot F \cdot \log(S_D + S_{D'})\big)}$.
- **Space:** $\boldsymbol{O\big((S_D + S_{D'} + V) \cdot F\big)}$.

A.3 SUMMARY

Table 5 summarizes the end-to-end complexity of each algorithm.

Algorithm	Time	Space
APPETITE Diagnosis	$O(S_{D'} \cdot V)$	$O(S_{D'} \cdot V)$
APPETITE Repair	$O(S_{D'})$	$O(V)$
BTAD Diagnosis	$O(S_{D'} \cdot V + \mathcal{H} \cdot (S_{D'} + V))$	$O(S_{D'} \cdot V)$
BTAD Repair	$O((S_D{+}S_{D'}) \cdot F \cdot \log(S_D{+}S_{D'}))$	$O((S_D{+}S_{D'}{+}V) \cdot F)$
Retraining (NEWALL)	$O((S_D{+}S_{D'}) \cdot F \cdot \log(S_D{+}S_{D'}))$	$O((S_D{+}S_{D'}) \cdot F)$
Retraining (NEWDRIFT)	$O(S_{D'} \cdot F \cdot \log S_{D'})$	$O(S_{D'} \cdot F)$

Table 5: Complexity comparison.

BTAD's diagnosis overhead relative to APPETITE is the MHS enumeration and BARINEL ranking ($\boldsymbol{O(\mathcal{H} \cdot (S_{D'} + V))}$), which is negligible in practice. Its repair phase is more expensive than APPETITE's ($\boldsymbol{O((S_D{+}S_{D'}) \cdot F \cdot \log(S_D{+}S_{D'}))}$ vs. $\boldsymbol{O(S_{D'})}$), yet it is bounded by the cost of full retraining (NEWALL) and produces structurally richer output (replacement subtrees rather than single-parameter adjustments).