Can XGBoost Learn Ratio and Product Features From Raw Inputs Under Distribution Shift?

Anonymous Author

Abstract

Many predictive tasks in tabular data use engineered features such as ratios or products. These features can encode invariances to rescaling and unit changes. This paper studies whether XGBoost learns these structures from raw positive inputs in a way that remains stable under targeted distribution shifts.

This paper builds a synthetic benchmark of binary classification tasks. Each task is defined by a small number of latent coordinates. The coordinates are log-ratios or log-products of subsets of raw features. A task then applies a simple motif to these coordinates. Motifs include monotone coordinates, contrasts, interactions, non-monotone transforms, and gating.

This paper trains XGBoost in two modes. The raw mode uses only raw inputs. The oracle mode augments inputs with either intermediate coordinates or the final latent signal. Across experiments, the raw mode often fits training data but fails under targeted shifts that preserve the intended coordinate. The largest gaps occur on interaction motifs. These failures indicate shortcut learning in raw space rather than recovery of the intended invariant structure.

1 Introduction

XGBoost is widely used for tabular prediction. It can approximate many nonlinear functions. Approximation does not guarantee robust generalization. A model can fit training data using incidental cues. This matters under distribution shift.

Ratios and products are common engineered features. They often encode invariances. A ratio is invariant to multiplying numerator and denominator by the same constant. A product is invariant to inverse scaling that keeps the product fixed. A robust predictor should not degrade when raw magnitudes change but the true coordinate stays the same.

This paper asks a targeted question. Can XGBoost trained on raw features learn ratio or product structure in a way that remains stable under shifts that preserve that structure?

When to care. This issue matters when feature magnitudes can change without changing the intended meaning. This includes unit changes, inflation or denomination changes, sensor calibration drift, vendor and geography mix shifts, and pipeline normalization changes.

What to do in practice. If a ratio or product is known and cheap, engineer it. If the structure is unknown, use targeted perturbation checks that preserve candidate invariances. If performance is sensitive to such checks, treat the model as using magnitude shortcuts.

Related work. Tree ensembles are a standard baseline for tabular prediction. Random forests and gradient boosting are widely used due to strong accuracy and simple deployment [1, 2]. XGBoost is a common implementation and is often competitive on structured datasets [3]. These models partition the feature space with axis aligned splits. They can represent interactions, ratios, and

Benchmark pipeline.

- Sample raw features x under a regime.
- Construct coordinates u from x using log-ratios and log-products.
- Construct a task signal s by applying a simple motif to the coordinates.
- Sample labels y from a noisy link, typically Eq. (1).
- Train XGBoost on raw inputs only. Train oracle baselines that add either coordinates or the final signal.
- Evaluate under static regimes and under targeted shifts, including preserve shifts.

Figure 1: Schematic of the benchmark.

products, but the representation can be inefficient when the signal depends on coordinated behavior across multiple raw features. Practitioners often address this with feature engineering such as log transforms, ratios, and products.

Robustness under shift and invariant structure. A separate literature studies generalization under distribution shift. One line of work uses invariance across environments as a guiding principle [4, 5]. Another line emphasizes that strong training performance can coexist with unstable behavior due to spurious correlations and underspecification [6, 7]. These viewpoints motivate evaluations that apply targeted interventions that preserve intended semantics while changing nuisance variation. This paper follows that logic in a tabular setting. It uses preserve shifts that keep a latent coordinate fixed while changing raw magnitudes. It then tests whether a model trained on raw inputs recovers the invariant coordinate or relies on scale dependent shortcuts.

2 Problem setup

Let $x \in \mathbb{R}^d_{>0}$ denote raw features. A task defines a latent signal $s = f(x)$. The label is a noisy function of s. This paper uses a logistic link for most tasks:

$$\mathbb{P}(y = 1 \mid x) = \sigma\big(\beta(s - t)\big), \qquad \sigma(z) = \frac{1}{1 + e^{-z}}, \tag{1}$$

where $\beta > 0$ and t controls prevalence.

This paper compares two feature modes. The raw mode uses only x. The oracle mode augments x with features derived from the data generating process. This paper considers two oracle variants. One variant adds intermediate coordinates. One variant adds the final latent signal s.

This paper measures performance with test PRAUC. This paper also measures a paired gap:

$$\Delta\text{PRAUC} = \text{PRAUC}_{\text{oracle}} - \text{PRAUC}_{\text{raw}}. \tag{2}$$

3 Synthetic benchmark

3.1 Coordinates and task motifs

The benchmark is built in log-coordinates. This is a modeling choice. It reduces dynamic range and makes the intended invariances explicit.

The implementation uses a stabilizer that is *per coordinate*. For each coordinate, the stabilizer is set as a small fraction of a robust scale estimate, such as a median of the relevant sum or product computed on the training split. This avoids undefined logs and reduces numerical artifacts in extreme tail regimes.

This paper uses two coordinate types. The first type is a log-ratio of sums:

$$u_R(A, B) = \log\left(\frac{\sum_{i\in A} x_i}{\sum_{j\in B} x_j + \epsilon_R(A, B)}\right). \tag{3}$$

The second type is a log-product:

$$u_P(A) = \log\left(\prod_{i\in A} x_i + \epsilon_P(A)\right). \tag{4}$$

A task then applies a simple motif to one or more coordinates. The benchmark assigns each task a level label that indexes a motif family. The level label is not a formal ordering by function-class complexity. Appendix A gives representative task definitions.

Level families are:

- **Level 1 (single coordinate).** $s = u$ where u is one log-ratio or one log-product coordinate.
- **Level 2 (aggregated coordinate).** $s = u$ where u is a log-ratio of sums with larger index sets.
- **Level 3 (contrast).** $s = u_1 - u_2$ for two coordinates of the same type.
- **Level 4 (interaction).** $s = u_1u_2$ for two coordinates. This level includes ratio$\times$ratio and product$\times$product subtypes.
- **Level 5 (hybrid interaction).** $s = u_1u_2$ where one coordinate is ratio-type and one is product-type.
- **Level 6 (non-monotone shape).** $s = g(u)$ where g is non-monotone in u.
- **Level 7 (gating).** s selects between two coordinates using a raw-feature gate. This level is provisional and is excluded from headline summaries.

3.2 Distribution regimes and shifts

This paper generates features under several regimes. Some regimes are static and change tails or correlations. Other regimes impose explicit train-test shifts.

This paper includes a *preserve shift*. The preserve shift changes raw magnitudes while preserving the intended coordinate in the idealized case without stabilizers. For ratio tasks, the shift multiplies numerator and denominator features by a common factor. For product tasks, the shift applies inverse scaling that keeps the product constant. With the per-coordinate stabilizers in Eqs. (3) and (4), invariance is approximate in finite samples. The approximation is tight when the stabilizer terms are negligible relative to the corresponding sums or products. Appendix B states the preserve shift more precisely.

The preserve shift is an identifiability device. Without it, a raw-only model can score well by exploiting magnitude cues that correlate with the label under a specific regime. Those cues can disappear when units or scales change. A model that truly uses the intended coordinate should be stable under preserve shifts.

3.3 Experimental protocol

This paper uses a fixed train-validation-test split within each dataset. The main experiment uses $n = 30{,}000$ training examples and a similarly sized test set. This paper repeats each configuration over three random seeds.

3.4 Models

This paper trains XGBoost classifiers with a small set of capacity settings. The main settings vary maximum tree depth. This paper uses a validation set for early stopping. Appendix E lists the training configuration.

4 Results

All tables in this section use the baseline XGBoost configuration. Unless stated otherwise, the oracle baseline is `oracle_coords_only`. This baseline is the closest analogue to feature engineering.

Benchmark panel stability. Each configuration produces paired outcomes for raw and oracle features on the same dataset instance. This paper treats each (`task_id`, `regime_id`, `seed`, `xgb_config_id`) instance as one experimental unit and summarizes the paired effect Δ PRAUC. Percentile ranges describe heterogeneity across benchmark instances. This paper also reports the fraction of instances with Δ PRAUC > 0 as a win rate. The Wilson interval in parentheses is a binomial style uncertainty summary for that win rate. It is not a statement of statistical significance.

4.1 Why preserve shift matters

Table 1 summarizes results by regime family. Static regimes change distributional shape but do not impose an explicit train-test shift. Shift regimes do. The preserve shift family produces much larger median gaps than other regime families.

Regime family	Raw med PRAUC	Oracle med PRAUC	Median ΔPRAUC	10 to 90 pct range	Runs	Win rate ($\Delta > 0$)
tail_corr	0.137	0.167	0.007	[-0.006, 0.037]	108	76.9% (68.1–83.8)
mixture_extremes	0.278	0.303	0.016	[0.003, 0.046]	36	97.2% (85.8–99.5)
shift_naive	0.295	0.340	0.016	[-0.010, 0.095]	36	72.2% (56.0–84.2)
shift_preserve	0.126	0.352	0.146	[0.017, 0.392]	36	97.2% (85.8–99.5)

Table 1: Results by regime family for the baseline model with `oracle_coords_only`. Win rate is the fraction of runs with Δ PRAUC > 0 with a Wilson interval in parentheses.

Under preserve shifts, the paired gap is positive in 97.2% of instances (Table 1). For the preserve shift family, the leave-one-task-out median gap ranges from 0.128 to 0.157.

4.2 Main results by motif level

Table 2 reports results by motif level. It compares raw features to two oracle variants. The coordinate oracle adds intermediate coordinates. The signal oracle adds the final latent signal s.

Levels 1 to 6 are included in the headline summary. Level 7 is provisional and is summarized in Appendix A.

Win rates increase for contrast and interaction motifs. For Level 4 under the coordinate oracle, the leave-one-task-out median gap ranges from 0.012 to 0.106.

Level	Raw med	Oracle coords med	Median Δ	Win rate (Δ > 0)	Oracle signal med	Median Δ	Win rate (Δ > 0)	Runs
1	0.205	0.244	0.006	72.2% (56.0–84.2)	0.239	0.008	69.4% (53.1–82.0)	36
2	0.132	0.161	0.011	77.8% (61.9–88.3)	0.157	0.014	88.9% (74.7–95.6)	36
3	0.211	0.319	0.028	83.3% (68.1–92.1)	0.330	0.047	94.4% (81.9–98.5)	36
4	0.116	0.236	0.035	88.9% (74.7–95.6)	0.246	0.052	97.2% (85.8–99.5)	36
5	0.376	0.453	0.050	94.4% (74.2–99.0)	0.477	0.062	100.0% (82.4–100.0)	18
6	0.054	0.061	0.006	88.9% (74.7–95.6)	0.065	0.008	88.9% (74.7–95.6)	36

Table 2: Results by level for the baseline model. Oracle coords denotes `oracle_coords_only`. Oracle signal denotes `oracle_s_only`. Win rate is the fraction of runs with Δ PRAUC > 0 with a Wilson interval in parentheses.

The median gaps are small on single-coordinate motifs. The gaps are larger on contrast and interaction motifs. This pattern is consistent with the need to coordinate information across multiple raw features.

4.3 Preserve shifts expose noninvariant learning

A preserve shift should not hurt a predictor that uses the correct coordinate. However, preserve shifts can cause large failures for the raw feature model.

Table 3 lists the largest ΔPRAUC cases under preserve shifts. These cases are dominated by interaction motifs. In the most extreme case in this table, the raw feature model reaches PRAUC 0.144 while the oracle reaches PRAUC 0.903.

Task	Level	Regime	Seed	Raw PRAUC	Oracle PRAUC	ΔPRAUC
l4_product_x_product	4	shift_preserve_c5	0	0.144	0.903	0.759
l4_product_x_product	4	shift_preserve_c5	2	0.261	0.899	0.638
l5_ratio_x_product	5	shift_preserve_c5	0	0.048	0.444	0.396
l5_ratio_x_product	5	shift_preserve_c5	2	0.051	0.446	0.395
l5_ratio_x_product	5	shift_preserve_c5	1	0.066	0.455	0.390

Table 3: Top five largest gaps under preserve shifts for the baseline model with `oracle_coords_only`.

4.4 Diagnostics align with performance gaps

This paper uses diagnostics that test invariance directly. This paper also measures iso-coordinate variation.

For a ratio coordinate, this paper scales numerator and denominator features together. In the idealized case without stabilizers, this keeps the coordinate fixed. With the stabilizer $\epsilon_R(A, B)$ in Eq. (3), invariance is approximate. A perfectly invariant predictor should not change under this perturbation. This paper reports a ratio invariance error based on prediction differences under such perturbations. This paper also reports an iso variation score that measures prediction variability along iso-coordinate directions.

These diagnostics are mechanistic checks. Across regime families, the coordinate oracle has smaller invariance errors than the raw model. Appendix D reports these medians. Preserve shift is the performance stress test. Runs with larger invariance error tend to have larger ΔPRAUC.

5 Discussion and future work

This benchmark isolates a specific risk. A raw-only model can fit data using magnitude cues that are cheap for trees to exploit. Those cues can be unstable under unit and scale changes. Oracle coordinates reduce this risk because they expose the invariant structure directly.

This paper has limitations. It uses synthetic data. It uses a small set of model settings. It does not yet include a wide set of baseline learners.

This paper will be updated with additional baseline runs and uncertainty estimates. Appendix F lists concrete requested runs for the coding and research agent.

A Task definitions

This appendix gives representative task definitions. Each task uses disjoint index sets unless stated otherwise. All tasks use the coordinate definitions in Eqs. (3) and (4).

Level 1 examples

Ratio: $s = u_R(\{0,1\},\{2,3\})$. Product: $s = u_P(\{0,1\})$.

Level 2 examples

Ratio of larger sums: $s = u_R(\{0,1,2,3\},\{4,5,6,7\})$.

Level 3 examples

Contrast of ratios: $s = u_R(\{0,1\},\{2,3\}) - u_R(\{4,5\},\{6,7\})$. Contrast of products: $s = u_P(\{0,1\}) - u_P(\{2,3\})$.

Level 4 examples

Ratio×ratio: $s = u_R(\{0,1\},\{2,3\}) \cdot u_R(\{4,5\},\{6,7\})$. Product×product: $s = u_P(\{0,1\}) \cdot u_P(\{2,3\})$.

Level 5 examples

Hybrid interaction: $s = u_R(\{0,1\},\{2,3\}) \cdot u_P(\{4,5\})$.

Level 6 examples

Band-pass shape on a coordinate:

$$s = \exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right), \qquad u = u_R(\{0,1\},\{2,3\}). \tag{5}$$

This creates a non-monotone relationship between u and the label.

Level 7 examples (provisional)

The implementation uses a raw feature as the gate. Let x_g denote a positive raw feature and let τ denote a threshold, such as the training median of x_g. Define

$$s = \begin{cases} u_a, & x_g > \tau, \\ u_b, & x_g \leq \tau, \end{cases} \qquad x_g = x_4, \quad u_a = u_R(\{0,1\},\{2,3\}), \quad u_b = u_P(\{5,6\}). \tag{6}$$

This level is excluded from headline summaries until the gate definition and shifts are fully aligned.

B Preserve shift construction

This appendix states the preserve shift at the coordinate level.

For a ratio coordinate $u_R(A,B)$, define a positive scalar c. Apply $x_i \leftarrow cx_i$ for $i \in A \cup B$. If $\epsilon_R(A,B) = 0$, this leaves $u_R(A,B)$ unchanged. With $\epsilon_R(A,B) > 0$ in the denominator, the transformed coordinate is

$$u'_R(A,B) = \log\left(\frac{c\sum_{i\in A} x_i}{c\sum_{j\in B} x_j + \epsilon_R(A,B)}\right) = u_R(A,B) + \log\left(\frac{\sum_{j\in B} x_j + \epsilon_R(A,B)}{\sum_{j\in B} x_j + \epsilon_R(A,B)/c}\right). \tag{7}$$

Therefore, the preserve shift is approximately invariance-preserving when $c\sum_{j\in B} x_j \gg \epsilon_R(A,B)$.

For a product coordinate $u_P(A)$, preserve shifts use inverse scaling. For example, for $A = \{i,j\}$ apply $x_i \leftarrow cx_i$ and $x_j \leftarrow x_j/c$. This keeps $\prod_{k\in A} x_k$ fixed. With $\epsilon_P(A) > 0$, the coordinate is approximately preserved when the product dominates $\epsilon_P(A)$.

C Level by regime-family breakdown

Table 4 reports median ΔPRAUC by level and regime family for the baseline model with `oracle_coords_only`. This table shows that the preserve shift family is the dominant source of large gaps.

Level	tail_corr	mixture_extremes	shift_naive	shift_preserve
1	0.002	0.011	-0.003	0.190
2	0.005	0.018	0.005	0.096
3	0.018	0.034	0.019	0.178
4	0.013	0.057	0.087	0.190
5	0.028	0.041	0.060	0.395
6	0.006	0.004	0.016	0.017

Table 4: Median ΔPRAUC by level and regime family for `oracle_coords_only`.

D Invariance diagnostics by regime family

This appendix reports invariance diagnostics aggregated by regime family for the baseline model with `oracle_coords_only`. The ratio invariance error is the mean absolute prediction change under joint scaling of ratio components. The product invariance error is the mean absolute prediction change under compensating product scalings.

Regime family	Ratio inv err (raw)	Ratio inv err (oracle)	Product inv err (raw)	Product inv err (oracle)
tail_corr	0.0105	0.0023	0.0113	0.0036
mixture_extremes	0.0070	0.0018	0.0118	0.0037
shift_naive	0.0109	0.0037	0.0167	0.0056
shift_preserve	0.0113	0.0020	0.0175	0.0035

Table 5: Median invariance errors by regime family for the baseline model with `oracle_coords_only`. Smaller values indicate better invariance.

E Training configuration

This paper uses standard XGBoost binary classification training. This paper uses early stopping on a validation split. This paper varies maximum depth in the main grid. Other settings follow common defaults. The experiment harness records full configurations alongside each run.

F Requested runs for the next iteration

This checklist is written for the coding and research agent.

- Add a baseline that trains the raw-only model on $\log(x + \epsilon)$ features. Report Table 1 and Table 2 for this baseline on the preserve shift family. This isolates whether failures are driven by dynamic range in raw x.
- Report ROC-AUC alongside PRAUC for all main tables. Keep PRAUC as the primary metric. Use ROC-AUC as a stability check.
- Add a seed expansion on a smaller subset. Use at least 10 seeds for a reduced grid that focuses on Levels 3 to 5 and the preserve shift family. Report uncertainty bands for ΔPRAUC.
- Add an explicit sweep that varies correlation while holding tail-heaviness fixed. Add a separate sweep that varies tail-heaviness while holding correlation fixed. This removes confounding across regimes.
- Add a dataset size sweep. Use at least three training sizes. Keep the same test set. This estimates sample complexity for learning invariant coordinates.
- Add baseline learners on oracle coordinates. Include a linear model on the coordinate features. Include a small neural net baseline. Keep capacity roughly comparable across models.
- Revisit Level 7 gating. Ensure that the gate definition and the preserve shift are aligned for that level. Delay strong claims about gating until this check is complete.

References

[1] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[2] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[3] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[4] J. Peters, P. Bühlmann, and N. Meinshausen. Causal inference using invariant prediction: Identification and confidence intervals. *Journal of the Royal Statistical Society: Series B*, 78(5):947–1012, 2016.

[5] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz. Invariant risk minimization. In *International Conference on Learning Representations*, 2019.

[6] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 2020.

[7] A. D'Amour, K. Heller, D. Moldovan, B. Adlam, B. Alipanahi, A. Beutel, and others. Underspecification presents challenges for credibility in modern machine learning. In *Advances in Neural Information Processing Systems*, 2020.