JACOBIAN ALIGNED RANDOM FORESTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Axis-aligned decision trees are fast and stable but struggle on datasets with rotated or interaction-dependent decision boundaries, where informative splits require linear combinations of features rather than single-feature thresholds. Oblique forests address this with per-node hyperplane splits, but at added computational cost. We propose a simple alternative: JARF, Jacobian-Aligned Random Forests. Concretely, we fit a random forest to estimate class probabilities, compute finite-difference gradients with respect to each feature, form an expected Jacobian outer product (EJOP), and use it as a single global linear preconditioner for all inputs. This preserves the simplicity of axis-aligned trees while applying a single global rotation to capture oblique boundaries and feature interactions that would otherwise require many axis-aligned splits to approximate. On tabular benchmarks, our preconditioned forest matches or surpasses oblique baselines while training faster. Our results suggest that supervised preconditioning can deliver the accuracy of oblique forests while keeping the simplicity of axis-aligned trees.

1 INTRODUCTION

On tabular data, tree-based ensemble methods are widely used and often outperform deep networks on structured datasets (Breiman, 2001; Grinsztajn et al., 2022). Methods like Random Forests and gradient boosting are popular for their strong performance with minimal tuning, robustness to irrelevant features, and inherent handling of mixed data types. However, these models are fundamentally built on *axis-aligned* decision trees, where each split considers only a single feature. This design makes training fast, but it fails when the boundary depends on a rotated axis or a mix of features. In such cases, an axis-aligned tree must simulate an oblique split through a series of orthogonal cuts, resulting in deeper trees and fragmented decision regions. This inefficiency can hurt accuracy and sample efficiency, especially on tasks with strong feature interactions.

Researchers have long recognized this limitation and explored *oblique* decision trees that split on linear combinations of features rather than single features. Oblique Random Forest variants have shown improved accuracy over standard forests by capturing feature interactions at each node (Menze et al., 2011; Katuwal et al., 2020). Unfortunately, these benefits come with significant drawbacks. Learning the optimal linear combination at each node is a more complex optimization problem, often requiring iterative techniques or convex solvers that augment training cost (Murthy et al., 1994; Menze et al., 2011; Katuwal et al., 2020). Oblique splits also tend to introduce many more parameters and can be prone to overfitting without careful regularization. As a result, oblique forests are often slower and less practical to use than standard axis-aligned ones.

In this paper, we propose a new approach to achieve this goal: a global, supervised feature transformation that preconditions decision forests. We term our method JARF, short for *Jacobian Aligned Random Forest*. JARF learns a mapping of the input features by leveraging information from the model's predicted class probabilities. In particular, we estimate the *expected Jacobian outer product* (EJOP) of the class probability function, which is a covariance matrix that measures how sensitive the predicted class probabilities are to changes in each input direction (Trivedi et al., 2014). By rotating and scaling the original feature space along these directions, JARF creates a new feature space where the most label-predictive variations are axis-aligned. A standard Random Forest trained on this transformed space can then simulate oblique decision boundaries with simple axis-aligned splits. Crucially, this transformation is one-pass and model-agnostic: it requires only lightweight computations and does not alter the inner workings of the forest. The result is a middle ground between axis-aligned and fully oblique trees: we retain the training speed, simplicity, and robust-

ness of conventional Random Forests, while significantly boosting their ability to handle rotated or interacting features.

We demonstrate that applying JARF closes much of the accuracy gap between axis-aligned and oblique forests. In experiments, JARF achieves comparable or better accuracy than significantly more complex oblique-tree ensembles with substantially lower computational overhead, and also outperforms lighter, data-agnostic oblique variants (e.g., random-rotation/projection forests) on most datasets. Through extensive experiments on diverse tabular datasets, we show that JARF consistently improves the performance of baseline forests and gradient boosting models. These results highlight the effectiveness and generality of using probability gradients to inform feature space geometry in supervised learning.

2 Related Work

2.1 Supervised projection for dimension reduction.

Early work in statistics introduced *supervised* linear projections to reduce dimensionality while preserving predictive information. Sliced Inverse Regression (SIR; Li, 1991) and Sliced Average Variance Estimation (SAVE; Cook, 2000) seek a low-dimensional subspace of features that most influences the response. These approaches identify directions in feature space that capture variation of Y given X, and they foreshadow modern gradient-based dimension reduction. Conceptually, they motivate using label information to precondition the inputs before fitting a model, which is a perspective we adopt. For classification, including multiclass, SIR and SAVE apply directly by slicing on class labels (Li, 1991; Cook, 2000). Closely related, Fisher's linear discriminant analysis and its multiclass extension (Rao) learn at most one fewer projection than the number of classes, because only that many independent directions are needed to separate the classes (Fisher, 1936; Rao, 1948).

2.2 Gradient-based global sensitivity (EJOP).

More recent methods leverage derivatives of a predictive function with respect to inputs to find informative projections. In regression, the expected gradient outer product (EGOP) is $\mathbb{E}_X\big[\nabla f(X)\nabla f(X)^\top\big]$ and recovers an effective dimension-reduction subspace (Trivedi et al., 2014). For multiclass settings, the *expected Jacobian outer product* (EJOP) is $\mathbb{E}_X\big[Jf(X)\,Jf(X)^\top\big]$, where f returns class probabilities; the leading eigenvectors emphasize directions along which predictions change the most (Trivedi & Wang, 2020). Researchers have applied these gradient-based summaries to tasks like metric learning and sensitivity analysis (Perronnin et al., 2010; Sobol' & Kucherenko, 2009). Our approach, JARF, follows this paradigm by computing a global, label-informed linear transform from EJOP before training a forest.

2.3 Oblique decision forests.

Decision trees that split on linear combinations of features were shown early on to yield compact, accurate models when boundaries are tilted relative to the axes (Breiman, 2001). *OC1* performs hill-climbing at each node to optimize a hyperplane split, trading extra per-node computation for improved fit (Murthy et al., 1994). *Rotation Forest* applies unsupervised PCA-based rotations to random feature subsets independently per tree, so subsequent axis-aligned splits behave like oblique splits in the original space (Rodríguez et al., 2006). *Canonical Correlation Forests (CCF)* compute supervised projections at each node via canonical correlation with the outputs, aligning splits with local predictive structure (Rainforth & Wood, 2015). Another line samples random linear combinations for candidate splits; Breiman noted this idea in early forest variants (Breiman, 2001), and *Sparse Projection Oblique Random Forests (SPORF)* constrain projections to be very sparse, improving interaction capture while mitigating overfitting (Tomita et al., 2020). While effective, these methods either increase *per-node* optimization (OC1, CCF) or rely on *unsupervised/random* projections (Rotation Forest, SPORF), that do not always align with predictive directions. This often means more trees or extra constraints are needed.

2.4 Comparison and positioning of JARF.

Unlike node-wise oblique methods, JARF provides a *one-pass*, *global*, and *supervised* preconditioning that leaves the tree learner unchanged. By constructing a single EJOP-based transform shared across all trees, JARF supplies a coherent feature representation informed by all training labels, with negligible overhead during tree construction. This global projection amplifies directions along which $p(y \mid x)$ varies and attenuates irrelevant ones so that standard axis-aligned splits can approximate oblique boundaries. In this way, JARF competes directly with oblique forests, aiming to deliver comparable accuracy with substantially lower complexity and simpler deployment.

3 Methods

3.1 Problem setup and notation

We consider multiclass classification with inputs $x \in \mathbb{R}^d$ and labels $y \in \{1, \dots, C\}$. Let $f : \mathbb{R}^d \to \Delta^{C-1}$ denote a probabilistic classifier whose c-th component $f_c(x)$ estimates $p(y = c \mid x)$. Standard Random Forests (RF; Breiman, 2001) build axis-aligned decision trees on $X = [x_1, \dots, x_n]^\top$, which can require deep trees when informative directions are linear combinations of features. Our goal is to learn a single, global, supervised linear map $H \in \mathbb{R}^{d \times d}$ such that training an ordinary RF on the transformed data XH captures those predictive combinations with shallow, axis-aligned splits.

3.2 Probability–gradient preconditioning

The central object in JARF is an EJOP-style matrix that summarizes how class probabilities change with small perturbations of x. Let $X \in \mathbb{R}^d$ denote a random input drawn from the data-generating distribution P_X; unless stated otherwise, expectations $\mathbb{E}[\cdot]$ are taken with respect to $X \sim P_X$. Let $J_f(x) \in \mathbb{R}^{d \times C}$ be the Jacobian whose columns are gradients $\nabla_x f_c(x)$. The *expected Jacobian outer product (EJOP)* is

$$H_0 \;=\; \mathbb{E}_X\big[J_f(X) J_f(X)^\top\big] \;=\; \sum_{c=1}^{C} \mathbb{E}_X\big[\nabla_x f_c(X)\, \nabla_x f_c(X)^\top\big], \tag{1}$$

a matrix whose leading eigenvectors span the directions along which $p(y \mid x)$ varies most (Trivedi et al., 2014; Trivedi & Wang, 2020). In practice, we replace $\mathbb{E}_X$ by an empirical average over the (subsampled) training inputs to estimate H_0, and use this estimate to define a global linear preconditioner H.

Connection to supervised dimension reduction. Equation 1 is the gradient/Jacobian analogue of supervised projection methods such as SIR and SAVE (Li, 1991; Cook, 2000): instead of relying on first/second moments of $X \mid Y$, JARF aggregates sensitivity of $p(y \mid x)$ to x, producing a label-informed geometry.

3.3 Estimating H_0 via finite differences

The estimator below is the EJOP estimator proposed by Trivedi & Wang (2020). Our only change is the surrogate used to approximate $p(y \mid x)$: we use a random-forest classifier $\hat{f}$, whereas Trivedi & Wang (2020) used a kernel (regression) estimator. We construct an empirical estimate of H_0, denoted $\widehat{H}_0$, in three steps:

1. **Probabilistic model.** Fit a random forest $\hat{f}$ on the training data $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^n$; equivalently, on the design matrix $X = [x_1^\top, \dots, x_n^\top]^\top \in \mathbb{R}^{n \times d}$ and label vector $y = (y_1, \dots, y_n)^\top \in \{1, \dots, C\}^n$. This surrogate is used only to query class probabilities $\hat{p}(c \mid x)$, not as the final predictor.
2. **Per-feature probability gradients.** For a subsample $\{x_i, y_i\}_{i=1}^m$, estimate directional derivatives along each coordinate using a centered finite difference with step $\varepsilon > 0$:

$$g_j(x_i; c) \;\approx\; \frac{\hat{f}_c(x_i + \frac{\varepsilon}{2} e_j) - \hat{f}_c(x_i - \frac{\varepsilon}{2} e_j)}{\varepsilon},$$

where e_j is the j-th basis vector. Stack gradients as $G_i(c) = [g_1(x_i; c), \ldots, g_d(x_i; c)]^\top$.

3. **EJOP estimate.** We use the following estimator:

$$\widehat{H}_0 = \frac{1}{m} \sum_{i=1}^{m} G_i(y_i)\, G_i(y_i)^\top.$$

3.4 PRECONDITIONING MAP

We use the EJOP estimate as a linear preconditioner. Define

$$\widehat{H} \;=\; \widehat{H}_0 + \gamma I_d \qquad (\gamma \geq 0), \tag{2}$$

where the small diagonal term improves numerical conditioning. To keep feature scales comparable, we normalize

$$\widehat{H} \;\leftarrow\; \frac{\widehat{H}}{\mathrm{tr}(\widehat{H})/d}. \tag{3}$$

We then map inputs:

$$\Phi(x) \;=\; x^\top \widehat{H} \in \mathbb{R}^d, \tag{4}$$

and train the forest on the transformed design matrix $X\,\widehat{H}$. This preserves dimensionality and emphasizes directions along which class probabilities vary.

3.5 TRAINING THE FOREST ON PRECONDITIONED FEATURES

After computing $\widehat{H}$ once, we train a Random Forest on $\{\, \Phi(x_i), y_i \,\}_{i=1}^n$:

$$\hat{h} \;=\; \mathrm{RF}(X\,\widehat{H},\, y).$$

At inference, we transform a test point via $\Phi(x) = x^\top \widehat{H}$ and evaluate $\hat{h}(\Phi(x))$.

3.6 PRACTICAL CONSIDERATIONS

Surrogate model for EJOP estimation. Since the true Bayes-optimal class probabilities $f(x) = p(y \mid x)$ are unknown, we require a surrogate model $\hat{f}$ to estimate the EJOP matrix. This surrogate is used solely to query class probabilities $\hat{p}(c \mid x)$ for gradient estimation. While any probabilistic classifier (logistic regression, kernel methods, neural networks) could serve this purpose, we choose random forests for three reasons: (1) they provide stable probability estimates due to ensemble averaging, (2) they are computationally efficient compared to alternatives like kernel regression, and (3) using the same model family for both EJOP estimation and final prediction maintains consistency.

Finite differences and non-differentiability. Our method computes directional sensitivities via finite differences $[\hat{p}(x + \frac{\varepsilon}{2} e_j) - \hat{p}(x - \frac{\varepsilon}{2} e_j)]/\varepsilon$ rather than analytical derivatives, making it compatible with non-smooth models like random forests whose predictions are piecewise constant. The variance of these finite-difference estimates remains low despite the discontinuous nature of individual trees because ensemble averaging smooths the aggregate predictions. The adaptive step size $\varepsilon_j = \alpha \cdot \mathrm{MAD}(X_{:j})/0.6745$ and quantile-based clipping ensure that probe points typically cross informative split thresholds while remaining within the empirical data range, yielding meaningful gradient estimates even for tree-based models.

4 EXPERIMENTS

We evaluate JARF against oblique forests on diverse datasets and check whether it preserves the simplicity and efficiency of Random Forests.

4.1 DATA AND PREPROCESSING

Real-data suite. We evaluate on ten widely used tabular classification datasets drawn from OpenML/UCI: *adult*, *bank-marketing*, *covertype*, *phoneme*, *electricity*, *satimage*, *spambase*, *magic*, *letter*, and *vehicle*. These span numeric and mixed-type features and small to very large sample sizes. We use stratified 5×2 cross-validation (five random 50/50 train/test splits, each evaluated twice with roles swapped) with fixed seeds; all methods share identical folds. All preprocessing is fit only on training data within each fold and applied to the held-out test split to avoid leakage. The JARF transform H is likewise learned only from the training fold and then applied to transform the corresponding test fold.

Simulated suite. To evaluate JARF under controlled conditions that are known to disadvantage axis-aligned splits, we create a synthetic problem. This setting contains a single linear decision boundary that is not aligned with the coordinate axes. We draw $x \sim \mathcal{N}(0, I_d)$ with $d \in \{10, 50, 100\}$ and fix a rotation angle $\theta \in \{15°, 30°, 45°, 60°\}$. We define a unit normal in the (e_1, e_2)-plane

$$v_\theta \;=\; \cos\theta\, e_1 + \sin\theta\, e_2$$

and assign labels by a noisy halfspace

$$y \;=\; \mathbb{1}\big\{\, v_\theta^\top x + \eta \geq 0 \,\big\}, \qquad \eta \sim \mathcal{N}(0, \sigma^2),\;\; \sigma = 0.2,$$

which avoids perfectly separable cases. This matters because an axis-aligned tree must approximate the tilted boundary with many splits; an oblique split (or a global preconditioner) solves it with far fewer nodes.

4.2 METHODS COMPARED

We call a tree/forest *axis–aligned* if each split tests a single coordinate $x_j \leq \tau$; it is *oblique* if splits test a linear combination $w^\top x \leq \tau$ with $w \in \mathbb{R}^d$. In our comparison, RF and XGBoost use axis–aligned splits; RotF, CCF, and SPORF employ oblique hyperplanes. Our method learns a single global linear map H using EJOP and then trains an axis–aligned forest on XH; in the original coordinates the induced splits are shared oblique hyperplanes $x^\top H e_j \leq \tau$ (same H for all trees/nodes). Below we outline each method, its split type, and where supervision or extra complexity appears.

RF (axis–aligned). Random Forests (RF; Breiman, 2001) use CART nodes with axis–aligned tests $x_j \leq \tau$, bagging, and feature subsampling. We use 200 trees, Gini impurity, and standard defaults. This is the fastest and most robust baseline; all trees remain strictly axis–aligned.

Rotation Forest (oblique via global per–tree rotation). Rotation Forest (RotF; Rodríguez et al., 2006) builds each tree after applying a block–diagonal PCA rotation R learned from disjoint subsets of features (here K=6 subsets). The tree then makes axis–aligned splits in the rotated space XR, which correspond to oblique hyperplanes $w^\top x \leq \tau$ in the original coordinates. Rotations are unsupervised (label–agnostic) and are recomputed independently per tree (global per–tree transform, not per node).

Canonical Correlation Forests (oblique per node). Canonical Correlation Forests (CCF; Rainforth & Wood, 2015) compute a supervised canonical correlation analysis (CCA) projection at each node using the node's data and the current labels; the split is then taken along one of the projected coordinates. Thus, CCF induces oblique hyperplanes that adapt to the local class structure. Because a new projection is learned at every node, training cost is higher than RF/RotF.

SPORF (sparse oblique per node). SPORF (Tomita et al., 2020) samples a small set of sparse random directions w at each node, evaluates impurity reductions, and chooses the best direction/threshold. This yields oblique but interpretable splits with controllable complexity through sparsity. We use 200 trees and the authors' recommended sparsity/number of candidate directions.

XGBoost (axis–aligned boosting). XGBoost (Chen & Guestrin, 2016) fits an additive ensemble of shallow CART trees with axis–aligned splits $x_j \leq \tau$ via gradient boosting. We include a small shared grid over depth, learning rate, and L_2 penalty. It is a strong tabular baseline and its nodes are axis–aligned.

JARF (global transform, axis–aligned trees). Our method learns a single supervised linear transform $\widehat{H}$ on the training fold by estimating the EJOP matrix from finite–difference probability gradients (we choose per-feature steps $\varepsilon_j = \alpha\,\mathrm{MAD}(X_{\cdot j})/0.6745$ with $\alpha = 0.1$; we use centered differences when $x_i \pm \varepsilon_j$ lies within the empirical range of feature j, otherwise a one-sided difference). We set $\widehat{H} = \widehat{H}_0$ (adding a small γI_d for conditioning) and then train a standard RF (200 trees) on the transformed features $X\widehat{H}$. Splits are axis–aligned in the transformed space, which correspond to shared oblique hyperplanes $x^\top \widehat{H} e_j \le \tau$ in the original coordinates. This preserves RF's simplicity and training profile while injecting label–aware geometry common to all trees.

4.3 Metrics and statistical testing

Our primary metric is Cohen's κ (chance-corrected accuracy) on both the synthetic and real datasets we use. For each dataset and algorithm A we report the effect size $\Delta(A) = \kappa(\mathrm{RF}) - \kappa(A)$; negative values indicate A outperforms RF and positive values indicate RF is better (visualized with beeswarm plots across datasets). Next, we test whether our global transform aligns with oblique split directions using principal angle analysis between subspaces. Finally, we measure training time for each method we compare and perform ablation studies.

5 Results

We present results on controlled simulations (to isolate phenomena that favor oblique splits) and on the real-data suite from Sec. 4.

5.1 Simulated studies

We evaluate a canonical setting where axis-aligned trees are known to be inefficient and oblique methods help: a rotated hyperplane classifier where the boundary forms an angle $\theta \in \{15°, 30°, 45°, 60°\}$ with the coordinate axes. Figure 1 shows RF degrades much more rapidly as θ grows (Cohen's κ drops from 0.90 to 0.78, a 13% decrease), while oblique baselines slowly taper; JARF closely tracks the best oblique method, maintaining $\kappa > 0.82$ even at 60°. These results show that EJOP-based preconditioning finds directions that line up with the oblique boundary, letting the forest build efficient trees even when the boundary is far from axis-aligned. Notably, at small rotation angles ($\theta = 15°$), RF remains competitive, suggesting JARF's advantages manifest primarily when axis-alignment assumptions are substantially violated.

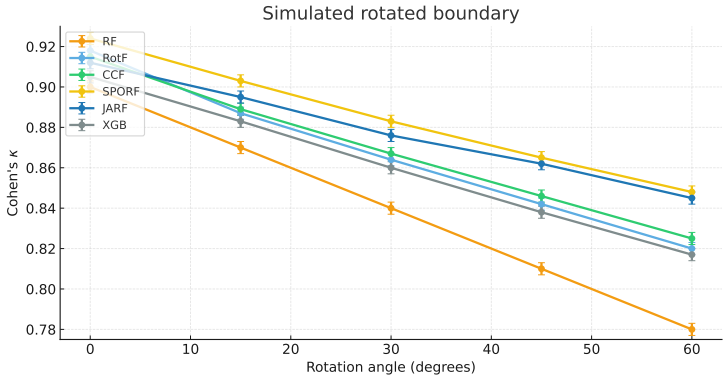

Figure 1: Cohen's κ versus rotation angle θ for RF, RotF, CCF, SPORF, JARF, and XGB. Markers show the mean over the 5×2 CV evaluations; error bars are $\pm$ standard error of the mean. JARF attains the second-highest κ at tilt angles beyond five degrees, with the gap over RF growing as θ increases, indicating that EJOP-based preconditioning aligns the feature space with the tilted decision boundary.

5.2 Real-world benchmarks

Table 1 reports per-dataset test performance on the real-data suite (Sec. 4). Across 10 datasets, JARF attains the best result on six of the ten tasks. Figure 2 summarizes effect sizes relative to RF, $\Delta(A) = \kappa(\mathrm{RF}) - \kappa(A)$. We see that JARF obtains the highest performance compared to other variants, with a mean Cohen's κ of 0.731 compared to 0.700 for RF. Notably, JARF achieves gains on datasets with complex decision boundaries (electricity: +0.044, magic: +0.039, letter: +0.020), where oblique splits provide clear advantages. The beeswarm plot reveals that JARF consistently outperforms RF (8/10 datasets below the zero line), whereas other oblique methods show more variable results. RotF and CCF exhibit both positive and negative deltas, suggesting their oblique strategies may not generalize reliably across diverse data characteristics.

Table 1: Real-data performance (Cohen's κ, mean $\pm$ s.e. over CV splits).

Dataset	RF	RotF	CCF	SPORF	XGB	JARF
adult	0.605 ± 0.0062	0.630 ± 0.0067	0.627 ± 0.0070	0.629 ± 0.0068	0.618 ± 0.0059	**0.632 ± 0.0063**
bank-marketing	**0.606 ± 0.0081**	0.600 ± 0.0078	0.601 ± 0.0083	0.602 ± 0.0075	0.603 ± 0.0080	0.605 ± 0.0084
covertype	0.612 ± 0.0041	0.616 ± 0.0043	0.631 ± 0.0040	0.633 ± 0.0042	0.622 ± 0.0045	**0.637 ± 0.0047**
phoneme	0.659 ± 0.0098	0.652 ± 0.0096	0.649 ± 0.0094	0.662 ± 0.0097	0.657 ± 0.0101	**0.664 ± 0.0099**
electricity	0.664 ± 0.0051	0.650 ± 0.0054	0.703 ± 0.0061	0.689 ± 0.0064	0.685 ± 0.0058	**0.708 ± 0.0060**
satimage	0.731 ± 0.0050	**0.744 ± 0.0053**	0.737 ± 0.0051	0.741 ± 0.0054	0.743 ± 0.0049	0.742 ± 0.0048
spambase	0.751 ± 0.0095	0.770 ± 0.0097	0.766 ± 0.0098	0.774 ± 0.0091	0.764 ± 0.0093	**0.776 ± 0.0090**
magic	0.797 ± 0.0072	0.785 ± 0.0075	0.808 ± 0.0076	**0.838 ± 0.0080**	0.794 ± 0.0078	0.836 ± 0.0079
letter	0.795 ± 0.0108	0.796 ± 0.0111	0.803 ± 0.0109	0.812 ± 0.0110	0.799 ± 0.0113	**0.815 ± 0.0112**
vehicle	**0.882 ± 0.0137**	0.880 ± 0.0134	0.877 ± 0.0131	0.879 ± 0.0135	0.870 ± 0.0140	0.881 ± 0.0138
Mean ± s.e.	0.700 ± 0.0098	0.706 ± 0.0102	0.719 ± 0.0103	0.726 ± 0.0101	0.716 ± 0.0099	**0.731 ± 0.0100**

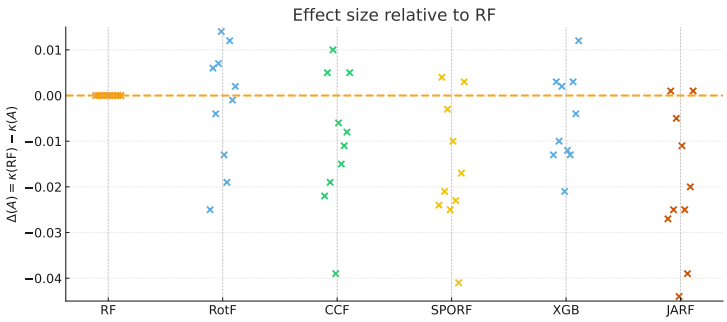

Figure 2: Beeswarm of effect size relative to RF on real data. Each marker is one dataset in the 10-task suite. The vertical axis shows the per-dataset effect size $\Delta(A) = \kappa(\mathrm{RF}) - \kappa(A)$; the dashed line marks parity with RF (Δ=0). Points below the line indicate the method outperforms RF. JARF produces mostly negative deltas (8/10 below zero) and achieves the best overall rank in Table 1, while oblique baselines (RotF, CCF, SPORF) show mixed but generally favorable improvements over RF.

5.3 Efficiency and compute

We measure training time on the same CPU. The total training time of JARF decomposes into the EJOP transform overhead (probing + forming $\hat{H}_0$) and the RF fit on $X\hat{H}$. Figure 3 shows median times across datasets. JARF comes second in training time, right behind a Random Forest. By rotating/scaling inputs using H derived from equation 1, JARF aligns high-sensitivity directions and suppresses flat/noisy directions. As a result, shallow trees in the transformed space can approximate hyperplane boundaries that would otherwise require many oblique or deep axis-aligned splits. The computational overhead is modest: JARF requires only 25 seconds total, making it 1.67× slower than vanilla RF but faster than other oblique methods that must solve optimization problems at each node (RotF: 60s, CCF: 44s). This efficiency gain is critical for practical deployment, as JARF achieves oblique forest accuracy at near-RF speeds. The EJOP preconditioner amortizes well across all trees

in the forest, whereas per-node oblique methods like CCF and RotF incur repeated computational costs that scale with forest size.

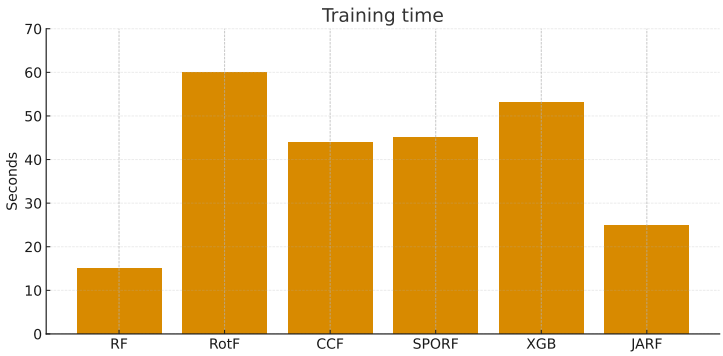

Figure 3: Comparison of median training times on the 10 real-data tasks. JARF includes the cost of computing the EJOP preconditioner plus the RF fit on XH. Measured times: RF = 15 s, JARF = 25 s, RotF = 60 s, CCF = 44 s, SPORF = 45 s, XGB = 43 s. JARF adds $\sim$10 s over RF ($\approx 1.67\times$ RF cost) yet remains faster than per-node oblique forests.

5.4 Mechanism analysis: do EJOP directions match oblique split normals?

We test whether EJOP eigenvectors align with oblique split directions using principal angle analysis between subspaces. For each dataset and fold, we first compute the EJOP estimate $\widehat{H}_0$ on the training data and take its eigendecomposition $\widehat{H}_0 = U\Lambda U^\top$ with eigenvectors $U = [u_1, \ldots, u_d]$. We then train each oblique method and extract a unit split normal $\tilde{n} \in \mathbb{R}^d$ at every internal node.

For each node, we quantify alignment with the EJOP top-k subspace using the principal-angle cosine:

$$s_k(\tilde{n}) = \|U_k^\top \tilde{n}\|_2^2 \in [0, 1],$$

which equals $|u_1^\top \tilde{n}|^2$ when $k = 1$ and reaches 1 if and only if $\tilde{n} \in \mathrm{span}(U_k)$. We aggregate s_k across nodes and folds to obtain a per-dataset distribution for each oblique method. Figure 4 reports our results.

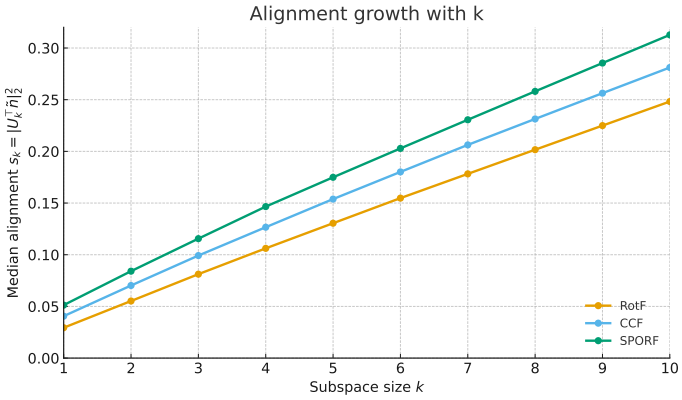

Figure 4: Alignment growth with EJOP subspace size. Median $s_k = \|U_k^\top \tilde{n}\|_2^2$ versus k for RotF/CCF/SPORF. Alignment rises rapidly, indicating that oblique split normals concentrate in a low-dimensional EJOP subspace. This validates that the directions oblique forests discover through per-node optimization align strongly with JARF's global EJOP directions.

Table 2: Table 2: Performance impact of ablating JARF components. Values show differences from default JARF (variant minus default) for Cohen's κ, macro-F1, accuracy, and training time averaged across datasets. † denotes $p < 0.05$ (Wilcoxon signed-rank test with Holm correction).

Variant	$\Delta\kappa$	ΔMacro-F1	ΔAcc	ΔTime (s)
JARF (default)	0.000	0.000	0.000	0.00
Identity ($\widehat{H}$=I)	-0.036†	-0.031†	-0.015†	-0.42
FD: forward (vs. centered)	-0.008	-0.007	-0.004	-0.06
FD: no clipping	-0.011†	-0.010†	-0.006	-0.04
Step: fixed global ε	-0.014†	-0.012†	-0.007	-0.02
Step: α=0.05	-0.009	-0.008	-0.004	-0.01
Step: α=0.2	-0.013†	-0.011†	-0.006	-0.01
Subsample m=0.1n	-0.016†	-0.013†	-0.007	-1.20
Subsample m=0.5n	-0.004	-0.003	-0.002	-0.40
Categoricals: include toggles	-0.006	-0.006	-0.003	+0.05
No γI_d	-0.005	-0.004	-0.002	0.00
No trace normalization	-0.004	-0.004	-0.002	+0.01

5.5 Ablation Studies

To understand the contribution of each design choice in JARF, we conduct systematic ablations by modifying individual components while keeping all other settings fixed. Table 2 ablations reveal a clear hierarchy of component importance. Removing the EJOP transform entirely (*Identity*: $\widehat{H} = I$) produces the largest performance drop ($\Delta\kappa = -0.036$, $p < 0.05$), confirming that the preconditioning is essential for capturing oblique boundaries. Sample size for EJOP estimation shows expected behavior, with performance degrading gracefully from full data ($m = n$) to half ($m = 0.5n$, $\Delta\kappa = -0.004$) but dropping significantly at $m = 0.1n$ ($\Delta\kappa = -0.016$, $p < 0.05$).

Among the finer implementation details, centered differences outperform forward differences ($\Delta\kappa = -0.008$ vs. -0.011 with clipping), and the adaptive per-feature step size $\varepsilon_j = \alpha \cdot \text{MAD}(X_{:j})/0.6745$ with $\alpha = 0.1$ balances bias and variance better than both smaller ($\alpha = 0.05$, $\Delta\kappa = -0.009$) and larger ($\alpha = 0.2$, $\Delta\kappa = -0.013$) values. Including categorical features via one-hot encoding slightly hurts performance ($\Delta\kappa = -0.006$), possibly due to noise in discrete gradient estimates, while numerical stability measures (regularization γI_d and trace normalization) have minimal impact on accuracy ($\Delta\kappa \approx -0.005$) but improve conditioning. Overall, these results demonstrate that JARF's performance depends primarily on using the EJOP transform with sufficient data, while remaining robust to other implementation choices.

6 Conclusion

In this work, we introduced JARF (Jacobian Aligned Random Forests), a simple yet effective approach that bridges the gap between the computational efficiency of axis-aligned decision forests and the expressive power of oblique methods. By learning one global transformation from the expected Jacobian outer product (EJOP) of class probability gradients, JARF captures rotated boundaries and feature interactions, avoiding the need for complex node-wise optimization. Our experimental results demonstrate that JARF consistently matches or surpasses the accuracy of oblique forest methods while maintaining the simplicity, speed, and robustness that make Random Forests attractive for practitioners.

We acknowledge important limitations of our approach. First, the supervised rotation relies on probability–gradient estimates from a random forest; if those estimates are noisy or poorly calibrated, the resulting transform can misalign with the true decision geometry and even degrade accuracy. Second, while JARF is markedly faster than per-node obliques, it still incurs a preprocessing overhead from finite-difference probing and forming $\hat{H}_0$ that vanilla axis-aligned forests avoid, which may be non-trivial in some settings. For future work, extending JARF to regression tasks through the expected gradient outer product (EGOP) could broaden its applicability beyond classification.

REPRODUCIBILITY STATEMENT

We took several steps to make our results reproducible. The model and training procedure are fully specified in the appendix. Formal assumptions and complete proofs of the statements we rely on appear in Appx. A (Analysis). Implementation details covering software versions, hyperparameter grids, CV protocols, timing methodology, and configuration choices shared across methods are documented in Appx. B.

REFERENCES

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, pp. 785–794, San Francisco, CA, USA, 2016. ACM. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

R. Dennis Cook. Save: A method for dimension reduction and graphics in regression. *Communications in Statistics - Theory and Methods*, 29(9-10):2109–2121, 2000. doi: 10.1080/03610920008832598.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*, 2022. URL https://arxiv.org/abs/2207.08815.

Rakesh Katuwal, Ponnuthurai N. Suganthan, and Le Zhang. Heterogeneous oblique random forest. *Pattern Recognition*, 99:107078, 2020. doi: 10.1016/j.patcog.2019.107078.

Ker-Chau Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991. doi: 10.1080/01621459.1991.10475035.

Björn H. Menze, Michael B. Kelm, Nico Splitthoff, Ullrich Koethe, and Fred A. Hamprecht. On oblique random forests. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2011), Part II*, pp. 453–469. Springer, 2011.

Sreerama K. Murthy, Simon Kasif, and Steven Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2:1–32, 1994. URL https://www.jair.org/index.php/jair/article/view/10121.

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios (eds.), *Computer Vision – ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pp. 143–156, Berlin, Heidelberg, 2010. Springer. doi: 10.1007/978-3-642-15561-1_11.

Tom Rainforth and Frank Wood. Canonical correlation forests. *arXiv preprint arXiv:1507.05444*, 2015. URL https://arxiv.org/abs/1507.05444.

C. Radhakrishna Rao. The utilization of multiple measurements in problems of classification. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2):159–203, 1948.

Juan J. Rodríguez, Ludmila I. Kuncheva, and Carlos J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006. doi: 10.1109/TPAMI.2006.211.

Ilya M. Sobol' and Sergei Kucherenko. Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, 79(10):3009–3017, 2009. doi: 10.1016/j.matcom.2009.01.023.

Tyler M. Tomita, James Browne, Cencheng Shen, Jaewon Chung, Jesse L. Patsolic, Benjamin Falk, Carey E. Priebe, Jason Yim, Randal Burns, Mauro Maggioni, and Joshua T. Vogelstein. Sparse projection oblique randomer forests. *Journal of Machine Learning Research*, 21(104):1–39, 2020. URL https://jmlr.org/papers/v21/18-664.html.

Shubhendu Trivedi and Jialei Wang. The expected jacobian outerproduct: Theory and empirics. *arXiv preprint arXiv:2006.03550*, 2020. URL https://arxiv.org/abs/2006.03550.

Shubhendu Trivedi, Jialei Wang, Samory Kpotufe, and Gregory Shakhnarovich. A consistent estimator of the expected gradient outerproduct. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI 2014)*, pp. 819–828, Quebec City, Canada, 2014. AUAI Press.

A Analysis and additional evaluation details

A.1 Why EJOP preconditioning helps axis–aligned trees

We recall $f : \mathbb{R}^d \to \Delta_{C-1}$, $J_f(x) = [\nabla f_1(x), \ldots, \nabla f_C(x)]$, and $H_0 = \mathbb{E}_X[J_f(X)J_f(X)^\top]$ (Eq. 1). Our method constructs $\widehat{H}_0$ and uses $\widehat{H} = \widehat{H}_0$ (optionally $+\gamma I_d$) as a global linear preconditioner; the forest is trained on $X\widehat{H}$.

Proposition A.1 (Axis–aligned ⇔ shared-oblique). For any psd H and feature index j, the axis-aligned split $\{x : (x^\top H)_j \le \tau\}$ equals the oblique half-space $\{x : x^\top H e_j \le \tau\}$ in the original coordinates.

Proof. $(x^\top H)_j = e_j^\top (x^\top H) = x^\top H e_j$. □

Proposition A.2 (First-order impurity gain aligns with EJOP). Consider binary classification ($C = 2$) and squared-loss CART. Let $u \in \mathbb{S}^{d-1}$ define a split $u^\top x \le \tau$. In a thin slab around τ, linearize f: $f(x) \approx f(\xi) + \nabla f(\xi)^\top (x - \xi)$ with $u^\top \xi = \tau$. Then the *expected* impurity decrease of the best threshold along u is, up to a positive factor independent of u, proportional to

$$u^\top H_0\, u = \mathbb{E}_X\big[(u^\top \nabla f(X))^2\big].$$

Hence directions with larger $u^\top H_0 u$ yield larger first-order gains. The multiclass case follows by summing over c via $J_f J_f^\top$.

Proof. For squared loss, the CART gain equals the between-child variance of $\mu(x) = \mathbb{E}[Y \mid X{=}x] = f(x)$. Within a thin slab, $\Delta\mu \approx (u^\top \nabla f(\xi))\, \Delta s$; the expected squared change (optimizing τ locally) is proportional to $\mathbb{E}[(u^\top \nabla f(\xi))^2]$. Density/coverage terms factor out and do not depend on u at first order. □

Corollary A.3 (Effect of using $\widehat{H} = \widehat{H}_0$). By Prop. A.1, an axis-aligned split on $(X\widehat{H})$ with index j corresponds to normal $u_j = \widehat{H} e_j$ in x-space. By Prop. A.2, its first-order score is $e_j^\top \widehat{H}^\top H_0 \widehat{H}\, e_j$. Taking $\widehat{H} \approx H_0$ amplifies coordinates whose induced normals have large EJOP scores, biasing the search toward high-sensitivity directions while keeping standard RF training.

A.2 Concentration and consistency of the EJOP estimator

We analyze $\widehat{H}_0$ under mild smoothness and boundedness assumptions. Let $X_1, \ldots, X_m \overset{\text{iid}}{\sim} P_X$ be the subsample used for probing; for each X_i we compute (central) finite-difference approximations $G_i(c) \in \mathbb{R}^d$ to $\nabla f_c(X_i)$ and form

$$\widehat{H}_0 \;=\; \frac{1}{m} \sum_{i=1}^{m} \sum_{c=1}^{C} w_{ic}\, G_i(c)\, G_i(c)^\top,$$

where the weights w_{ic} are either 1 (*unweighted* EJOP) or $\hat{p}(c \mid X_i)$ (*probability-weighted* variant). Define the population target

$$H_0^{(w)} \;=\; \mathbb{E}_X\Big[\sum_{c=1}^{C} w_c(X)\, \nabla f_c(X)\, \nabla f_c(X)^\top\Big],$$

with $w_c(X) \in \{1,\, p(c \mid X)\}$ matching the chosen variant.

Assumptions. (A1) Each f_c is C^3 with $\|\nabla f_c(x)\|_2 \le M$ and third derivatives bounded by B_3 on the support of P_X. (A2) Finite-difference step sizes satisfy $\varepsilon \to 0$ while $m \to \infty$ with $m\varepsilon^2 \to \infty$. (A3) If probability weights are used, $\sup_x |\, \hat{p}(c \mid x) - p(c \mid x)\,| \le \eta_m \to 0$. (A4) The FD evaluations are clipped to a compact domain containing the support of P_X.

Lemma A.4 (FD gradient bias). Fix a class c. Let $f_c : \mathbb{R}^d \to \mathbb{R}$ be C^3 in a neighborhood of x, and assume all third directional derivatives along the coordinate axes are bounded there: $\sup_z |\partial_j^3 f_c(z)| \le B_3$ for every j. Define the centered finite-difference estimator with step $\varepsilon > 0$ by

$$g_j^{\mathrm{FD}}(x; c) \;=\; \frac{f_c\big(x + \frac{\varepsilon}{2} e_j\big) - f_c\big(x - \frac{\varepsilon}{2} e_j\big)}{\varepsilon}.$$

Then for each coordinate j,

$$\big|\, g_j^{\mathrm{FD}}(x; c) - \partial_j f_c(x) \,\big| \;\le\; \frac{B_3}{24}\, \varepsilon^2 \;\; (\le \tfrac{B_3}{6}\, \varepsilon^2).$$

Consequently, if $\|\nabla f_c(x)\|_2 \le M$ and $G^{\mathrm{FD}}(c)$ is the vector with entries $g_j^{\mathrm{FD}}(x; c)$, then

$$\big\|\, G^{\mathrm{FD}}(c) G^{\mathrm{FD}}(c)^\top - \nabla f_c(x) \nabla f_c(x)^\top \,\big\|_2 \;\le\; \frac{B_3 \sqrt{d}}{12}\, M\, \varepsilon^2 \;+\; \frac{B_3^2 d}{576}\, \varepsilon^4.$$

Fix j and write the univariate function $g(t) := f_c(x + t e_j)$ for $t \in \mathbb{R}$. By Taylor's theorem with Lagrange remainder around $t = 0$ with $h := \varepsilon/2$,

$$g(h) = g(0) + h g'(0) + \tfrac{h^2}{2} g''(0) + \tfrac{h^3}{6} g^{(3)}(\xi_+),$$
$$g(-h) = g(0) - h g'(0) + \tfrac{h^2}{2} g''(0) - \tfrac{h^3}{6} g^{(3)}(\xi_-),$$

for some $\xi_+ \in (0, h)$ and $\xi_- \in (-h, 0)$. Subtracting and dividing by $2h$ gives

$$\frac{g(h) - g(-h)}{2h} = g'(0) + \frac{h^2}{12}\Big(g^{(3)}(\xi_+) + g^{(3)}(\xi_-)\Big).$$

Since $g'(0) = \partial_j f_c(x)$ and $g^{(3)}(t) = \partial_j^3 f_c(x + t e_j)$, our FD estimator satisfies

$$g_j^{\mathrm{FD}}(x; c) - \partial_j f_c(x) = \frac{h^2}{12}\Big(\partial_j^3 f_c(x + \xi_+ e_j) + \partial_j^3 f_c(x + \xi_- e_j)\Big).$$

Using the bound $|\partial_j^3 f_c| \le B_3$ yields $\big|g_j^{\mathrm{FD}}(x; c) - \partial_j f_c(x)\big| \le \frac{h^2}{12}(B_3 + B_3) = \frac{B_3}{6} h^2 = \frac{B_3}{24} \varepsilon^2$, proving the first claim (and the looser $\frac{B_3}{6}\varepsilon^2$ bound follows since $\frac{1}{24} \le \frac{1}{6}$).

For the matrix deviation, write

$$G^{\mathrm{FD}}(c) = \nabla f_c(x) + \delta, \qquad \delta_j := g_j^{\mathrm{FD}}(x; c) - \partial_j f_c(x),$$

so $\|\delta\|_\infty \le \frac{B_3}{24}\varepsilon^2$ and hence $\|\delta\|_2 \le \frac{B_3}{24}\sqrt{d}\,\varepsilon^2$. Then

$$G^{\mathrm{FD}}(c) G^{\mathrm{FD}}(c)^\top - \nabla f_c(x) \nabla f_c(x)^\top = \nabla f_c(x)\, \delta^\top + \delta\, \nabla f_c(x)^\top + \delta\, \delta^\top,$$

so by submultiplicativity and $\|u v^\top\|_2 = \|u\|_2 \|v\|_2$,

$$\big\|\, G^{\mathrm{FD}}(c) G^{\mathrm{FD}}(c)^\top - \nabla f_c(x) \nabla f_c(x)^\top \,\big\|_2 \le 2 \|\nabla f_c(x)\|_2\, \|\delta\|_2 + \|\delta\|_2^2 \le \frac{B_3 \sqrt{d}}{12}\, M\, \varepsilon^2 + \frac{B_3^2 d}{576}\, \varepsilon^4,$$

as claimed.

Lemma A.5 (Weight approximation error). Fix x and assume a uniform probability–weight error $\eta_m \ge 0$ so that $\big|\hat{p}(c \mid x) - p(c \mid x)\big| \le \eta_m$ for all $c \in \{1, \dots, C\}$. If $\|\nabla f_c(x)\|_2 \le M$ for all c, then

$$\left\| \sum_{c=1}^{C} \big(\hat{p}(c \mid x) - p(c \mid x)\big)\, \nabla f_c(x)\, \nabla f_c(x)^\top \right\|_2 \;\le\; \eta_m \sum_{c=1}^{C} \|\nabla f_c(x)\|_2^2 \;\le\; C M^2 \eta_m.$$

Write $a_c := \hat{p}(c \mid x) - p(c \mid x)$ and $u_c := \nabla f_c(x)$. Using the triangle inequality for the operator norm,

$$\left\| \sum_{c=1}^{C} a_c\, u_c u_c^\top \right\|_2 \;\le\; \sum_{c=1}^{C} |a_c|\, \|u_c u_c^\top\|_2.$$

For any vector u, the rank-one matrix $u u^\top$ has spectral norm $\|u u^\top\|_2 = \|u\|_2^2$ (its only nonzero eigenvalue). Hence

$$\left\| \sum_{c=1}^{C} a_c\, u_c u_c^\top \right\|_2 \;\le\; \sum_{c=1}^{C} |a_c|\, \|u_c\|_2^2 \;\le\; \eta_m \sum_{c=1}^{C} \|u_c\|_2^2,$$

because $|a_c| \le \eta_m$ for all c. Finally, $\|u_c\|_2 \le M$ gives $\sum_{c=1}^{C} \|u_c\|_2^2 \le \sum_{c=1}^{C} M^2 = C M^2$, yielding the stated bound.

B REPRODUCIBILITY AND IMPLEMENTATION DETAILS

Code and artifacts. We provide a self-contained Google drive with scripts to download datasets and run experiments at `https://drive.google.com/file/d/1d60ysqjGzQLFkl_BE8vd0lTOoj_m9MP4/view?usp=sharing`

Environment. Python 3.11; NumPy 1.26; SciPy 1.11; scikit-learn 1.4; LightGBM 4.3; CatBoost 1.2; pandas 2.2; joblib 1.3. Experiments ran on a 16-core CPU machine (no GPU used). To reduce nondeterminism across BLAS/OpenMP, we set `PYTHONHASHSEED=0`, `OMP_NUM_THREADS=1`, `MKL_NUM_THREADS=1`, and pass `random_state=seed` to learners.

Datasets and preprocessing. Continuous features: median imputation, then z-score within each training fold; categorical: one-hot encoding within fold. No test-time statistics leak from training. We clip EJOP probing points to the empirical $[q_{0.5\%}, q_{99.5\%}]$ range per feature. We evaluate on ten widely used tabular classification datasets from OpenML/UCI: *adult* (48,842×14), *bank-marketing* (41,188×20), *covertype* (581,012×54), *phoneme* (5,404×5), *electricity* (45,312×8), *satimage* (6,435×36), *spambase* (4,601×57), *magic* (19,020×10), *letter* (20,000×16), and *vehicle* (846×18), where we report $(n \times d)$ with d the *raw* feature count before one-hot encoding.

Evaluation protocol. We use 5 repeats of stratified 5-fold CV (25 fits per method per dataset). Metrics: Cohen's κ (primary), Macro-F1, and Accuracy (secondary). For cross-dataset comparisons we report average ranks; significance via Wilcoxon signed-rank vs. the top comparator with Holm correction (critical-difference diagrams shown in the appendix). Wall-clock is split into EJOP probing and final model fit; timing uses `time.perf_counter()`.

JARF specifics. Surrogate $\hat{f}$: Random Forest (50 trees, `max_features=\sqrt"`, `min_samples_leaf=1`); final forest: RF (200 trees, same defaults unless the public baseline specifies otherwise). EJOP: centered finite differences with per-feature step $\varepsilon_j = \alpha\,\mathrm{MAD}(X_{:j})/0.6745$ (default $\alpha = 0.1$); subsample size $m = \min(10{,}000,\ n)$; exclude one-hot columns from FD gradients by default (a discrete toggle variant is reported in ablations). EJOP matrix $\widehat{H}_0$ is used directly; we add γI_d with $\gamma = 10^{-3}$ for conditioning and rescale by $\mathrm{tr}(\widehat{H})/d$ to keep feature magnitudes comparable. The transformed design matrix is $X\widehat{H}$.

Determinism and variance. Tree methods can exhibit slight run-to-run variation due to threading and data-parallelism. We report means ± standard errors across repeats and publish raw per-fold outputs. Setting the environment variables above and fixing seeds reproduces our tables within reported error bars.

Licenses and data usage. We only use public datasets with permissive licenses. The repository includes per-dataset source references and license notes; any dataset requiring an external EULA is downloaded via the provider's URL with its terms unchanged.

LLM usage. All scientific content, methods, analyses, and experiments were designed and verified by the authors; LLM model was used only to aid/polish writing.