FACTORIZED CHAIN UTILITY: A CALIBRATED DIVERGENCE DIAGNOSTIC FOR SYNTHETIC TABULAR DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

Synthetic tabular data helps share information while protecting commercial or personal interests, so its value depends on how well it replicates the joint distribution of the real data. Machine-learning utility measures whether a model trained on synthetic data predicts a target variable on real data. Because this score depends only on a single conditional distribution, a synthetic table can score well on this measure while failing to preserve important relationships among the other variables. We propose Factorized Chain Utility (FCU), a diagnostic for synthetic tables based on a sparse chain of conditional prediction tasks. FCU uses proper log scores to quantify distortion in selected conditional relationships. The selected conditionals multiply to a distribution, unlike the leave-one-out conditionals used by prior all-column metrics. This gives FCU's aggregate score a well-defined divergence interpretation for the selected chain. We report a pruning deficit that quantifies the dependence information omitted by the sparse factorization, and per-factor scores identify the relationships responsible for degraded utility. Controlled validation shows that FCU responds to dependency loss even when marginal distributions are preserved. Across diverse datasets and generators, FCU further separates synthetic datasets that appear comparable under single-task utility or discriminator-based evaluations, providing a principled generator ranking with actionable diagnostics.

1 INTRODUCTION

Synthetic tabular data can ease privacy and access constraints, augment datasets, and support data imputation while preserving the structure needed for downstream modeling (Jordon et al., 2022; Dankar & Ibrahim, 2021). A generator aims to approximate the joint distribution over a table's columns. In practice, generators are often judged by machine-learning utility (MLU). The dominant protocol, Train-on-Synthetic, Test-on-Real (TSTR) (Esteban et al., 2017), selects one target column, trains a predictor on synthetic rows, and compares its performance on real rows with that of the same predictor trained on real data.

The TSTR protocol answers a narrower question than the one it is often used for. Let Y denote the target and Z the remaining columns. In the population limit, TSTR depends on the synthetic distribution Q only through $Q(Y \mid Z)$ (Proposition 1). It cannot detect distortion in $Q(Z)$ or in dependencies within Z. Under the population assumptions of Proposition 1, changing the model, loss, or target cannot make a single TSTR task assess the remaining conditional structure.

A natural response to the limits of single-task utility is to score every column in turn. Prior all-column metrics, including the global utility metric of Jiang et al. (2026), predict each column from all the others (El Emam et al., 2022; Dankar et al., 2022; Yang et al., 2024). They therefore assess utility across many conditional prediction tasks rather than one. However, leave-one-out conditionals do not generally multiply to a joint distribution. Their aggregate consequently has no direct divergence interpretation, can grow with table width, and does not state which dependence relationships are left outside the evaluation (Section 5.2).

We propose *Factorized Chain Utility* (FCU), which scores a sparse chain of conditional prediction tasks whose factors define a distribution. For each factor, FCU compares the log loss of a model trained on synthetic data with that of the same model trained on real data. For a fixed chain, their sum has a stated target: the divergence from the real distribution to the chain projection of the synthetic one, less the dependence the chain omits (Theorem 1). Compared with all-column metrics, FCU

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

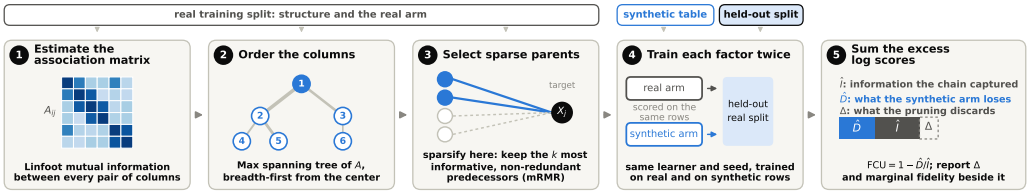


Figure 1: FCU computation procedure. Using only real training rows: (1) estimate pairwise associations; (2) order columns along their strongest relationships; and (3) retain a few informative, non-redundant earlier columns as parents. For each resulting conditional task, (4) train the same model on real and synthetic rows and evaluate both on held-out real rows. (5) Their summed excess log score is $\widehat{D}$, the conditional information lost by the synthetic table. After clipping the summed excess to $[0, \widehat{I}]$, FCU is $\mathrm{FCU} = 1 - \mathrm{clip}(\widehat{D}, 0, \widehat{I})/\widehat{I}$. The dependence omitted by pruning is Δ; we report it, when available, together with marginal fidelity MF.

conditions on one or two parents per factor instead of every column, its sum can be checked against that target, and it names both the relationships a generator loses and the dependence it did not score.

Contributions.

- **A utility score over a chain of conditional tasks.** FCU selects a sparse chain of conditional prediction tasks from the real data and, for each task, compares the same model trained on synthetic rows and on real rows. The chain is chosen so that its conditionals multiply to a distribution. Single-task utility scores one conditional; prior all-column metrics score tasks that do not form a distribution.
- **A score with a divergence interpretation.** For categorical factors, the summed excess log score equals the divergence from the real distribution to the chain projection of the synthetic one, minus the dependence the chain omits (Theorem 1). FCU reports that omitted dependence as a pruning deficit, and reports each factor's score, so a low value points to the relationships that were lost.
- **Validation and a benchmark.** We test FCU where the answer is known: controlled corruptions of real tables, constructions that preserve only the benchmark target, and distribution families with known divergences, where the all-column sum has no stated target. We then benchmark twelve mixed-type tables and eleven generators, and show that FCU separates generators that single-task utility or a real-versus-synthetic discriminator treats as equal, identifies the relationships responsible, and predicts utility on columns it did not score.

2 Preliminaries: From One Task to a Chain

Let $X = (X_1, \ldots, X_d)$ denote a random vector from real-data distribution P, where each X_j is numeric or categorical, and let $\tilde{X}$ denote a random vector from synthetic distribution Q. We ask how well Q approximates P as a distribution.

A common method to measure this approximation is the TSTR machine-learning utility metric. TSTR chooses one column X_j as the target Y and uses the remaining columns $Z = X_{-j}$ as predictors. It trains a predictor $f_{\hat{\theta}(Q)}$ on synthetic samples $(\tilde{Z}, \tilde{Y}) \sim Q$ and evaluates the predictor on real-data samples:

$$U_{\mathrm{TSTR}}(Q; P) := \mathbb{E}_{(Z,Y)\sim P}\Big[\ell\big(f_{\hat{\theta}(Q)}(Z),\, Y\big)\Big]. \tag{1}$$

Proposition 1 (Single-task utility depends on one conditional)**.** *Assume that the model class is rich enough for population training under Q to recover the Bayes predictor of Y given Z, and that training attains this minimizer. Let the evaluation score depend only on the joint distribution of the predictor and Y under P, as for a proper loss or AUC. If Q_1 and Q_2 imply the same distribution of Y given $Z = z$ for every z in the support of $P(Z)$, then*

$$U_{\mathrm{TSTR}}(Q_1; P) = U_{\mathrm{TSTR}}(Q_2; P).$$

Thus, in this population setting, TSTR depends only on the synthetic conditional $Q(Y \mid Z)$ over the real-data support; it cannot distinguish changes in $Q(Z)$ or dependencies internal to Z.

The proof (Appendix A) shows that both generators yield the same Bayes predictor. The statement is for TSTR; TRTS is not covered (Remark 1). With finitely many synthetic rows, $Q(Z)$ affects where the training examples fall, so the invariance holds only approximately. Appendix A.2 discusses that regime and Section 5.1 measures it.

One way to address this issue is to aggregate multiple machine-learning utility scores across many columns. All-column machine-learning utility metrics (allMLU) combine results from scoring every column given *all the others* (El Emam et al., 2022; Dankar et al., 2022; Yang et al., 2024; Jiang et al., 2026). However, these leave-one-out conditionals observe overlapping information. For example, they double-count pairwise dependence: for a dependent pair relative to the product of its marginals, the divergence is $I(X;Y)$, but the leave-one-out sum is $2\,I(X;Y)$ (Appendix C.3). Also, each task uses $d-1$ predictors, which is inefficient.

The chain rule decomposition of joint distributions suggests a more natural set of machine-learning utility tasks. Under any fixed ordering, the chain rule writes the joint distribution as $P(X_1,\dots,X_d) = P(X_1)\prod_{j=2}^{d} P(X_j \mid X_1,\dots,X_{j-1})$. If Q matches P on the marginal $P(X_1)$ (the *root* marginal) and on every conditional factor, then the joint distributions are the same. This suggests a utility metric that "sees" the joint distribution: attach a supervised prediction task to each conditional, and measure how well the synthetic data support learning the prediction rule of the real data.

3 Factorized Chain Utility

FCU evaluates a sparse, real-data-defined set of conditional prediction tasks (Figure 1). It compares identical real- and synthetic-trained predictors on the same held-out real rows. All information quantities and log-score differences below are measured in nats. Section 3.1 states what the summed excess estimates, Section 3.2 how the chain is chosen from the real data, and Section 3.2.2 the plug-in estimator and what is reported beside it.

3.1 Population target and chain identity

Fix an ordering of the columns and, for convenience, relabel them $X_1,\dots,X_d$. Let $\mathrm{Pa}(X_j) \subseteq \{X_1,\dots,X_{j-1}\}$ be the selected parents of X_j, let

$$R_j := \{X_1,\dots,X_{j-1}\} \setminus \mathrm{Pa}(X_j), \qquad \mathcal{C} := \{j : \mathrm{Pa}(X_j) \neq \emptyset\},$$

and write $\mathcal{S} := (\pi, \mathrm{Pa})$ for the resulting sparse chain. The conditional discrepancy scored by FCU and the information captured by the chain are

$$D_{\mathcal{C}}(P,Q) := \sum_{j\in\mathcal{C}} \mathbb{E}_{X_{\mathrm{Pa}(j)}\sim P}\left[D_{\mathrm{KL}}\big(P(X_j \mid X_{\mathrm{Pa}(j)}) \,\big\|\, Q(X_j \mid X_{\mathrm{Pa}(j)})\big)\right], \tag{2}$$

$$I(P) := \sum_{j\in\mathcal{C}} I_P\big(X_j; \mathrm{Pa}(X_j)\big). \tag{3}$$

The population FCU score is

$$\mathrm{FCU}^{\star}(P,Q) := 1 - \frac{\min\{D_{\mathcal{C}}(P,Q),\, I(P)\}}{I(P)}, \qquad I(P) > 0. \tag{4}$$

Thus, FCU normalizes discrepancy in the selected synthetic conditionals by the dependence information captured by those same tasks.

For roots, set $\mathrm{Pa}(X_1) = \emptyset$ and $Q(x_1 \mid x_{\mathrm{Pa}(1)}) := Q(x_1)$, and define the chain projection

$$Q^{\mathrm{Pa}}(x) := \prod_{j=1}^{d} Q\big(x_j \mid x_{\mathrm{Pa}(j)}\big). \tag{5}$$

Theorem 1 (Chain-projection identity). *Assume $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) < \infty$, and define*

$$\Delta(P) := \sum_{j=1}^{d} I_P\big(X_j; R_j \mid \mathrm{Pa}(X_j)\big), \tag{6}$$

$$D(P,Q) := \sum_{j=1}^{d} \mathbb{E}_{X_{\mathrm{Pa}(j)} \sim P}\left[D_{\mathrm{KL}}\big(P(X_j \mid X_{\mathrm{Pa}(j)}) \,\big\|\, Q(X_j \mid X_{\mathrm{Pa}(j)})\big)\right]. \tag{7}$$

Then

$$D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) = \Delta(P) + D(P,Q). \tag{8}$$

Moreover,

$$\Delta(P) = \min_{\widetilde{Q}:\, \widetilde{Q}(x) = \prod_{j=1}^{d} \widetilde{Q}(x_j \mid x_{\mathrm{Pa}(j)})} D_{\mathrm{KL}}(P \,\|\, \widetilde{Q}). \tag{9}$$

The pruning deficit $\Delta(P)$ is therefore the dependence in P omitted by the fixed selected chain, rather than the cost of the best sparse structure over all orderings and parent sets (Cover & Thomas, 2006; Koller & Friedman, 2009). The minimizing distribution retains P's conditionals on the selected parent sets. If Q factorizes over the chain, then $Q^{\mathrm{Pa}} = Q$.

Note the difference between the definitions in (3) (7). The full discrepancy of the joint distribution is defined by summing over all factors of the chain, as in $D(P,Q)$. However, we prefer to separate the accuracy in the conditional distributions from that of the marginals. We therefore discard the root marginals (variables without parents) from the FCU computations and report all marginal fidelity values as a separate statistic.

Let $B(P,Q)$ denote the summed marginal divergences of the root variables. Then

$$D_{\mathcal{C}}(P,Q) = D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P) - B(P,Q). \tag{10}$$

Hence FCU targets conditional mismatch along the selected chain, while separately exposing omitted dependence through $\Delta(P)$.

Corollary 1 (Excess log-loss risk). *Let*

$$\mathcal{R}_j(q) := \mathbb{E}_P\big[-\log q\big(X_j \mid X_{\mathrm{Pa}(j)}\big)\big]$$

be the real-data log-loss risk for factor j. If real- and synthetic-trained predictors q_j^{real} and q_j^{syn} attain their Bayes predictors, then

$$\mathcal{R}_j(q_j^{\mathrm{syn}}) - \mathcal{R}_j(q_j^{\mathrm{real}}) = \mathbb{E}_{X_{\mathrm{Pa}(j)} \sim P}\left[D_{\mathrm{KL}}\big(P(X_j \mid X_{\mathrm{Pa}(j)}) \,\big\|\, Q(X_j \mid X_{\mathrm{Pa}(j)})\big)\right]. \tag{11}$$

A predictor that uses the marginal $P(X_j)$ for each prediction problem, ignoring the parents, has excess risk $I_P(X_j; \mathrm{Pa}(X_j))$ over the real arm. It will therefore have a population $\mathrm{FCU}^\star$ value of 0.

3.2 Estimation of FCU

3.2.1 Choice of chain and parent sets

We estimate all structure on the real training split. Pairwise mutual information is estimated with type-appropriate methods and mapped to $A \in [0,1]^{d \times d}$ using the Linfoot coefficient (Linfoot, 1957):

$$A_{jk} = \sqrt{1 - e^{-2I(X_j; X_k)}}, \tag{12}$$

which converts mutual information to a unit-free correlation-like scale so that numeric and categorical column pairs are directly comparable.

A breadth-first traversal of the maximum spanning tree of A, rooted at its most central node, defines the ordering. With mutual-information weights this tree is the Chow–Liu tree (Chow & Liu, 1968), which solves

$$\max_{\text{spanning tree } T} \sum_{(j,k) \in T} I(X_j; X_k), \tag{13}$$

placing the most strongly associated column pairs early in the chain so that each factor conditions on an informative predecessor. Its KL-optimality does not extend to FCU's directed, pruned chain (Appendix B.1).

For each factor we retain at most k earlier columns using max-relevance–min-redundancy (Peng et al., 2005): starting from the empty set S, we repeatedly add the candidate X_m that maximizes

$$A_{jm} - \lambda \cdot \mathrm{mean}_{s \in S} A_{ms}, \tag{14}$$

which rewards relevance to the target and penalizes overlap with parents already selected. Under adaptive k, parents are added only while a probe model improves held-out utility by more than δ (Appendix B.1). This yields narrower, more stable prediction tasks at the cost of potentially missing higher-order interactions (Section 5.2).

3.2.2 Estimation procedure

For each $j \in \mathcal{C}$, we fit $q_j(x_j \mid x_{\mathrm{Pa}(X_j)})$ using identical model classes, seeds, and budgets on real and synthetic training data, then evaluate both on the same held-out real split. Let ℓ_j^{real} and ℓ_j^{syn} be their mean losses, and let ℓ_j^{ref} be the loss of a predictor using only the real marginal of X_j.

For categorical targets, the loss is negative log probability.[1] For numeric targets, we use $\frac{1}{2}\log(2\pi e\,\mathrm{MSE})$ after mapping targets to normal scores with the real-training empirical CDF. This conditional-mean surrogate detects shifts in the conditional mean but not changes in conditional spread. The categorical identity in Corollary 1 is exact; the numeric version is approximate.

Define

$$\widehat{I} = \sum_{j \in \mathcal{C}} \big(\ell_j^{\mathrm{ref}} - \ell_j^{\mathrm{real}}\big), \qquad \widehat{D} = \sum_{j \in \mathcal{C}} \big(\ell_j^{\mathrm{syn}} - \ell_j^{\mathrm{real}}\big), \qquad \mathrm{FCU} = 1 - \frac{\mathrm{clip}(\widehat{D}, 0, \widehat{I})}{\widehat{I}}, \quad \widehat{I} > 0. \tag{15}$$

Here $\widehat{I}$ estimates captured real-data predictive gain and $\widehat{D}$ estimates synthetic-versus-real conditional excess. Clipping applies to their sums, not individual factors, allowing factor-level noise to offset. If $\widehat{I} \le 0$, FCU is undefined; if $\widehat{D} \ge \widehat{I}$, FCU is zero, while the unbounded $\widehat{D}$ still distinguishes generators.

3.3 Complementary statistics

FCU measures conditional utility along the selected chain, not discrepancies outside its tasks or privacy and memorization. We therefore report FCU with $(\widehat{D}, \widehat{I})$, MF, and per-factor results.

Marginal fidelity. FCU excludes root marginals, so we report MF separately: the mean per-column marginal agreement, measured as one minus total variation for categorical variables and one minus Kolmogorov–Smirnov distance for numeric variables (Appendix B.4).

Pruning diagnostic. When P is known, $\Delta(P)$ is exact. On real tables, we instead report the full-versus-pruned gain $\hat{G}$: real-arm information under the all-predecessors chain minus that under the pruned chain. It is a diagnostic, not an error-guaranteed estimator of $\Delta(P)$.

4 Evaluation question and protocol

FCU makes three claims. First, it should detect degradation of selected conditional structure that marginal fidelity or a single downstream task can miss. Second, its unbounded summed excess $\widehat{D}$ should track the selected conditional discrepancy $D_{\mathcal{C}}(P, Q)$ of Theorem 1 when that target is known. Third, on realistic synthetic tables, FCU should distinguish generators that standard utility or fidelity measures treat as similar, while localizing the conditional relationships responsible.

[1] Each factor uses at most $C_{\max}$ classes, merging rare levels; categorical predictions are calibrated toward each arm's training marginal (Appendices B.3 and B.2).

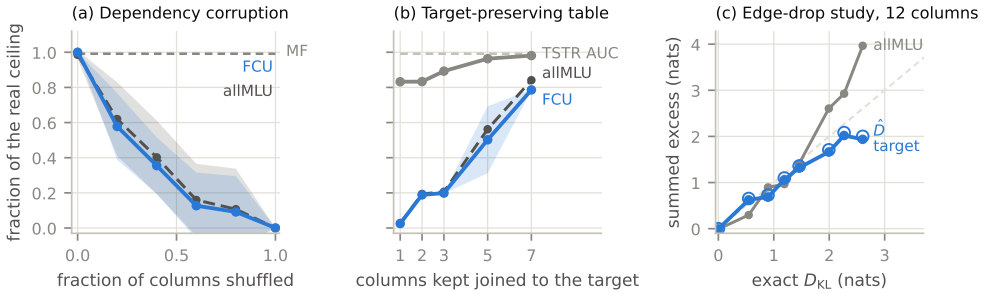


Figure 2: Controlled tests of conditional sensitivity and divergence calibration. (Sections 5.1 and 5.2). FCU (blue) is compared to other utility or fidelity measures: marginal fidelity (MF, light-grey), allMLU (Jiang et al., 2026) and TSTR AUC (Esteban et al., 2017). See Section 4. **(a)** On ADULT_CENSUS, a growing share of columns is permuted independently, which keeps every marginal and destroys the dependence. FCU and allMLU fall from 1 to 0; MF does not move. **(b)** On the same table, the benchmark target is kept joined to its m strongest associates and every other column is permuted. Single-task utility (TSTR-AUC) returns to within two points of its train-on-real ceiling; FCU and allMLU do not. **(c)** In a twelve-column network with exact divergences, generators drop the m strongest dependencies. FCU divergence, $\widehat{D}$, tracks its stated target (hollow markers). allMLU has no target and returns wide, 0.54 to 1.52 times the divergence.

We evaluate these claims in two stages. Section 5 tests the measurement in controlled settings: dependency corruption with preserved marginals, target-preserving constructions, memorization controls, and families with known targets. Section 6 then tests whether FCU changes practical generator evaluation: whether it separates generators matched on single-task utility, transfers to unscored conditional tasks, and complements broader utility, fidelity, and disclosure diagnostics.

Our benchmark contains twelve mixed-type real tables (5k–671k rows and 11–43 columns), eleven tabular generators, and SMOTENC as an oversampling control. Every generator is trained on the same real training split and sampled with three seeds. FCU structure and both prediction arms are fit without access to held-out real test rows; the arms use the same LightGBM learner, seed, and training size. Crossing three synthetic seeds with three evaluator splits gives nine runs per dataset-generator cell. For each cell, we report FCU, its unbounded numerator $\widehat{D}$, normalizer $\widehat{I}$, marginal fidelity, and per-factor results; log-score and information quantities are in nats. Nearest-record and memorization diagnostics are reported separately.

We call all non-FCU measures *comparators*. They include single-task utility and *all-model machine-learning utility* (allMLU), which predicts each column from all other columns and aggregates the resulting leave-one-out tasks. Thus, allMLU offers broader conditional coverage, but its tasks do not generally define a joint factorization or a chain-projection-divergence target. Other comparators measure real-versus-synthetic discrimination, association error, sample-level fidelity, and nearest-record disclosure risk. Definitions are in Appendix B.4; the full protocol is in Appendix F.3.

5 Does FCU measure its stated quantity?

Before ranking generators, we test the first two claims of Section 4: that FCU moves when the conditional relationships on its chain are damaged, and only then, and that $\widehat{D}$ is the quantity Theorem 1 names wherever it can be computed.

5.1 Behavioral checks

We first test what moves the score, on real tables with variants whose answer is known in advance: a held-out subset of real rows scored as if it were synthetic (nothing is wrong); tables with a growing share of columns permuted independently (every marginal kept, the dependence destroyed); a table that keeps the benchmark target joined to its strongest associates and scrambles the rest (only the designated task survives); and a verbatim copy of the training rows (Table 3; Figure 2a,b).

A held-out real-data slice, treated as synthetic data, scores near the ceiling on every table. Thus, FCU does not penalize ordinary sampling differences between two independent samples from the same

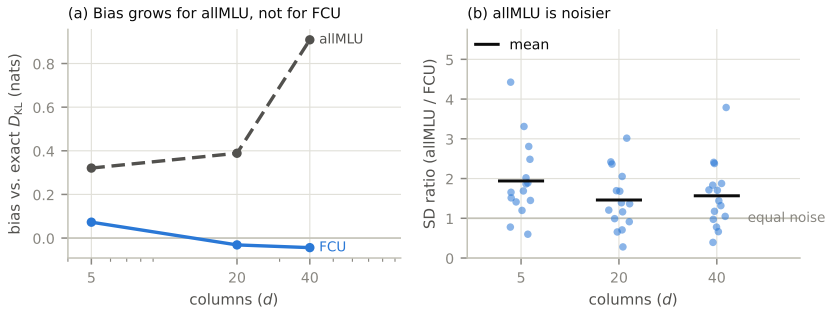


Figure 3: FCU vs. allMLU under finite-sample noise, $d \in \{5, 20, 40\}$, on a wide Gaussian copula family (Appendix C.3). **(a)** Bias against the exact divergence at $n_{\text{train}} = 4{,}000$: allMLU's grows with d; $\widehat{D}$'s does not. **(b)** The ratio of allMLU's to $\widehat{D}$'s standard deviation across seeds, one point per (copula rung, training size) cell (15 per d); allMLU is noisier on average at every d.

real-data distribution. In the results, we find that FCU behaves as expected. Permuting columns separates the two fidelity measures: marginal fidelity does not move, while FCU falls from one to zero as the dependence goes (Figure 2a). The target-preserving table separates FCU from single-task utility in the other direction. On ADULT_CENSUS, once the target keeps its seven strongest associates, the benchmark task is within two points of its train-on-real ceiling, yet a fifth of the captured conditional information is still missing, and this is identified by the FCU (Figure 2b), as predicted by Proposition 1.

As expected, a verbatim copy of the training rows scores FCU $= 1$ on every tested table. FCU measures whether its selected conditionals can be learned from the synthetic rows, not where the rows came from. We report copying diagnostics alongside FCU (Appendix D.8).

allMLU passes all of these checks as well (Appendix C). They do not separate the two metrics; the calibration experiments below do.

5.2 Calibration and coverage

We next test whether $\widehat{D}$ estimates its stated target, on Gaussian, discrete, and mixed-type families of five to forty columns where the answer is exact (Appendix C.2). For each family we build a *ladder* of synthetic distributions that move away from P in known steps, so the divergence and the chain target are known at every rung; the evaluator sees only samples and learns the structure as it would on a real table.

On every family but one, $\widehat{D}$ is within 0.07 of its target and orders the rungs almost perfectly (Table 4). The exception is the ladder that shifts the mean of a numeric column, which fails as Section 3.2.2 predicts: the numeric score ranks the shifts correctly but understates large ones, and it cannot see a change in spread alone (Table 23). Both are limits of the numeric score, not of the identity (Appendix C.3).

Calibration applies to the selected chain, not to the full joint; two experiments measure the difference. The first tests a dependence with no pairwise trace, using a parity relation where a column depends on several others jointly but on none of them alone (Appendix C.4). The chain omits almost all of that dependence, and a generator that drops it scores as highly as a faithful sample. FCU cannot see a failure outside its chain, but the deficit reports the omission in every such cell, so the two are reported together.

The *edge-drop study* grades the same effect on a twelve-column network with 22 known dependencies, removed one at a time and the m strongest together. Each removal is a generator that fails on exactly that dependence and nowhere else, with a known divergence. Where the chain covers the removed dependence, $\widehat{D}$ registers it in full; where it does not, about a quarter shows through, and the deficit counts the rest (Table 7; Appendix C.5). allMLU ranks these generators correctly but has no target to be checked against: its excess reads from half to one and a half times the divergence, while $\widehat{D}$ stays within 0.07 of its target on every rung (Figure 2c). The gap grows with table width and does not shrink with more training data (Figure 3; Appendix C.3).

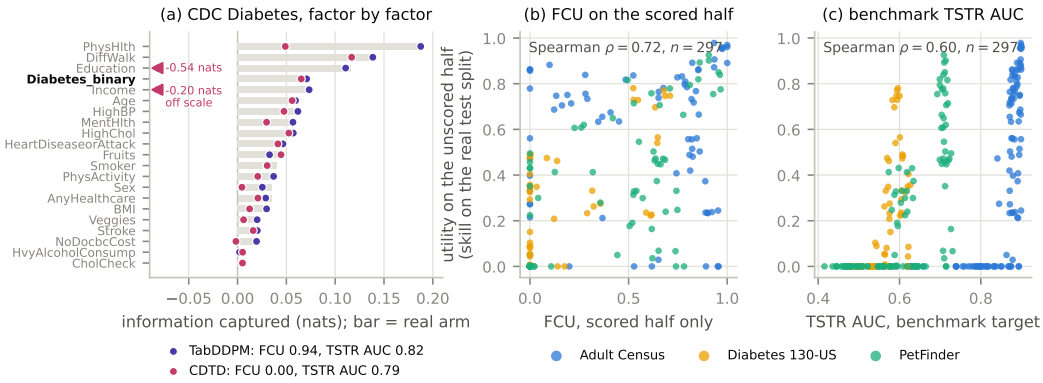


Figure 4: **(a)** Generators can have similar utility for the benchmark target variable, while differing on conditional relationships. On CDC_DIABETES, TABDDPM and CDTD have similar TSTR AUC, but FCU identifies lost signal in `Education` and `Income`. Bars show information captured by the real arm; synthetic-arm values below zero are worse than marginal prediction. **(b,c)** FCU computed on one half of a table better predicts utility on held-out columns than benchmark TSTR AUC.

The validation shows that FCU responds to lost selected dependencies even when marginals or a single task are preserved. Where the target is known, the summed excess tracks the selected-chain discrepancy; incomplete coverage and numeric-score limits are reported separately.

6 BENCHMARK

We now test the third claim: whether FCU separates generators that standard metrics call similar, predicts utility on tasks it never scored, and tells a practitioner more than a rank.

6.1 BEYOND SINGLE-TASK UTILITY

Figure 4a shows one pair in detail. On CDC_DIABETES, TABDDPM and CDTD are matched on the benchmark task: their TSTR AUCs differ by three hundredths and both reproduce the marginals well, so the standard protocol would accept either. FCU is 0.94 for one and 0.00 for the other, and the factor scores say why: from CDTD's rows a model learns nothing about `Education` or `Income`, two columns the benchmark target never tests.

The same gap appears across the grid (Table 25): of its 656 within-table generator pairs, 168 agree on TSTR AUC to within 0.03, yet their FCU differs by a median of 0.15 and by as much as 0.94 (Appendix D.2; Figure 6). Agreement on the designated task leaves most of the conditional structure unexamined. The picture is unchanged when FCU is rebuilt from the categorical factors alone, where Theorem 1 is exact (Appendix D.5).

6.2 RANKING, TRANSFER, AND ROBUSTNESS

Table 26 ranks the generators within each table with paired intervals; Table 20 gives simultaneous rank sets over the corpus (Neuhof & Benjamini, 2024). There is no universal winner: TABDIFF, FORESTDIFFUSION, CDTD, and TABSYN share the top set and change order from table to table; CDTD is confidently first on two tables and near zero on three.

We test whether the ranking carries to tasks FCU never scored by splitting each table's columns into two halves at random, computing FCU on one half, and measuring conditional utility on the other. FCU on the scored half predicts utility on the other half better than the benchmark task does (Spearman 0.72 against 0.60 over 297 runs; Figure 4b,c), though not on every table (Appendix D.5).

allMLU predicts the held-out half better still, since its tasks are the held-out kind. Where both are computed, they agree on the ranking ($\rho = 0.98$) and on the leading generator of every table (Table 13), differing in cost and interpretation rather than in ranking: allMLU conditions on three to thirty times more columns (median ten, Table 14) and reports a sum with no stated target, while FCU conditions on one or two parents per factor, has the target of Theorem 1, and reports the dependence it left out.

A classifier trained to tell real rows from synthetic ones agrees with FCU where it can still tell generators apart, but on two cells in five it is saturated (AUC ≥ 0.99) and FCU still separates those cells (Figure 7; Appendix D.7). FCU does not measure copying: the oversampling control leads four tables by FCU while 12.7% of its rows lie closer to a training row than fresh real rows do, against 5.2% for fresh real rows, so the leaderboard must be read beside these diagnostics (Appendix D.8).

The full-versus-pruned gain $\hat{G}$ of Section 3.2.2 is negative on three of four tables: the extra parents cost more to estimate than they add. On BEIJING_PM25 it is positive, also the table where FCU and allMLU agree least (Appendix D.4). Implementation choices do not drive the rankings: alternative association measures, orderings, parent budgets, and tree learners leave them essentially unchanged ($\rho \geq 0.99$ on two representative tables), while a linear learner, the earlier per-factor skill-score aggregation, and the binned numeric score reorder cells more ($\rho = 0.91$ to 0.96; Appendix D.6).

7 Related work

Synthetic tabular generators include copula models (Patki et al., 2016), GANs (Xu et al., 2019), and autoencoder, transformer, and diffusion models (Borisov et al., 2022; Kotelnikov et al., 2023; Kim et al., 2022; Lee et al., 2023; Zhang et al., 2023; Jolicoeur-Martineau et al., 2024). Their evaluation commonly combines marginal and pairwise fidelity, sample-level similarity, discrimination, and downstream utility (Jordon et al., 2022; Patki et al., 2016; Qian et al., 2023; Dankar & Ibrahim, 2021; Alaa et al., 2022). Single-task TSTR (Esteban et al., 2017) remains common, but evaluates only one conditional relationship. Appendix G gives an extended survey.

The closest approaches extend utility to many conditional prediction tasks: the all-models tests of El Emam et al. (2022) and Dankar et al. (2022), the leave-one-out conditional hierarchy of Yang et al. (2024), and the *global utility* metric allMLU of Jiang et al. (2026), which applies when no ground-truth causal graph is available. These methods share FCU's premise that one target is insufficient. They typically predict each column from all remaining columns, maximizing conditional-task coverage but creating wide prediction problems whose leave-one-out conditionals do not, in general, define a joint distribution.

allMLU is the right tool when the risk is a dependence FCU's chain never selected. FCU is the right tool when the evaluation should be sparse, checkable against a target, and explicit about what it omits. TabStruct evaluates structural fidelity against a supplied causal graph and otherwise uses allMLU; FCU instead estimates its evaluation chain from real training data.

8 Limitations and conclusion

Limitations. FCU evaluates only the relationships selected by its sparse, pairwise-derived chain, which can miss jointly informative interactions; the pruning deficit states the dependence omitted by the factorization, not generator errors outside its scored tasks. The categorical-factor identity is exact only under Theorem 1's assumptions; numeric factors use a conditional-mean surrogate and can miss conditional-spread changes. Scores are model- and configuration-relative, and missing-data mechanisms are out of scope. FCU also does not measure privacy or memorization (see Appendix E).

Conclusion. Single-task utility evaluates one conditional relationship and can miss substantial variation elsewhere in a synthetic table. FCU compares real- and synthetic-trained predictors over a sparse chain of conditional tasks that define a distribution, so its summed excess has a stated target (Theorem 1). Controlled experiments show that the excess tracks that target where it is known, and the benchmark shows why this matters: generators matched on TSTR differ in FCU by up to 0.94, per-factor terms name the relationships responsible, and FCU predicts utility on columns it never scored. It agrees with allMLU on rankings while conditioning on one or two parents per factor instead of every column, and it reports the dependence its chain omits rather than treating sparse evaluation as complete joint-fidelity assessment. FCU is therefore a model-relative diagnostic of conditional utility, to be reported alongside marginal, sample-level, and privacy-oriented diagnostics. It provides an auditable view of which selected relationships a generator preserves, what the selected chain omits, and where conditional utility degrades.

Reproducibility statement

An anonymized copy of the code needed to reproduce every table and figure in this paper is linked at the start of the Appendix. The metric is defined in full in Section 3. Every configurable choice is carried as a provenance column on every result row, so results produced under different configurations are never pooled. Theorem 1 and Proposition 1 are proved in Appendix A, with their assumptions stated. Section 4 states the evaluation design and Appendix F.3 gives the full protocol: the held-out split, repeats, sampling seeds, model family, calibration and the bootstrap procedure. Dataset sources, access routes, target derivations and the complete-case preparation rule are in Appendix F.1; the generators' implementations, versions, hyperparameters and seeds are in Appendix F.2. The ground-truth calibration of Section 5.2 runs on synthetic distributions whose answers are closed-form or Monte-Carlo exact, so it is reproducible without any dataset access. Evaluation code, configurations, and per-cell records are released as anonymous supplementary material. Nine of the twelve prepared tables are released with them; for the three that may not be redistributed (Appendix F.1) we release the preparation scripts and column specifications instead.

Ethics statement

This work evaluates synthetic tabular data and does not involve human subjects, user studies, or the collection of new data. Two aspects merit comment. First, MIMIC_III is de-identified clinical data used under credentialed PhysioNet access and its data-use agreement; it is not redistributed here, nor are the two Kaggle competition datasets. No record-level data from any gated source appears in this paper or its artifact. Second, FCU is a *utility* metric and assigns a perfect score to a generator that copies its training rows. We include an oversampling control in the grid to demonstrate exactly this, and we report a distance-to-closest-record column beside every FCU value. Reading FCU as evidence of privacy would be a misuse of it. We make no privacy guarantee for any generator evaluated here.

AI use statement

We used generative AI tools (LLM-based coding assistants) for software implementation and checks, analysis scripting, and editing assistance and throughout this project. All code produced was reviewed by the authors and is covered by the released test suite; every empirical claim in this paper was recomputed from the released code and result records. We did not use generative AI to originate and develop the research question, the metric definition, or the theoretical results, which are the authors' own. An AI assistant helped verify and edit the exposition of the proofs in Appendix A, and the authors checked every step of the final text. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including all text, claims, and artifacts.

References

Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? Sample-level metrics for evaluating and auditing generative models. In *International Conference on Machine Learning*, pp. 290–306. PMLR, 2022.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

C K Chow and C N Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. Wiley, 2nd edition, 2006.

Fida K Dankar and Mahmoud Ibrahim. Fake it till you make it: guidelines for effective synthetic data generation. *Applied Sciences*, 11(5):2158, 2021.

Fida K. Dankar, Mahmoud K. Ibrahim, and Leila Ismail. A multi-dimensional evaluation of synthetic data generators. *IEEE Access*, 10:11147–11158, 2022. doi: 10.1109/ACCESS.2022.3144765.

Khaled El Emam, Lucy Mosquera, Xi Fang, and Alaa El-Hussuna. Utility metrics for evaluating synthetic health data generation methods: Validation study. *JMIR Medical Informatics*, 10(4):e35734, 2022. doi: 10.2196/35734.

Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional GANs. *arXiv preprint arXiv:1706.02633*, 2017.

François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Jean Hausser and Korbinian Strimmer. Entropy inference and the James–Stein estimator, with application to nonlinear gene association networks. *Journal of Machine Learning Research*, 10:1469–1484, 2009.

Xiangjian Jiang, Nikola Simidjievski, and Mateja Jamnik. TabStruct: Measuring structural fidelity of tabular data. In *International Conference on Learning Representations*, 2026.

Alexia Jolicoeur-Martineau, Kilian Fatras, and Tal Kachman. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees. In *International conference on artificial intelligence and statistics*, pp. 1288–1296. PMLR, 2024.

James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N Cohen, and Adrian Weller. Synthetic data – what, why and how? *arXiv preprint arXiv:2205.03257*, 2022.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. *arXiv preprint arXiv:2210.04018*, 2022.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In *International conference on machine learning*, pp. 17564–17579. PMLR, 2023.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E*, 69(6):066138, 2004.

Chaejeong Lee, Jayoung Kim, and Noseong Park. Codi: Co-evolving contrastive diffusion models for mixed-type tabular synthesis. In *International Conference on Machine Learning*, pp. 18940–18956. PMLR, 2023.

Edward Hubert Linfoot. An informational measure of correlation. *Information and Control*, 1(1):85–89, 1957.

Russell Lyons. Distance covariance in metric spaces. *The Annals of Probability*, 41(5):3284–3305, 2013.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B*, 72(4):417–473, 2010.

George A Miller. Note on the bias of information estimates. In *Information Theory in Psychology: Problems and Methods*, pp. 95–100. Free Press, 1955.

Markus Mueller, Kathrin Gruber, and Dennis Fok. Continuous diffusion for mixed-type tabular data. In *International Conference on Learning Representations*, 2025.

Bitya Neuhof and Yuval Benjamini. Confident feature ranking. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 238 of *Proceedings of Machine Learning Research*, pp. 1468–1476. PMLR, 2024.

Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, Oct 2016. doi: 10.1109/DSAA.2016.49.

Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

Zhaozhi Qian, Bogdan-Constantin Cebere, and Mihaela van der Schaar. Synthcity: facilitating innovative use cases of synthetic data in different data modalities. *arXiv preprint arXiv:2301.07573*, 2023.

Brian C Ross. Mutual information between discrete and continuous data sets. *PLOS ONE*, 9(2):e87357, 2014.

Zineb Senane, Axel Karlsson, Lele Cao, Oleg Smirnov, Cheng Zhang, Sahar Asadi, Hedvig Kjellström, Gustav Eje Henter, and Ruibo Tu. Causality for tabular data synthesis: A high-order structure causal benchmark framework. *arXiv preprint arXiv:2406.08311*, 2024.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. Tabdiff: a mixed-type diffusion model for tabular data generation. *arXiv preprint arXiv:2410.20626*, 2024.

Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769–2794, 2007.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

Scott Cheng-Hsin Yang, Baxter Eaves, Michael Schmidt, Ken Swanson, and Patrick Shafto. Structured evaluation of synthetic tabular data. *arXiv preprint arXiv:2403.10424*, 2024.

Zexi Yao, Nataša Krčo, Georgi Ganev, and Yves-Alexandre de Montjoye. The DCR delusion: Measuring the privacy risk of synthetic data. In *Computer Security – ESORICS 2025*, pp. 469–487. Springer, 2025. doi: 10.1007/978-3-032-07884-1_24.

Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. *arXiv preprint arXiv:2310.09656*, 2023.

Appendix

The main text ends with Section 8. Everything below is supplementary, ordered by how directly it supports the claims of the paper.

- Appendix A: proofs, the error decomposition of the estimator, and why FCU clips the sums.
- Appendix B: structure estimation, the class cap, protocol safeguards, and metric definitions.
- Appendix C: the controlled validation, including the ground-truth families and the pairwise blind spot.
- Appendix D: the analyses behind Section 6, the full grid, and the paired leaderboard.
- Appendix E: limitations in full.
- Appendix F: datasets, generator configurations, split validity, and compute.
- Appendix G: extended related work.
- Appendix H: long tables referenced from the sections above.

Reproducibility. An anonymized copy of the code needed to reproduce every table and figure in this paper, including this appendix, is available at `https://anonymous.4open.science/r/factMLU-anon-review-2947/README.md`.

A Proofs and theoretical details

Notation. We index the columns by their position in the ordering π. We write $X_{<j} := (X_1, \ldots, X_{j-1})$ for the predecessors of X_j. The predecessors split into the selected parents and the omitted ones, $X_{<j} = (\mathrm{Pa}(X_j), R_j)$. All logarithms are natural, so every quantity is in nats. For a predictive distribution q of factor j, its risk on real data is $\mathcal{R}_j(q) := \mathbb{E}_P\big[-\log q(X_j \mid X_{\mathrm{Pa}(j)})\big]$.

A.1 Proof of Proposition 1

The hypothesis of the proposition requires $Q_1(Y \mid Z = z)$ and $Q_2(Y \mid Z = z)$ to be defined for every z in $\operatorname{supp} P(Z)$. This means that both synthetic distributions cover the real support of Z.

Step 1: the trained predictor depends on Q only through $Q(Y \mid Z)$. By assumption (iii), training returns the population risk minimizer under Q,

$$f_Q \;=\; \arg\min_{f \in \mathcal{F}} \; \mathbb{E}_{(Z,Y)\sim Q}\big[\ell(f(Z), Y)\big].$$

By assumption (i), this minimizer is the Bayes predictor of Y given Z under Q. The Bayes predictor is computed separately at each point z, and at z it uses only $Q(Y \mid Z = z)$. It does not use the marginal $Q(Z)$.

Step 2: the two predictors agree wherever they are evaluated. If $Q_1(Y \mid Z = z) = Q_2(Y \mid Z = z)$ for all $z \in \operatorname{supp} P(Z)$, then $f_{Q_1}(z) = f_{Q_2}(z)$ for all such z.

Step 3: the score only looks at that region. The score in Eq. (1) is an expectation under P. It only evaluates the predictor at points $z \in \operatorname{supp} P(Z)$. Hence $U_{\mathrm{TSTR}}(Q_1; P) = U_{\mathrm{TSTR}}(Q_2; P)$. The same argument covers assumption (ii) in its general form. Any evaluation that depends only on the joint law of $\big(f(Z), Y\big)$ under P, such as AUC, gives the same value for both predictors. □

Neither $Q(Z)$ nor the dependence among the components of Z appears anywhere in the argument. That is the content of the proposition.

Remark 1 (The statement is for TSTR, not TRTS)**.** TRTS *trains on real data and evaluates under Q: its value is $\mathbb{E}_{(Z,Y)\sim Q}\big[\ell(f_P(Z), Y)\big]$ for the real-trained predictor f_P. The expectation is over Q, so $Q(Z)$ enters as the weighting of the evaluation and the invariance above does not hold.* TRTS *is still a single-task score: any change in Q registers only through the loss of one fixed predictor at the one target, so it is not a substitute for scoring the other conditionals. Both scores are reported as comparators (Appendix B.4).*

A.2 Single-task utility with finite samples

Proposition 1 is a statement about the population limit. With finitely many synthetic rows, $Q(Z)$ does affect the fitted model, because it decides where the training examples fall. If $Q(Z)$ places few rows in some region of $\operatorname{supp} P(Z)$, the model is learned poorly there, even when $Q(Y \mid Z)$ is exact.

Remark 2 (Finite samples)**.** *With finite synthetic data, the invariance in Proposition 1 holds only approximately. The corruption experiments of Section 5.1 measure how closely it holds in practice. Severe distortion of the non-target columns leaves* TSTR *almost unchanged, so the population-limit reading describes practice well.*

A.3 Proof of Theorem 1

We assume $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) < \infty$. This guarantees that every conditional $Q(x_j \mid x_{\mathrm{Pa}(j)})$ used below is defined P-almost everywhere. Note also that Q^{Pa} is a valid distribution. Each factor conditions only on earlier columns, so the product in Eq. (5) sums (or integrates) to one.

Step 1: expand both distributions along the chain. The chain rule gives $P(x) = \prod_j P(x_j \mid x_{<j})$. By definition, $Q^{\mathrm{Pa}}(x) = \prod_j Q(x_j \mid x_{\mathrm{Pa}(j)})$. The log ratio of two products is a sum of log ratios, so

$$D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) \;=\; \sum_{j=1}^{d} \mathbb{E}_P \left[\log \frac{P(X_j \mid X_{<j})}{Q(X_j \mid X_{\mathrm{Pa}(j)})} \right].$$

Step 2: add and subtract $\log P(X_j \mid X_{\mathrm{Pa}(j)})$ *inside each term.*

$$D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) \;=\; \underbrace{\sum_{j=1}^{d} \mathbb{E}_P \left[\log \frac{P(X_j \mid X_{<j})}{P(X_j \mid X_{\mathrm{Pa}(j)})} \right]}_{\text{(a)}} \;+\; \underbrace{\sum_{j=1}^{d} \mathbb{E}_P \left[\log \frac{P(X_j \mid X_{\mathrm{Pa}(j)})}{Q(X_j \mid X_{\mathrm{Pa}(j)})} \right]}_{\text{(b)}}.$$

Every term in (a) and in (b) is an expected log ratio between two conditional distributions of the same variable. Each is therefore nonnegative. Their total is finite by assumption, so every term is finite and the split is valid.

Step 3: identify the two sums. Since $X_{<j} = (\mathrm{Pa}(X_j), R_j)$, the j-th term of (a) is the conditional mutual information $I_P\big(X_j; R_j \mid \mathrm{Pa}(X_j)\big)$, by its definition. So (a) equals $\Delta(P)$. For (b), take the expectation over X_j given $X_{\mathrm{Pa}(j)}$ first. The j-th term becomes

$$\mathbb{E}_{X_{\mathrm{Pa}(j)} \sim P} \Big[D_{\mathrm{KL}}\big(P(X_j \mid X_{\mathrm{Pa}(j)}) \,\big\|\, Q(X_j \mid X_{\mathrm{Pa}(j)})\big) \Big],$$

so (b) equals $D(P, Q)$. This proves Eq. (8). We also have $D(P, Q) \geq 0$. Equality holds if and only if, for every j, $Q(\cdot \mid x_{\mathrm{Pa}(j)}) = P(\cdot \mid x_{\mathrm{Pa}(j)})$ for P-almost every $x_{\mathrm{Pa}(j)}$.

Step 4: the remaining claims.

- *When the deficit is zero.* $\Delta(P)$ is a sum of nonnegative terms. It is zero if and only if every term $I_P(X_j; R_j \mid \mathrm{Pa}(X_j))$ is zero. That holds if and only if $P(x_j \mid x_{<j}) = P(x_j \mid x_{\mathrm{Pa}(j)})$ for every j, which says that P factorizes over the chain.
- *The deficit is the best achievable divergence.* Let $\mathcal{Q}_{\mathrm{Pa}} = \big\{Q' : Q'(x) = \prod_j q_j(x_j \mid x_{\mathrm{Pa}(j)})\big\}$. For any member Q', the conditional of X_j given its parents is q_j itself, so $Q'^{\mathrm{Pa}} = Q'$. Eq. (8) then gives $D_{\mathrm{KL}}(P \,\|\, Q') = \Delta(P) + D(P, Q') \geq \Delta(P)$. The choice $q_j = P(\cdot \mid x_{\mathrm{Pa}(j)})$ makes $D(P, Q') = 0$ and attains the bound. Hence $\min_{Q' \in \mathcal{Q}_{\mathrm{Pa}}} D_{\mathrm{KL}}(P \,\|\, Q') = \Delta(P)$. This is the classical cost of projecting P onto a pruned factorization (Cover & Thomas, 2006; Koller & Friedman, 2009).
- *When Q factorizes over the chain.* Then $Q^{\mathrm{Pa}} = Q$, and Eq. (8) reads $D_{\mathrm{KL}}(P \,\|\, Q) = \Delta(P) + D(P, Q)$. □

Root columns. A column with an empty parent set has no conditional task. Its term in (b) is the marginal divergence $D_{\mathrm{KL}}(P(X_j) \,\|\, Q(X_j))$, and the sum of these terms is $B(P, Q)$ of Section 3.1. The estimator of Section 3.2.2 scores only the conditional factors $\mathcal{C}$, so its target is $D_{\mathcal{C}}(P, Q) = D(P, Q) - B(P, Q) = D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P) - B(P, Q)$. A generator that reproduces every scored conditional and distorts only a root marginal therefore receives $\mathrm{FCU} = 1$; the marginal fidelity MF reports the distortion as a similarity, not as the missing KL term. The chain of Section 3.2 has a single root, so B is one term.

Excess vs. the full divergence. For a general Q, the full divergence $D_{\mathrm{KL}}(P \,\|\, Q)$ and the target $D(P, Q)$ differ. Their gap, $D_{\mathrm{KL}}(P \,\|\, Q) - D(P, Q)$, is the part of the divergence that only shows up when conditioning on omitted predecessors. Two cases are exact. The gap is 0 when $Q = P$. The gap equals $\Delta(P)$ when Q keeps every marginal of P and removes all dependence, because then $D_{\mathrm{KL}}(P \,\|\, Q)$ is the total dependence $I(P) + \Delta(P)$ and $D(P, Q) = I(P)$. Between these two cases the gap has no fixed sign. On the five-column Gaussian ladder of Appendix C.2, where $\Delta(P) = 0$, it is slightly negative in the middle of the ladder and zero at both ends. This is why Table 4 reports the bias of $\widehat{D}$ against both targets.

A.4 Proof of Corollary 1

Fix a parent value x and write $p := P(X_j \mid X_{\mathrm{Pa}(j)} = x)$. For any predictive distribution q,

$$\mathbb{E}_{X_j \sim p}\big[-\log q(X_j) \big] \;=\; H(p) + D_{\mathrm{KL}}(p \,\|\, q).$$

The expected log score is therefore smallest at $q = p$, and the excess of any other q over that minimum is $D_{\mathrm{KL}}(p \,\|\, q)$. This is the statement that the log score is strictly proper (Gneiting & Raftery, 2007).

First claim. If the real arm attains the Bayes predictor of P, its prediction at x is p, and its excess is 0. If the synthetic arm attains the Bayes predictor of Q, its prediction at x is $Q(X_j \mid X_{\mathrm{Pa}(j)} = x)$, and its excess is $D_{\mathrm{KL}}\big(p \,\|\, Q(X_j \mid X_{\mathrm{Pa}(j)} = x)\big)$. The difference of the two risks at x is that divergence. Averaging over $X_{\mathrm{Pa}(j)} \sim P$ gives Eq. (11).

Second claim. The climatological reference predicts $P(X_j)$ and ignores the parents. Its excess over the real arm is

$$\mathbb{E}_{X_{\mathrm{Pa}(j)} \sim P}\Big[D_{\mathrm{KL}}\big(P(X_j \mid X_{\mathrm{Pa}(j)}) \,\big\|\, P(X_j)\big)\Big] \;=\; I_P\big(X_j; \mathrm{Pa}(X_j)\big),$$

which is one of the standard forms of mutual information. □

A.5 What the finite-sample estimator adds

The corollary assumes that both arms attain their Bayes predictor. With finite samples neither does. Let $\mathcal{R}_j^\star$ be the smallest achievable risk on real data, attained by $P(X_j \mid X_{\mathrm{Pa}(j)})$. Define three errors:

$$\begin{aligned}
\epsilon_j^{\mathrm{real}} &:= \mathcal{R}_j\big(q_j^{\mathrm{real}}\big) - \mathcal{R}_j^\star \;\geq\; 0,\\
\epsilon_j^{\mathrm{syn}} &:= \mathcal{R}_j\big(q_j^{\mathrm{syn}}\big) - \mathcal{R}_j\big(Q(X_j \mid X_{\mathrm{Pa}(j)})\big),\\
\epsilon_j^{\mathrm{ref}} &:= \mathcal{R}_j\big(\text{fitted reference}\big) - \mathcal{R}_j\big(P(X_j)\big) \;\geq\; 0.
\end{aligned}$$

The first is the real arm's distance from the best possible predictor. The second is the synthetic arm's distance from the Bayes predictor of Q. It can have either sign, because the Bayes predictor of Q is not the best predictor on real data. The third is the error of estimating the climatology from training frequencies, and it is small. Write D_j for the j-th term of $D(P, Q)$, and recall that $\mathcal{R}_j\big(Q(X_j \mid X_{\mathrm{Pa}(j)})\big) = \mathcal{R}_j^\star + D_j$ by Corollary 1. Then the two per-factor quantities that enter Eq. (15) decompose as

$$\ell_j^{\mathrm{syn}} - \ell_j^{\mathrm{real}} \;\approx\; D_j + \epsilon_j^{\mathrm{syn}} - \epsilon_j^{\mathrm{real}}, \qquad \ell_j^{\mathrm{ref}} - \ell_j^{\mathrm{real}} \;\approx\; I_P\big(X_j; \mathrm{Pa}(X_j)\big) + \epsilon_j^{\mathrm{ref}} - \epsilon_j^{\mathrm{real}}, \tag{16}$$

where "≈" stands for the sampling noise of a finite test split. Three points follow.

- *The paired design cancels shared error.* In the numerator the two errors enter with opposite signs. Both arms use the same learner, budget, and task. To the extent that the learner errs alike under P and under Q, the errors cancel. They do not cancel fully, which is why Section 5.2 measures what remains.
- *The denominator carries the real arm's error alone.* An overconfident real arm makes $\epsilon_j^{\mathrm{real}}$ large and can push $\ell_j^{\mathrm{ref}} - \ell_j^{\mathrm{real}}$ below zero. The calibration of Section 3.2.2 limits this, and we report the number of factors with a negative value as a diagnostic.
- *Numeric factors have their own target.* All three arms of a numeric factor use the Gaussian log score $\frac{1}{2}\log(2\pi e \cdot \mathrm{MSE})$, with MSE measured on held-out real rows, so no predictive variance is fitted. This gives $\ell_j^{\mathrm{syn}} - \ell_j^{\mathrm{real}} = \frac{1}{2}\log\big(\mathrm{MSE}_j^{\mathrm{syn}}/\mathrm{MSE}_j^{\mathrm{real}}\big)$ and $\ell_j^{\mathrm{ref}} - \ell_j^{\mathrm{real}} = -\frac{1}{2}\log\big(1 - R_j^2\big)$, neither a conditional KL in general. The first is a divergence of conditional means only: a change in conditional spread alone leaves the excess at 0 while the conditional KL is positive (0.318 nats for $Y \mid X \sim \mathcal{N}(x, 1)$ against $\mathcal{N}(x, 4)$); a mean shift $\mu(x)$ at fixed variance gives excess $\frac{1}{2}\log(1 + 2D_j)$, growing logarithmically where the KL grows linearly (shown on the mean-shift ladder, Appendix C.3). Corollary 1 is therefore exact for categorical factors and an approximation for numeric ones; a quantile-binned target scored as categorical makes the identity exact, at the cost of the information the bins discard (Appendix C.3 measures this trade-off, and keeps the Gaussian score as the default). The score is computed on the target's normal scores from the empirical CDF of the real training split, so the map is monotone but not invertible, and synthetic values outside the training range all map to one extreme.

A.6 Clipping the sum, not each factor

Eq. (15) is a ratio of two sums with one clip. The natural alternative, and the form of the all-column utility metrics of Section 7, is a mean of clipped per-factor ratios, which we run as an ablation arm (Appendix D.6):

$$\mathrm{FCU}_{\mathrm{ratio}} \;=\; \frac{1}{|\mathcal{J}|}\sum_{j \in \mathcal{J}} \phi_j, \qquad \phi_j \;=\; \min\Big(\frac{U_j^{\mathrm{syn}}}{U_j^{\mathrm{real}} + \epsilon},\, 1\Big).$$

Here U_j^{real} and U_j^{syn} are per-factor utilities clipped at zero, and $\mathcal{J}$ keeps only the factors clearing a gate $U_j^{\mathrm{real}} \geq \tau$. This form fails in two ways. Clipping each factor before averaging adds a bias that grows with the number of factors: on the oracle ($Q = P$, so $\widehat{D}_j$ has mean near zero), a clip at zero replaces $\widehat{D}_j$ by $\max(\widehat{D}_j, 0)$, whose mean is about $\sigma/\sqrt{2\pi}$ per factor for a roughly Gaussian $\widehat{D}_j$ with SD σ; clipping m factors separately

adds about $m\,\sigma/\sqrt{2\pi}$, against $\sqrt{m}\,\sigma/\sqrt{2\pi}$ for one clip on the sum, which is why Eq. (15) clips only once (Section 3.2.2) and stays close to 1 on the oracle (Table 3). Second, ϕ_j is unstable when the real signal is weak. We test this directly (`scripts/phi_degeneracy_demo.py`) on one factor $y = \beta x + \varepsilon$ with an *ideal* synthetic dataset drawn i.i.d. from the same joint (100 seeds, $n = 1000$, gradient-boosted arms, no gate). A well-behaved per-factor score would give $\phi \approx 1$ at every β; Table 1 shows it does not.

Table 1: Distribution of the per-factor ratio ϕ for an ideal generator, over 100 seeds. At weak signal the ratio is 0 or 1 almost at random.

β	true R^2	$\phi = 0$	$\phi = 1$	$0 < \phi < 1$
0.00	0.000	100%	0%	0%
0.05	0.003	99%	1%	0%
0.10	0.010	98%	2%	0%
0.20	0.040	54%	32%	14%
0.50	0.250	0%	55%	45%

At weak signal, ϕ is 0 or 1 depending on which arm's utility happens to clear zero, and only settles at strong signal. The gate τ hides the symptom but needs an arbitrary threshold and still inflates weak generators' scores by about 0.1 just above it (Appendix D.6). The ratio of sums in Eq. (15) avoids both problems without a threshold: a factor with no real signal contributes about 0 to both $\widehat{I}$ and $\widehat{D}$, unless Q invents a dependency there, in which case $D_j > 0$ correctly lowers FCU.

B METHOD DETAILS

This appendix gives the full construction behind Section 3.

B.1 STRUCTURE ESTIMATION

All structure estimation uses the real training split only. Held-out test rows never influence the tasks that later score them.

Association matrix. We estimate a symmetric matrix $A \in [0,1]^{d\times d}$ of pairwise dependence. Using a different measure for each type pair, such as $|\rho|$ for numeric pairs and η^2 for mixed pairs, puts blocks of the matrix on different scales. Every later step that compares entries across blocks is then biased. We therefore use one measure, mutual information, with an estimator suited to each type pair. Numeric pairs use KSG (Kraskov et al., 2004). Mixed pairs use the Ross estimator (Ross, 2014). Categorical pairs use the plug-in estimator with the Miller–Madow correction (Miller, 1955; Hausser & Strimmer, 2009). The Linfoot coefficient (Linfoot, 1957) maps mutual information to the scale of a correlation:

$$A_{ij} \;=\; \sqrt{1 - e^{-2I(X_i;\,X_j)}}. \tag{17}$$

For a bivariate Gaussian with correlation ρ, $I = -\frac{1}{2}\log(1-\rho^2)$, so $A_{ij} = |\rho|$ exactly. All type pairs thus share one scale, and the measure still captures nonlinear dependence. Constant columns receive zero association. We compute A on a seeded row subsample, since the sampling error of an association estimate shrinks roughly as $1/\sqrt{n}$. The effect on the resulting ranking is small (Appendix D.6). As an alternative measure we implement distance correlation (Székely et al., 2007), with the discrete metric for categorical variables (Lyons, 2013), and report it as an ablation arm.

Ordering. Any ordering gives a valid chain. The ordering matters because it decides which columns may serve as parents. We order by breadth-first traversal of the maximum spanning tree of A, starting at its most central node. With mutual-information weights, that undirected tree is the Chow–Liu tree, the best tree-structured approximation of the joint in KL divergence (Chow & Liu, 1968). This guarantee does not transfer to the chain we score, for two reasons. The tree is undirected, so the choice of root fixes the direction of every edge arbitrarily. Parent selection may also add parents that are not tree neighbors. The ordering therefore remains a modeling choice. To bound its influence we implement an *ordering ensemble*: FCU averaged over M seeded random permutations, reported with the spread across orderings. We use it as an ablation arm.

Parent sets. For the j-th column we choose $\mathrm{Pa}(X_j) \subseteq \{X_1, \ldots, X_{j-1}\}$ with $|\mathrm{Pa}(X_j)| \le k$. Ranking predecessors by association alone may be wastful, for example when there is high correlation between predecessors. We use max-relevance min-redundancy instead (Peng et al., 2005). Starting from the empty set S, we repeatedly add the candidate c that maximizes

$$A_{jc} \;-\; \lambda \cdot \mathrm{mean}_{s\in S}\, A_{cs}.$$

The first term rewards relevance to the target. The second penalizes overlap with parents already selected.

Adaptive pruning of the parent sets k. The parent set grows only while it helps. A small probe model is trained on an internal split of the real training data, without tuning. A candidate parent is kept only if it improves the probe's held-out utility by more than δ. This sets $|\mathrm{Pa}(X_j)|$ separately for each factor, and we report the mean parent-set size as a diagnostic.

Settings and tie rules. *Associations:* the KSG estimator for numeric pairs and the Ross estimator for mixed pairs both use 3 nearest neighbors; the plug-in estimator for categorical pairs carries the Miller–Madow correction; a corrected estimate below 0 is set to 0 before the Linfoot map; the matrix is computed on a seeded subsample of at most 50,000 training rows (2,000 for the distance-correlation ablation); a constant column has association 0 with every other. *Ordering:* the maximum spanning tree is computed on the weights $1+10^{-9}-A_{ij}$, so ties and zero associations still receive an edge and the tree stays connected; the root is the column with the largest mean off-diagonal association; the breadth-first traversal visits the neighbors of a node in column order. *Parents:* mRMR with $\lambda = 1$ over a candidate budget of at most $k = 10$ parents; at each step the candidate with the largest score is taken, and a tie goes to the candidate that comes first in the ordering. *Adaptive k:* the probe is the evaluation learner with at most 50 trees and no tuning, fitted on a seeded $80/20$ split of the real training rows; the ranked parents are added one at a time and kept up to the last one whose held-out gain over the previous set exceeds $\delta = 0.005$ in the probe's skill score. The per-factor cap on probe rows (`factor_max_rows`) is off.

Further selectors, used as ablations. We implement three alternatives. *Stability selection* (Meinshausen & Bühlmann, 2010) requires a candidate to be selected in at least a fraction π_0 of bootstrap resamples of the training split. It multiplies the cost of structure estimation, so we use it as a robustness check and not as a default. A *conditional mutual information* selector follows Fleuret (2004). The *tree-gain screen* ranks candidates by the split gain of one small boosted model fitted on all predecessors, under the same parent budget. It costs one extra fit per column, and that fit is shared by every generator. Unlike any pairwise statistic, it can see interactions (Appendix C.4).

B.2 THE CLASS CAP

A cap on the number of categories was applied to categorical variables with more than C_{max} categories. A categorical column with thousands of levels defines a task that is neither tractable nor informative. Gradient-boosted multiclass models fit one tree per class per boosting round, so cost grows linearly with the number of levels. A level this rare that occurs a few hundred times in half a million rows also carries almost no probability mass for a scoring rule to credit.

We learn one target-level map from the real training split: the $C_{\max}$ most frequent levels are kept and all other levels merge into one *other* class, so a capped factor is scored over $C_{\max} + 1$ classes. The map is applied unchanged to the test split and to every synthetic arm; a category that occurs only in synthetic data falls into *other* and does not widen the label space. The map affects the scored target only. The same column keeps its full vocabulary wherever it serves as a parent of another factor and in the marginal fidelity MF. The consequence for the estimand is that a capped factor scores the coarsened conditional $P(\kappa(X_j) \mid \mathrm{Pa}(X_j))$, where κ is the level map, while the column enters other factors uncoarsened; the scored factors therefore do not form a product of conditionals over one common coarsened space. The cap is a reported quantity, because it defines the task that was scored, and FCU then measures how well a generator reproduces the conditional structure of the levels that carry mass, not of the extreme tail.

At $C_{\max} = 500$ the cap binds on three tables: CRITEO_1M (4,600 levels), DIABETES_130US (632), and MIMIC_III (571). Its effect is largest on the first. There it merges the tail of three columns into a residual class that holds 10 to 30% of rows, and the least frequent scored class has about 160 training examples. A separate per-factor cap on training rows exists in the library but was not used for any reported result. In every reported run, both arms train on the same number of rows: the smaller of the synthetic file and the real training subsample of Section 4, which in this corpus is the synthetic file's size.

To analyze the effect of using the cap, we reran the full protocol of Section 4 on DIABETES_130US at $C_{\max} \in \{50, 100, 250, 500, 1000\}$. This is the wider of the two binding tables that a local machine can sweep. Each rung has twelve entries, three sampling seeds, and three repeats, for 108 cells. Cells are matched to the campaign's own cells (the $C_{\max} = 500$ rung) by model, seed, and repeat. No run failed.

Table 2 gives every rung. Mean FCU stays within 0.007 of the default over a twenty-fold range of the cap (Table 2). The Spearman correlation with the default rung never drops below 0.969. The change is not monotone in $C_{\max}$. The $C_{\max} = 1000$ rung equals the default in every cell. These runs train both arms on as many rows as the synthetic file holds. So neither cap binds at $C_{\max} = 1000$. The lower rungs move in both directions. This pattern fits a cap that changes one column's task, not a steady loss of accuracy as the cap shrinks. This analysis was not applied to MIMIC_III.

Table 2: Class-cap sensitivity on DIABETES_130US under the protocol of Section 4 (twelve entries, three sampling seeds, three repeats, 108 cells per rung). Each rung is an independent evaluation, matched to the campaign's own cells, the $C_{\max} = 500$ rung, by (model, seed, repeat). Spearman and mean $|\Delta\text{FCU}|$ are against that rung.

$C_{\max}$	mean FCU	Spearman vs. default	mean $\lvert\Delta\text{FCU}\rvert$
50	0.1063	0.9685	0.0105
100	0.1001	0.9775	0.0033
250	0.0934	0.9821	0.0058
1000	0.0990	1.0000	0.0000
500 (default)	0.0990	–	–

B.3 PROTOCOL SAFEGUARDS

Six rules keep the two arms comparable.

- *No leakage.* The association matrix, the ordering, and all parent sets come from the training split only.
- *Equal sample sizes.* Both training sets are subsampled to the smaller of the two sizes, so a difference between the arms reflects the distribution and not the number of rows.
- *Symmetric models.* Both arms share one model class, seed, and budget. If hyperparameters are tuned, tuning runs once on the real arm and the synthetic arm reuses the result.
- *Shared label spaces.* Where the class cap binds, both arms use the level map of the real training split.
- *Three layers of uncertainty.* Split repeats; sampling seeds crossed with the repeats; and a paired bootstrap over test rows, which recomputes every score from cached predictions (no refitting) and resamples rows jointly across factors, since they share the test split and are correlated.
- *Calibration.* Each arm's predictions are shrunk toward the marginal of its own training rows: a seeded 15% slice is held out, and a weight $w \in [0, 1]$ on the arm's training class frequencies (training mean for a numeric target) is chosen to minimize the loss on that slice, giving prediction $(1 - w)$ times the model's output plus w times that marginal. The weight is near zero on most factors and limits the overconfidence of boosted models on high-cardinality targets; the synthetic arm is shrunk toward its own table's marginal, so no real frequency enters it. An arm whose slice or remainder would hold fewer than 50 rows is fit on all its rows without shrinkage.

B.4 DEFINITIONS OF THE REPORTED METRICS

- **FCU and** MF**.** Eq. (15) and the marginal fidelity of Section 3.2.2, with paired-bootstrap confidence intervals. The two sums $\widehat{D}$ and $\widehat{I}$ are reported in nats beside FCU.
- **Single-task utility.** TSTR, TRTS, and TRTR on the dataset's standard target, as chance-corrected skill scores. We do not report a ratio of macro-F1 values. Macro-F1 is not zero at chance level, so such a ratio is not comparable across datasets. We also report the threshold-free AUC. On an imbalanced target, a 0.5 decision threshold can leave both arms predicting one class for every row.
- **Discrimination AUC.** The AUC of a gradient-boosted classifier trained to tell real rows from synthetic rows, with real rows as the positive class. It is trained on a 70% slice of the real rows and a matching slice of the synthetic rows and evaluated on the remainder of each. A value of 0.5 means indistinguishable for this classifier; it does not establish equality of the two distributions. A generator that copies training rows places the same records in both classes of the classifier's training set, while the evaluation real rows are ones it never saw, which drives the raw AUC *below* 0.5. We therefore also report the symmetric score $1 - 2\,|\text{balanced accuracy} - 0.5|$, a different quantity from the AUC, which is low both for copied data and for easily detected data.
- **DCR.** The distance to the closest record. It is a diagnostic of nearest-neighbor disclosure and not a privacy guarantee. Let $d(x, \mathcal{R})$ be the Gower distance from row x to its nearest neighbor in a reference set $\mathcal{R}$ of real rows. Let q_5 be the fifth percentile. The score is

$$\text{DCR} = \text{clip}\left(\frac{q_5\{d(x, \mathcal{R}) : x \in \mathcal{D}_{\text{syn}}\}}{q_5\{d(x, \mathcal{R}) : x \in \mathcal{D}_{\text{cmp}}\}},\ 0,\ 1\right),$$

  where $\mathcal{D}_{\text{cmp}}$ is a separate slice of real rows that supplies the real-to-real yardstick. We use the lower tail because a few near-copies barely move a median. A score of 1 means that the closest synthetic rows are no closer to real records than other real records are. A score of 0 means that at least one synthetic row in twenty coincides with a real record. On a table with only discrete columns, such as CDC_DIABETES, exact coincidences happen for many generators. A 0 there should be read together with the two median distances, which we report with every score. Values below 1 are evidence about disclosure risk. They are not a proof of copying and not a bound on it.

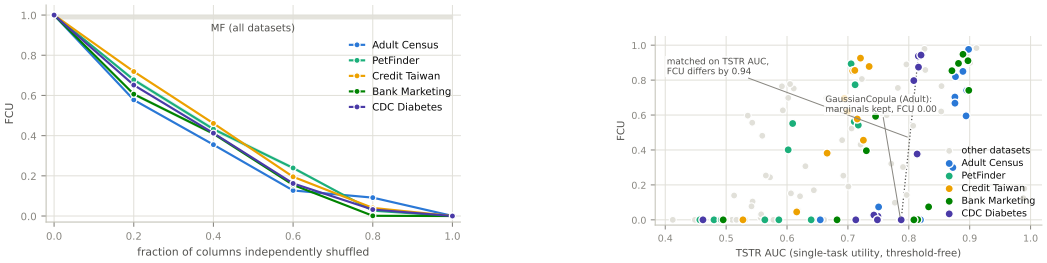


Figure 5: **Left:** the corruption ladder. Shuffling a growing fraction of columns independently destroys their dependencies and preserves every marginal exactly. FCU (mean over five seeds, shaded 95% paired-bootstrap CIs) falls toward zero, while MF (gray band) stays flat; five tables drawn, all twelve in Table 3. **Right:** FCU against threshold-free TSTR AUC for every cell of the benchmark (means over runs; five datasets colored). The vertical spread is what the single-task protocol discards. Eleven of twelve tables: TSTR AUC is undefined on BEIJING_PM25 (Appendix D.5).

- **Per-factor terms.** The three log scores of every factor. These locate a failure in a named conditional.
- **Sample-level fidelity.** α-precision, β-recall, and authenticity (Alaa et al., 2022), computed on the same Gower embedding as DCR.

A further diagnostic concerns copying, computed on a subset of the tables and never combined with FCU (Appendix D.8). **Near-copy share**: the share of synthetic rows that lie closer to a training row than real rows usually do. Distances are the Gower distances of DCR. The cutoff τ is the 5% quantile of the distances from held-out real rows to their nearest training row. A synthetic row is a near-copy if its distance to the nearest training row is below τ, or if it equals a training row exactly. A fresh sample of real rows therefore scores about 5% by construction, and the setting is a false-positive rate and not a distance. Distances are computed against at most 2,000 sampled training rows, while exact matches are searched in the full training split. Every run also reports the rate at which the rule flags the held-out real rows themselves, which checks the calibration.

C CONTROLLED VALIDATION: DETAILS

C.1 CHECKS ON REAL DATA, PER DATASET

Section 5.1 summarizes four checks whose answers are known in advance. Table 3 gives them per dataset and Figure 5 draws the corruption ladder. In the oracle check, the training slice and the pseudo-synthetic slice have equal size, so both arms train on the same number of rows. The oracle's standard deviation across seeds is below 0.005 everywhere except DIABETES_130US (0.063) and CREDIT_DEFAULT_TAIWAN (0.066).

The target-preserving construction. Figure 2b keeps the benchmark target of ADULT_CENSUS joined to its m strongest associates in the Linfoot matrix (`relationship`, `marital-status`, `capital-gain`, `education-num`, `education`, `age`, and `occupation`, in that order) and permutes every other column independently. Every marginal is preserved, and only the dependence outside that block is destroyed. We run five seeds at $m \in \{1, 2, 3, 5, 7\}$ with the campaign evaluator (`scripts/teaser_data.py`). The TSTR AUC, as a ratio to its train-on-real ceiling, reads 0.83, 0.83, 0.89, 0.96, and 0.98. FCU reads 0.03, 0.19, 0.20, 0.50, and 0.79, and allMLU reads 0.02, 0.19, 0.21, 0.56, and 0.84; MF is 0.99 throughout. At $m = 7$ the single task is within two points of its ceiling while the two conditional sums are still a fifth or more below theirs.

allMLU on the same splits. allMLU (Section 4) was scored beside FCU on the oracle and every ladder rung of the ten tables, on the same split and synthetic sample (`factmlu-validate -global-utility`; the `validation_global_utility_v5` result directories). The two agree on real tables: the oracle reads 0.954–1.000 for FCU and 0.948–0.999 for allMLU, both fall from 1.0 to 0.0 along the ladder, and the 300 ladder cells rank at Spearman 0.977 (mean gap 0.05 on interior rungs, largest 0.30 on BEIJING_PM25 and DIABETES_130US). The ladder cannot separate them, since both are sums of proper scores over conditionals of the same table; what differs is the number. At full corruption the allMLU excess is 2.5–19.5 nats against 1.2–11.0 nats for the chain on the same tables, because the all-others sum double-counts dependence and has no population quantity to be read against (derivation in Appendix C.3). The exact ladder of Appendix C.5 measures that gap directly.

C.2 THE GROUND-TRUTH FAMILIES

Section 5.2 compares FCU with divergences that are known exactly. This subsection specifies each family.

Table 3: Validation on real data (defaults of Section 3; mean over seeds 42 to 46). *Oracle*: a held-out real slice of equal size, scored as if synthetic, ± SD across seeds. *Ladder*: a fully corrupted copy (f=1), with all dependencies destroyed and all marginals preserved. *Memorization*: copied training rows. Twelve of twelve tables for Oracle and Ladder; eleven of twelve for Memorization – CRITEO_1M's memorization probe did not complete within an achievable compute budget (its narrower conditioning tasks are unaffected) and is reported as unavailable, marked with a dash, not as a failure of the metric.

	Oracle	Ladder (f=1)		Memorization	
Dataset	FCU ± SD	FCU	MF	FCU	Disc. AUC
CDC_DIABETES	1.000 ± 0.000	0.000	0.998	1.000	0.44
CRITEO_1M	1.000 ± 0.000	0.000	0.990	–	–
BEIJING_PM25	1.000 ± 0.000	0.000	0.991	1.000	0.35
ADULT_CENSUS	1.000 ± 0.000	0.000	0.992	1.000	0.36
MIMIC_III	0.999 ± 0.001	0.000	0.996	1.000	0.44
PETFINDER_TAB	0.999 ± 0.002	0.000	0.985	1.000	0.31
SUPPORT2	0.998 ± 0.003	0.000	0.982	1.000	0.20
HOME_CREDIT	0.998 ± 0.003	0.000	0.996	1.000	0.42
MAGIC	0.997 ± 0.002	0.000	0.986	1.000	0.27
BANK_MARKETING	0.996 ± 0.004	0.000	0.993	1.000	0.35
DIABETES_130US	0.972 ± 0.063	0.000	0.994	1.000	0.34
CREDIT_DEFAULT_TAIWAN	0.954 ± 0.066	0.000	0.989	1.000	0.31

What is exact and what is estimated. In every cell, P and Q_t are population distributions with declared parameters. The divergence $D_{\mathrm{KL}}(P \,\|\, Q_t)$ is computed from those parameters. It is never estimated from samples. FCU is computed exactly as on real data. We draw $n_{\mathrm{train}} = 4000$ rows from P as the real training split, $n_{\mathrm{test}} = 2000$ rows from P as the held-out split, and n_{train} rows from Q_t as the synthetic training split. We then run the whole pipeline on those samples: association matrix, ordering, parent selection, and both arms. **The structure is estimated, not supplied.** Each FCU value therefore includes the error of structure estimation as well as model error. The x-axis of every calibration is a population quantity and the y-axis is a finite-sample statistic. Every rung is run at five seeds (42 to 46), and each seed redraws all three splits.

Gaussian copula, $d = 5$. A row is drawn as $Z \sim \mathcal{N}(0, R)$ with $X_j = F_j^{-1}(\Phi(Z_j))$. Every F_j is standard Gaussian on this track, so $X = Z$ and the model is a Gaussian with unit variances. The general model we use also offers several classes, a t-copula, and marginals with point masses. All of these are switched off here, because the closed forms below require a single Gaussian.

The correlation matrix R is built from a sparse *precision* matrix Θ. The off-diagonal non-zeros of Θ follow the path $X_1 - X_2 - X_3 - X_4 - X_5$. Edge weights are drawn from $[0.45, 0.9]$ with a random sign. Each diagonal entry equals 0.25 plus the absolute sum of its row, which makes Θ positive definite. We invert Θ and rescale to a unit diagonal. Rescaling by a positive diagonal matrix keeps the zeros of the precision matrix in place. The conditional independencies of the model are therefore exactly those of the path, which gives the structure-recovery check a known target. With these constants the condition number of R is about 10 and the strongest correlation is about 0.69. At weaker settings an omitted parent costs about 0.001 nats, which is too small to detect.

The perturbation weakens the dependence and leaves the marginals unchanged:

$$Q_t: \quad R_t = (1-t)\,R + t\,I, \qquad t \in \{0, 0.1, \ldots, 1.0\}.$$

R_t is a valid correlation matrix with a unit diagonal at every t. Each X_j is therefore standard Gaussian under every Q_t, and the whole divergence comes from the dependence. The divergence is the standard Gaussian expression

$$D_{\mathrm{KL}}\big(\mathcal{N}(0,R)\,\big\|\,\mathcal{N}(0,R_t)\big) = \tfrac{1}{2}\Big[\,\mathrm{tr}\left(R_t^{-1}R\right) - d + \log\frac{\det R_t}{\det R}\Big]. \tag{18}$$

At $t = 1$ we have $R_t = I$ and $\mathrm{tr}(R) = d$, so the divergence reduces to $-\frac{1}{2}\log\det R$. This is the total dependence (multi-information) of P. Since every marginal of Q_t equals that of P in the population, MF is flat on this ladder by construction. A finite sample still differs from another finite sample by sampling error. This is why MF reads about 0.975 here and not 1.000. It varies by 0.003 across the ladder, which is the noise floor of the estimator.

Mean-shift ladder. A second ladder shifts the mean of one column j by δ and leaves R unchanged. Two Gaussians with the same covariance and means that differ by μ have divergence $\frac{1}{2}\,\mu^{\top}R^{-1}\mu$. With $\mu = \delta e_j$ this

gives

$$D_{\mathrm{KL}}(P \,\|\, Q_\delta) \;=\; \tfrac{1}{2}\,\delta^2 \left(R^{-1}\right)_{jj}.$$

The shift leaves every correlation and every correlation ratio within Q_δ unchanged. A metric that only looks at dependence structure would not move, while the joint diverges and MF must fall.

Discrete Bayesian network, $d = 4$. Four columns with $(3, 3, 3, 2)$ levels and the graph $a \to b$, $b \to c$, $\{a, c\} \to d$. Each conditional probability table is drawn once from a symmetric Dirichlet distribution with concentration 0.3. A concentration below 1 gives peaked tables, so the edges are detectable. The joint is an explicit finite array. Every quantity, including $D_{\mathrm{KL}}(P \,\|\, Q)$ and the conditional mutual informations of Theorem 1, is an exact finite sum. The perturbation blends every non-root table toward the uniform distribution,

$$Q_t: \quad p_t\big(x_j \mid x_{\mathrm{Pa}(j)}\big) = (1-t)\, p\big(x_j \mid x_{\mathrm{Pa}(j)}\big) + \frac{t}{|\mathcal{X}_j|},$$

where $|\mathcal{X}_j|$ is the number of levels of X_j. The graph is the same at every rung, and each conditional weakens until the columns are independent at $t = 1$.

This perturbation does not preserve marginals. Blending a child's table toward uniform also pulls the child's marginal toward uniform, unless that marginal was uniform already. MF therefore falls from 0.990 to 0.865 down this ladder, while it stays flat on the copula ladder. The discrete track establishes that FCU responds to an exact divergence. It does not establish that FCU is insensitive to marginals. Only the Gaussian track shows that.

Wide families. *Wide Gaussian:* we draw a random sparse DAG over $d \in \{20, 40\}$ columns with in-degree up to 4 and use its skeleton as the sparsity pattern of a precision matrix, built by the construction above. The model is therefore a Gaussian Markov random field on that skeleton, not a linear structural model simulated along the DAG; the DAG only supplies the edge set. All targets, including the deficit, stay in closed form and are computed from the parent sets the pipeline actually selects. We run parent budgets 5 and 2; the mean deficits over cells are 0.27 and 0.80 nats (Table 4). *Wide discrete:* a random sparse DAG over 20 columns with Dirichlet tables (concentration 0.3). The joint no longer fits in memory. The two joint densities are exact, but the conditional tables of Q^{Pa} are estimated from 10^6 draws of Q, and $D_{\mathrm{KL}}(P \,\|\, Q_t)$ and $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}})$ are Monte Carlo integrals with 10^6 draws. The target of this family therefore carries estimation error as well as integration error; we report the standard error of the latter.

Mixed-type mixture, $d = 6$. A Gaussian-copula mixture with $M = 3$ latent classes, four numeric columns with lognormal, Student-t, beta, and bimodal marginals, and two nominal columns with five levels each. The class means are drawn as $0.8\,\mathcal{N}(0, 1)$ in the latent space, the nominal tables from a symmetric Dirichlet with concentration 5, and each class has its own sparse correlation matrix, built as above (parameter seed 0). The perturbation is the copula attenuation $R_t = (1-t)R + tI$ applied to every class, which preserves every marginal. It has none of the features that make a density unavailable, such as point masses, so its density can be evaluated at any point and its divergence is computed by Monte Carlo from exact densities. This is the mixed-type family with a known target.

Realistic track, without a target. We also run a cell that resembles the benchmark tables, with a t-copula, point masses, missing values, a deterministic dependency, and duplicate rows. The divergence between two such distributions is defined as a divergence between measures, and can be infinite, but neither distribution has a density we can evaluate, so we compute no target. For this track we only check that FCU falls monotonically.

Computing the estimand. For every family we compute the target of the summed excess, $D(P, Q) = D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P)$, beside the full divergence. This target includes the root term $B(P, Q)$ of Section 3.1, which the estimator omits; the copula, shift, and four-column ladders leave the root marginal unchanged, so the omission costs nothing there. For a Gaussian, the chain projection Q^{Pa} is the linear model of each column on its parents, with coefficients taken from the covariance of Q. For a wide network, the conditional tables of Q^{Pa} are estimated from 10^6 draws of Q.

C.3 Calibration results

Table 4 comes from the reporting sweep (`scripts/ground_truth_sweep.py`, five seeds, 100 trees, $n_{\mathrm{train}} = 4000$), and Table 23 (Appendix H) gives the same cells rung by rung.

Bias against the target. The bias of $\widehat{D}$ against $D(P, Q)$ is $+0.000$ on the four-column network, $+0.042$ on the five-column copula, $+0.026$ and $+0.064$ on the twenty- and forty-column Gaussian families, and $+0.010$ on the twenty-column network. The exception is the mean-shift ladder, at -0.73 nats, explained below. A pooled bias can hide local error: the full-KL RMSE reaches 0.53 nats on the forty-column Gaussian family and 1.10 on the shift ladder. Table 23 therefore reports the error rung by rung.

Table 4: Calibration against exact divergences, at reporting settings ($n_{\text{train}} = 4000$, 100 trees, five seeds, eleven rungs per ladder; the wide families pool parent budgets 5 and 2). Bias and RMSE are of the summed excess $\widehat{D}$ in nats, against the full $D_{\text{KL}}(P \,\|\, Q)$ and against its estimand $D_{\text{KL}}(P \,\|\, Q^{\text{Pa}}) - \Delta(P)$ (which includes the unscored root term of Section 3.1); the deficit column is the mean $\Delta(P)$ over cells. ρ is Spearman against the exact KL, negative for FCU (which falls) and shown as its magnitude for the two comparators, allMLU and the pairwise association error. On the shift ladder the numeric score responds logarithmically to the shift (Section 5.2); Table 23 gives the error by rung.

				vs. full KL		vs. estimand		comparator $\lvert\rho\rvert$	
Family	target	cells	ρ_{FCU}	bias	RMSE	bias	deficit	global	pairwise
Gaussian copula, $d = 5$	exact	55	-0.992	+0.058	0.093	+0.042	0.000	0.993	0.937
Gaussian copula, mean shift	exact	25	-0.973	-0.729	1.104	-0.729	0.000	0.968	-0.130
Discrete network, $d = 4$	exact	55	-0.964	-0.002	0.019	+0.000	0.009	0.992	0.990
Gaussian, sparse DAG, $d = 20$	exact	110	-0.987	-0.064	0.206	+0.026	0.273	0.992	0.788
Gaussian, sparse DAG, $d = 40$	exact	110	-0.989	-0.169	0.532	+0.064	0.796	0.992	0.891
Discrete network, sparse DAG, $d = 20$	Monte Carlo	110	-0.972	-0.141	0.463	+0.010	0.955	0.975	0.991
Mixed-type mixture, $d = 6$	Monte Carlo	55	-0.982	+0.034	0.043	–	–	0.961	0.832

Bias against the full divergence. The bias against $D_{\text{KL}}(P \,\|\, Q)$ is -0.06 nats at twenty columns and -0.17 at forty. This is expected. The summed excess does not target the full divergence. A pruned chain cannot see the part of the divergence that only appears when conditioning on omitted predecessors (Appendix A). The difference between the two bias columns is the mean of that part over the ladder, 0.09 and 0.23 nats here. It is not the deficit itself. The deficit column reports how much dependence of P the pruned chain leaves out, 0.27 and 0.80 nats. On a ladder that removes dependence, the gap starts at 0 and reaches the deficit only at the last rung, where Q has no dependence left. Its mean over the ladder is therefore smaller than the deficit.

Ranking. The Spearman correlation between FCU and the exact KL is between -0.96 and -0.99 on every family. On the four narrow ladders FCU falls strictly monotonically in every seed. On the wide families, single seeds are not monotone at neighboring rungs whose exact divergences differ by a few hundredths of a nat. The mean curve is monotone there, and the rank correlation is the claim we make. All of these are calibrations of $\widehat{D}$ in nats. On these ladders the clip of Eq. (15) binds only at the two ends: at zero divergence, where noise puts $\widehat{D}$ just below 0, and past the captured information, where FCU reads 0 by design.

Sample size. On the five-column copula, at three rungs and five seeds per size, the bias of $\widehat{D}$ against its target is $+0.046$, $+0.058$, $+0.055$, $+0.050$, and $+0.030$ nats at $n_{\text{train}} = 1{,}000$, 2,000, 4,000, 8,000, and 16,000. The RMSE is between 0.06 and 0.09 nats. The estimate does not drift with n over this range.

Mean-shift response. The divergence grows from 0 to 3.18 nats across the shift ladder, and the excess follows it in rank (Spearman -0.97, monotone in 5 of 5 seeds) but not in size. Table 4 reports the unclipped $\widehat{D}$, so the clip of Eq. (15) cannot be the cause. The cause is the numeric score of Appendix A.5. The shift changes the conditional mean of one numeric factor and leaves its conditional variance unchanged, so the excess of that factor is $\frac{1}{2}\log(1 + 2\,D_{\text{KL}})$: the measured excess is 0.15, 0.45, 0.74, and 0.97 nats at divergences of 0.20, 0.79, 1.79, and 3.18, against 0.17, 0.48, 0.76, and 1.00 from the closed form, within 0.03 nats at every rung. The mean bias of -0.73 nats is this logarithmic response. FCU reaches 0 only at the last rung, where the excess exceeds the 0.9 nats the chain captured. The pairwise association error does not respond at all (correlation 0.13 in magnitude with the exact KL), as expected for a metric of dependence structure. MF falls monotonically from 0.977 to 0.847.

Binned vs. Gaussian scoring. Scoring a numeric target over K quantile bins as a categorical factor makes Corollary 1 exact for any conditional shape, at the cost of the information the bins discard. We score the same samples with the Gaussian score and with $K = 4$, 8, and 16 bins on four Gaussian families whose targets are exact under both scores: the five-column copula ladder, the mean-shift ladder, a two-column family $Y \mid X \sim \mathcal{N}(x, 1)$ against $\mathcal{N}(x, s^2)$ with s from 1 to 4, and the twenty-column network at $k = 2$ and $k = 5$ ($n = 4{,}000$, five seeds; `scripts/binned_score_study.py`). Each score is judged against its own target: the Gaussian score against the divergence of conditional means, the binned score against the KL of the binned conditionals with the edges it used. The binned score sees what the Gaussian score cannot. On the spread family the Gaussian target is 0 and the score reads 0.03 to 0.16 nats; at $K = 8$ the binned score is within 0.001 nats of its own target on average and registers 0.63 of the 0.92 nats at $s = 4$. On the mean-shift ladder the bias against the full divergence shrinks from -0.73 to -0.26 nats. It pays three ways. Its bias against its own target grows with K, to -0.15 to -0.25 nats at $K = 16$, where the Gaussian score is within 0.07 on every family. The information it captures collapses: at $K = 16$ the real arm keeps 0.36 of its 0.93-nat target on the copula and 0.07 of 0.76 on the twenty-column network, against 0.88 of 0.96 and 0.49 of 0.78 for the Gaussian score,

so FCU sits on a small and noisy denominator and falls 0.25 to 0.33 below its ideal value, against 0.11 to 0.22. And the rungs are ranked worse on every family: Spearman against the exact divergence is -0.98 to -0.99 for the Gaussian score and -0.89 to -0.92 at $K = 16$, and on the shift ladder the binned FCU is clipped to 0 from $\delta = 1$ on. The budget does not matter ($k = 2$ and $k = 5$ agree to 0.01). We therefore keep the Gaussian score as the default, with its stated blind spot, and report the binned score as the ablation of Appendix D.6.

Mixture and realistic track. On the mixture, the Monte Carlo standard error of the divergence is below 0.001 nats. The excess is biased by $+0.034$ nats, with RMSE 0.043 and Spearman -0.98, and it is monotone in every seed. On the realistic track, FCU falls monotonically in the mean over seeds. The spread between seeds is about ten times wider than on the analytic families.

Comparators. allMLU, every column predicted from all the others, ranks the rungs as FCU does ($|\rho| \geq 0.96$ on every family). Its excess overshoots the true divergence when the conditioning set is wide. On the Gaussian families the overshoot is $+0.43$ nats at twenty columns and $+0.98$ at forty, against -0.06 and -0.17 for FCU. On the twenty-column network it is $+1.05$ nats, against -0.14. Part of the overshoot is a population property and not an estimation cost. The all-others conditionals $P(X_j \mid X_{-j})$ do not multiply to a distribution, so their summed divergences have no chain-rule identity; for a dependent pair (X, Y) against the product $Q = P_X P_Y$, the sum is $2\, I_P(X;Y)$ while the joint divergence is $I_P(X;Y)$. Each dependence is counted from both sides. The finite-sample cost of a wide conditioning set adds to this. The pairwise association error follows the divergence on the discrete families ($|\rho| = 0.99$), less well on the Gaussian ones (0.79 at twenty columns), and not at all under a mean shift.

Double-counting derivation. Let $G(P,Q) := \sum_j \mathbb{E}_{X_{-j}\sim P}\big[D_{\mathrm{KL}}\big(P(X_j \mid X_{-j}) \,\|\, Q(X_j \mid X_{-j})\big)\big]$ be the population value of the all-others sum under Bayes-optimal arms, the analogue of $D(P,Q)$ with every conditioning set equal to all the other columns. Take $Q = \prod_j P(X_j)$, the product of P's marginals, so that $D_{\mathrm{KL}}(P \,\|\, Q)$ is the total correlation $C(P) = \sum_j H(X_j) - H(X)$. Under Q the columns are independent, so $Q(X_j \mid X_{-j}) = P(X_j)$ and the j-th term of G is $\mathbb{E}\big[D_{\mathrm{KL}}(P(X_j \mid X_{-j}) \,\|\, P(X_j))\big] = I_P(X_j; X_{-j}) = H(X_j) - H(X_j \mid X_{-j})$. Hence

$$G(P,Q) - D_{\mathrm{KL}}(P \,\|\, Q) \;=\; H(X) - \sum_j H(X_j \mid X_{-j}) \;=\; \sum_j \big[H(X_j \mid X_{<j}) - H(X_j \mid X_{-j})\big] \;\geq\; 0, \tag{19}$$

by the chain rule for $H(X)$ and because conditioning on more columns cannot raise entropy. Every term is a conditional mutual information $I_P(X_j; X_{>j} \mid X_{<j})$, so the gap is zero only when the columns are independent, and otherwise every dependence that runs between X_j and its successors is counted a second time. For a dependent pair, $G = 2\, I_P(X;Y)$ against $D_{\mathrm{KL}} = I_P(X;Y)$. The chain sum has no such term: for the same Q, $D(P,Q) = \sum_j I_P(X_j; \mathrm{Pa}(X_j))$, which equals $D_{\mathrm{KL}}(P \,\|\, Q) - \Delta(P)$ by Theorem 1 and never exceeds the divergence. In the edge-drop study of Appendix C.5, whose last rung is exactly this Q, the measured all-others sum is 3.96 nats against a divergence of 2.60, and the chain sum is 1.94 against its target of 2.00.

Structure recovery. A Chow–Liu tree is undirected, so the root fixes a direction arbitrarily. A true chain $X_a \to X_b \to X_c$ is recovered in reverse about as often as not. The reverse chain is an equally exact factorization of the same joint, and the oracle still scores 0.999 on it. We therefore compare the recovered parents with the *undirected* true graph. Recovery is perfect on the Gaussian path. On the discrete network it misses one edge, and the deficit reports the cost of that edge (0.009 nats, Table 4). Missing a true parent is the error that matters. In the population, extra parents lose no information. They only cost compute. At finite sample size, extra parents can hurt, as Appendix D.4 shows.

C.4 The pairwise blind spot

A pairwise screen cannot detect a parity relation. If y is the parity of several independent fair bits, then y is independent of each single bit, and the whole $\log 2 \approx 0.693$ nats of information appears only when all bits are used together. This is the standard counterexample to pairwise pruning. To examine how selection behaves when an interaction acquires a detectable pairwise component, we mix the parity conditional with a conditional that depends on one bit.

Construction. Let $x_0, \ldots, x_{k-1}$ be independent fair bits and let

$$P(y = 1 \mid x) \;=\; (1-\alpha)\,\big[x_0 \oplus \cdots \oplus x_{k-1}\big] + \alpha\, x_0.$$

At $\alpha = 0$ this is pure parity of order k. At $\alpha = 1$ it is $y = x_0$, which a pairwise screen finds at once. For $k \geq 2$ and any α, a single bit x_i with $i \geq 1$ tells nothing about y: given x_i, both the parity term and x_0 are still fair coins, so $P(y = 1 \mid x_i) = \frac{1}{2}$ and $I(y; x_i) = 0$. Only x_0 is visible to a pairwise screen. The parameter α

controls how visible the relation is. It also changes the noise: given all k bits, y has conditional entropy $\frac{1}{2}h(\alpha)$, which is 0 at both ends and largest at $\alpha = \frac{1}{2}$.

The deficit in closed form. Let $h(p) = -p\log p - (1-p)\log(1-p)$ be the binary entropy. Given all k bits, y is deterministic when the parity equals x_0, and has entropy $h(\alpha)$ otherwise. The two cases are equally likely, so $I(y; x_0, \ldots, x_{k-1}) = \log 2 - \frac{1}{2}h(\alpha)$. Given x_0 alone, $P(y = 1 \mid x_0) = \frac{1-\alpha}{2} + \alpha x_0$, so $I(y; x_0) = \log 2 - h\big(\frac{1+\alpha}{2}\big)$. The bits are independent of each other, so the total dependence of the joint is $I(y; x_0, \ldots, x_{k-1})$, whatever the ordering of the columns. A chain that captures only the relation between y and x_0, in either direction, therefore discards the difference,

$$\Delta(\alpha) \;=\; h\Big(\frac{1+\alpha}{2}\Big) - \frac{1}{2}\,h(\alpha), \tag{20}$$

which does not depend on k. It equals $\log 2$ at $\alpha = 0$, 0.526 at $\alpha = 0.1$, and 0 at $\alpha = 1$. Eq. (20) reproduces the "discarded" columns of Table 5 under mRMR at every α. In other words, mRMR captures the x_0 relation and nothing else.

Protocol. Every cell of Table 5 is an oracle cell: the synthetic data are drawn from the true network, so a high FCU is the correct answer, and any FCU below 1 is caused by the pruning or the estimator, never by a generator. What the table measures is the coverage of the selected tasks, through the deficit. Whether an incomplete chain lets an unfaithful generator pass is a separate question, answered with a projected generator below. We use five seeds per cell, $n = 4000$, and the campaign settings.

Table 5: FCU and the deficit of Theorem 1 as a function of visibility α and interaction order k (mean over five seeds). *Discarded* is the deficit in nats; its maximum is $\log 2 = 0.693$. *Undef.* is the share of seeds in which the chain captured no information ($\widehat{I} \le 0$), so that FCU is undefined. The last two columns give the deficit under the tree-gain screen. At $k = 2$ the chain over the three columns is complete, the deficit is 0, and FCU is about 1 at every α, so that order is omitted.

	$k=3$, mRMR			$k=4$, mRMR			tree-gain, discarded	
α	FCU	undef.	discarded	FCU	undef.	discarded	$k=3$	$k=4$
0.0	0.333	0.4	0.693	0.000	0.8	0.693	0.693	0.693
0.1	**0.981**	0.0	**0.526**	**0.884**	0.0	**0.526**	0.529	0.528
0.2	0.991	0.0	0.423	0.976	0.0	0.423	0.350	0.427
0.3	1.000	0.0	0.342	0.988	0.0	0.342	0.292	0.342
0.4	1.000	0.0	0.274	0.992	0.0	0.274	0.126	0.274
0.5	1.000	0.0	0.216	0.995	0.0	0.216	0.130	0.216
0.6	1.000	0.0	0.164	0.998	0.0	0.164	0.164	0.164
0.8	0.998	0.0	0.075	0.997	0.0	0.075	0.075	0.075
1.0	1.000	0.0	0.000	1.000	0.0	0.000	0.000	0.000

Partly visible interactions. At $\alpha = 0$ the selector finds no parent for y in most seeds. The chain captures nothing, and FCU is undefined in 40% of seeds at $k = 3$ and 80% at $k = 4$. Where it is defined, it is low. A reader is warned in either case. The problem is the middle of the range. At $\alpha = 0.1$ and $k = 3$, the selected chain omits 0.526 nats: 99% of the 0.531 nats of dependence present, and 76% of the maximum $\log 2$. FCU is 0.98 there, which is correct for a faithful generator, but it is computed on a chain that covers almost none of the relation. At $\alpha = 0.3$ it is 1.00 with 0.342 nats omitted. The selector takes x_0, and a model that sees only x_0 predicts equally well in both arms. Of the 16 cells whose chain is incomplete (deficit ≥ 0.05 nats), 13 have FCU > 0.9. The deficit reports the omitted information in every one of those cells, which is why the two are reported together.

A generator that drops the interaction. Let Q be P projected onto the chain that a pairwise screen selects: the bits stay independent and y depends on x_0 alone, with $Q(y = 1 \mid x_0) = P(y = 1 \mid x_0)$. This Q is exact on the conditional $y \mid x_0$ and on every marginal, so a chain that selects only x_0 gives it an ideal FCU of 1, while $D_{\mathrm{KL}}(P \,\|\, Q) = \Delta(\alpha)$ by construction (0.526 nats at $\alpha = 0.1$). We score it with the same protocol (Table 6). At $k = 3$ it receives FCU 0.79 at $\alpha = 0.1$, 0.98 at $\alpha = 0.2$, and 0.99 or more from $\alpha = 0.3$ on, while its divergence from P is 0.53, 0.42, and 0.34 nats and below; at $k = 4$ the pattern is the same from $\alpha = 0.1$. Of the 16 cells with an incomplete chain, 12 score this generator above 0.9. At $k = 2$ the three-column chain is complete, the projection removes a scored conditional, and FCU falls to 0.00 at $\alpha = 0$ and 0.36 at $\alpha = 0.5$: the same generator is caught wherever the chain covers the interaction. This is the failure an incomplete chain cannot see, and the deficit is what marks the cells where it can occur.

Table 6: FCU of the projected generator, which reproduces $y \mid x_0$ and every marginal and drops the parity interaction (mean over five seeds, mRMR). $D_{\mathrm{KL}}(P \,\|\, Q)$ is exact and equals the omitted information of Eq. (20). At $k = 2$ the chain is complete and the missing interaction is a scored conditional, so FCU falls; at $k \geq 3$ the chain selects x_0 alone and the generator passes.

α	$D_{\mathrm{KL}}(P \,\|\, Q)$	FCU, $k = 2$	FCU, $k = 3$	FCU, $k = 4$
0.0	0.693	0.003	0.667	0.000
0.1	0.526	0.016	0.792	0.861
0.2	0.423	0.044	0.982	0.950
0.3	0.342	0.124	0.990	0.989
0.4	0.274	0.232	0.994	0.981
0.5	0.216	0.357	0.996	0.992
0.6	0.164	0.538	1.000	0.991
0.8	0.075	0.831	0.998	0.997
1.0	0.000	1.000	1.000	1.000

The tree-gain screen. Ranking candidates by the split gain of one boosted model on all predecessors can see interactions. At $k = 3$ it recovers both parity parents in some seeds from $\alpha = 0.2$ on. At $\alpha = 0.4$ it halves the discarded information (0.126 against 0.274 nats). It does not remove the blind spot. At $k = 4$ the deficit is unchanged, because the ordering is still pairwise and often places a parity parent after y, where no screen can select it. The remaining weakness is the ordering.

C.5 The edge-drop study: generators that fail on one edge

The parity sweep shows the blind spot at its extreme, an interaction with no pairwise trace at all. Here we measure it on a graded family. We take the random sparse network of Appendix C.2 at $d = 12$ (cardinalities 2 to 4, in-degree up to 4), which has 22 edges carrying 3.38 nats of conditional mutual information in total. Removing one edge and refitting the child's conditional on its remaining parents is the I-projection of P onto the family without that edge, so $D_{\mathrm{KL}}(P \,\|\, Q)$ equals the conditional mutual information of the dropped edge, exactly. Such a Q keeps every marginal and every other conditional: it is a generator that fails on one dependence and nowhere else. We score each of the 22 single-edge projections, and the projections that drop the m strongest edges together, under the reporting settings ($n = 4{,}000$, five seeds) at budgets $k = 2$ and $k = 5$, with the chain recovered from the training sample as in the main protocol (`scripts/projection_ladder.py`). The exact deficit of the recovered chain is computed in every cell, so the target $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P)$ is known.

Table 7: The edge-drop study on the twelve-column network of Appendix C.5 (22 edges). Dropping an edge from Q is an exact projection whose divergence is that edge's conditional mutual information, so KL is exact. *Upper block:* the m strongest edges dropped together; *target* is $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P)$, exact, and $\widehat{D}$ the measured summed excess, both in nats, mean over five seeds. *Lower block:* the single-edge rungs pooled by whether the recovered chain covers the dropped edge in every seed, in none, or in some; sums over the edges in each group.

		$k = 2$			$k = 5$		
m dropped	KL	target	$\widehat{D}$	FCU	target	$\widehat{D}$	FCU
0	0.000	-0.001	-0.003	1.00	0.000	-0.005	1.00
1	0.547	0.605	0.584	0.63	0.654	0.625	0.68
2	0.895	0.611	0.578	0.64	0.731	0.688	0.65
3	1.191	1.049	1.005	0.37	1.099	1.061	0.45
5	1.458	1.274	1.238	0.22	1.359	1.321	0.32
8	1.993	1.438	1.400	0.13	1.710	1.663	0.14
12	2.268	1.772	1.722	0.00	2.082	2.020	0.00
22	2.603	1.639	1.597	0.00	2.001	1.936	0.01
single edges	KL	target	$\widehat{D}$	edges	target	$\widehat{D}$	edges
covered	1.829 / 2.098	1.955	1.863	9	2.242	2.129	12
partial	0.400 / 1.126	0.338	0.321	4	0.836	0.761	7
uncovered	1.150 / 0.155	0.335	0.314	9	0.046	0.027	3

The estimator tracks its target. Over the 60 rungs, $\widehat{D}$ minus its target lies between -0.065 and $+0.013$ nats. This is the accuracy of Table 4, now on generators that fail in one place.

Effect of one uncovered edge. At $k = 2$ the recovered chain covers 9 of the 22 edges in every seed and misses 9 in every seed (Table 7, lower block). The covered edges carry 1.83 nats and $\widehat{D}$ registers 1.86. The uncovered edges carry 1.15 nats, their target is 0.34, and $\widehat{D}$ registers 0.31. The second-strongest edge of the network (0.35 nats) is among them and moves FCU from 1.00 to 0.97. The share that is seen is a property of the target and not of the estimator: an omitted edge shows only through the covered conditionals it changes. At $k = 5$ three edges of 0.16 nats in total stay uncovered, and the covered 2.10 nats register 2.13.

Dose response. Dropping the 1, 3, 5, 8, and 12 strongest edges, FCU reads 0.63, 0.37, 0.22, 0.13, and 0.00 at $k = 2$, and 0.68, 0.45, 0.32, 0.14, and 0.00 at $k = 5$ (Table 7, upper block). Against the full divergence the shortfall grows to -0.55 and -0.25 nats at twelve edges and to -1.0 and -0.67 when every edge is dropped and Q is the product of marginals. The target itself is not monotone in m: at twelve dropped edges $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) = 2.73$ exceeds $D_{\mathrm{KL}}(P \,\|\, Q) = 2.27$, so projecting Q onto the chain can move it further from P, and $\widehat{D}$ follows the target in that case too. What the ladder shows is that the blind spot has the size the theory gives it. What it cannot show is which edge is missing, since the deficit reports only the total that the chain omits.

allMLU in the same study. allMLU (Section 5.2) is scored on the $k = 5$ ladder beside $\widehat{D}$ (Table 8); with no stated target of its own (Appendix C.3), it is judged against the exact divergence. On the cumulative track it reads 0.54–1.52 times the divergence across the ladder, an overshoot of 1.36 nats at all 22 edges dropped, where $\widehat{D}$ reads 0.74–1.14 times the divergence and stays within 0.065 nats of its target throughout. Both rank the rungs correctly, but on the 22 single edges they miss in opposite places: $\widehat{D}$ under-registers the three edges its chain does not cover (0.16 nats read as 0.03), while allMLU registers those (0.27) and under-registers the twelve edges every seed covers (1.84 against 2.10; on the strongest edge, 0.55 nats, it reads 0.30 where $\widehat{D}$ reads 0.63), because a dropped parent explains little of a child once the child's own children are also in the conditioning set. Across the 22 edges allMLU errs by 0.110 nats RMSE (Spearman 0.69) against $\widehat{D}$'s 0.067 (0.91); $\widehat{D}$ reads the same quantity on every rung, and allMLU has none to be checked against.

Table 8: allMLU, every column predicted from all the others, in the edge-drop study of Table 7 at $k = 5$ (22 edges, five seeds). *Global* is the summed excess when every column is predicted from all the others, and *ratio* divides each summed excess by the exact KL. $\widehat{D}$ has the target of Theorem 1; allMLU has none. *Lower block:* the single-edge rungs pooled by coverage as before, with the pairwise association error (mean absolute difference of the Linfoot matrices) as a third comparator.

m dropped	KL	target	$\widehat{D}$	ratio	global	ratio	assoc. error
0	0.000	0.000	-0.005	–	-0.010	–	0.016
1	0.547	0.654	0.625	1.14	0.297	0.54	0.080
2	0.895	0.731	0.688	0.77	0.895	1.00	0.087
3	1.191	1.099	1.061	0.89	0.966	0.81	0.123
5	1.458	1.359	1.321	0.91	1.411	0.97	0.134
8	1.993	1.710	1.663	0.83	2.604	1.31	0.145
12	2.268	2.082	2.020	0.89	2.926	1.29	0.161
22	2.603	2.001	1.936	0.74	3.964	1.52	0.185
single edges	KL	target	$\widehat{D}$	ratio	global	ratio	edges
covered	2.098	2.242	2.129	1.01	1.839	0.88	12
partial	1.126	0.836	0.761	0.68	1.512	1.34	7
uncovered	0.155	0.046	0.027	0.17	0.274	1.76	3

D Additional benchmark results

D.1 Two more views of what single-task utility misses

Figure 4 in the main text shows the mechanism on one pair and the consequence on tasks FCU never scores. Figures 6 and 7 show the same contrast from two more sides, using only numbers reported elsewhere in this paper: every pair the single-task metric cannot separate, and where the two-sample test stops ranking and FCU does not.

D.2 Sensitivity of the matched-pair comparison

The headline comparison of Section 6.1 matches generator pairs whose TSTR AUC differs by at most 0.03. That band width is a choice, so Table 9 reports four widths. The reference is the median FCU gap among *all* within-dataset pairs, which is 0.370.

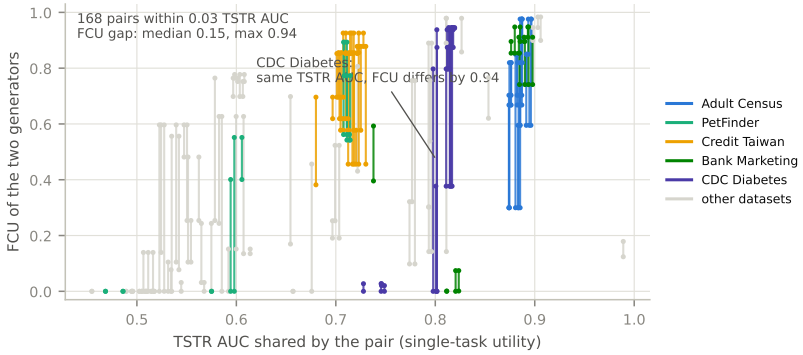


Figure 6: Every within-dataset pair of generators whose TSTR AUC agrees to within 0.03 (cell means over nine runs; 168 of 656 pairs on the eleven classification tables), drawn as a segment from the FCU of one generator to the FCU of the other at the pair's shared TSTR AUC. Single-task utility does not separate the two ends of any segment. The five featured tables keep their color; the rest are gray.

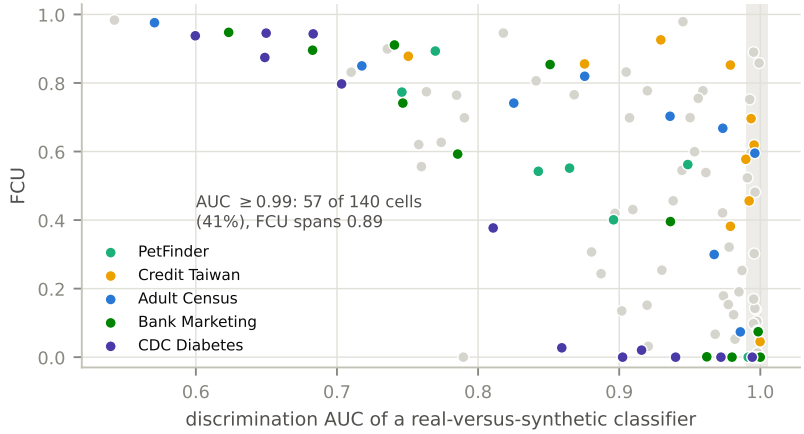


Figure 7: FCU against the discrimination AUC of a real-versus-synthetic classifier, one mark per cell (140 cells with the classifier defined). In the shaded band the classifier is at or above AUC 0.99 and can no longer order generators; FCU still spans 0.89 across those 57 cells. Over all 140 cells the two agree at $\rho = 0.75$ (Section 6.2).

Table 9: FCU gap between generator pairs matched on TSTR AUC, by band width. The pool is 656 within-dataset pairs on the eleven tables with a classification target. *Share surviving* is the band's median gap divided by the median over all pairs (0.370). Every band differs from a resampled all-pairs null at $p < 0.0002$ (5,000 permutations). The standardized effect d is negative because matching selects generators that resemble each other.

Band (abs. TSTR AUC)	Pairs	Median gap	Gap > 0.25	Max	Share surviving	d
0.01	73	0.145	32%	0.747	0.39	−0.73
0.03	168	0.148	34%	0.938	0.40	−0.72
0.05	242	0.160	39%	0.945	0.43	−0.68
0.10	390	0.254	51%	0.979	0.69	−0.38

Matching on TSTR AUC removes three fifths of the FCU spread at the 0.03 band. The rest survives, and so do the extremes: the widest matched gap, 0.938, is 95% of the widest unmatched gap (0.984). The surviving share grows with the band width, as it must.

The matched-pair statistic depends on a threshold, so we also fit a model. We regress cell-mean FCU on TSTR AUC with dataset fixed effects, over the 125 cells with a classification target. The slope is 1.96 and $R^2 = 0.45$. The residual standard deviation is 0.28 and the residual interquartile range is 0.42. At a fixed dataset and a fixed single-task utility, generators still differ in FCU by that much (`scripts/posthoc_review2.py`).

D.3 PER-FACTOR DIAGNOSTICS

FCU is a sum over named prediction tasks, so a low score can be traced to specific conditionals. Figure 8 compares TABSYN and GAUSSIANCOPULA on ADULT_CENSUS. For each factor it plots the information captured by the real arm against the information captured by the synthetic arm, both relative to climatology and in nats.

TABSYN's factors lie on the diagonal. Whatever a model can learn from real data, it learns almost as well from TABSYN's data. GAUSSIANCOPULA's factors fall to zero and below. The clearest case is the strongest conditional of the table, `education-num` given `education`. The real arm captures 2.03 nats. The synthetic arm scores 0.31 nats *worse* than climatology, because the synthetic table breaks a deterministic recoding that a Gaussian dependence model cannot represent. `marital-status` falls from 0.72 to 0.06 nats, and `sex` from 0.27 to 0.00. Both generators face the same factors and the same real arm, so the contrast comes from the synthetic data alone. The discrimination AUC is a single number, saturated at 1.00 here. TSTR does not look at these factors. Neither gives a diagnosis this specific.

Figure 4a in the main text is the same view on CDC_DIABETES, drawn as bars. There the real arm captures 0.116 nats on `Education` and 0.074 on `Income`. TABDDPM keeps 0.111 and 0.073 of them; CDTD scores 0.542 and 0.201 nats *worse* than climatology on the same two factors, which is where its FCU = 0.00 comes from while its TSTR AUC stays within 0.03 of TABDDPM's.

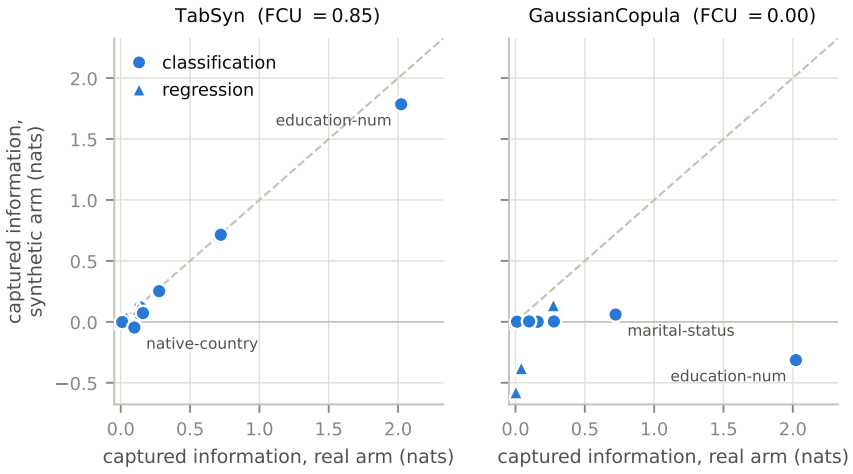


Figure 8: Per-factor diagnostics on ADULT_CENSUS (sampling seed 420, repeat 0). Each mark is one conditional factor. On the dashed line, the synthetic arm captures as much information as the real arm. The solid line is climatology. TABSYN (left) preserves most learnable conditionals; the marks below the diagonal are its deficits, and its FCU is 0.85. GAUSSIANCOPULA (right) loses almost all of them, most visibly the deterministic mapping from `education` to `education-num`. Both reproduce the per-column marginals (MF 0.947 and 0.905).

D.4 WHAT THE PRUNING COSTS ON REAL TABLES

Theorem 1 gives the cost of pruning in the population, $\Delta(P) = \sum_j I(X_j; R_j \mid \mathrm{Pa}(X_j))$. On real data this quantity is unknown. We report instead the *full-versus-pruned gain* $\hat{G} = \hat{I}_{\mathrm{full}} - \hat{I}_{\mathrm{pruned}}$ in nats: we fit the same chain with every factor conditioned on *all* its predecessors and compare the information the real arm captures with and without pruning (Table 10). $\hat{G}$ uses no synthetic data and is a property of the table and the estimator, not of any generator. It is not an estimate of $\Delta(P)$ with an error guarantee. Its population value is $\Delta(P)$ plus the difference between the learner's excess risk under the two conditioning rules, which can have either sign, so a small or negative $\hat{G}$ does not show that little dependence was omitted; a wide learner can fail to recover information that is there. The all-predecessors chain is expensive, so we fitted it on the four tables of the subset run.

Gain by table. The full-chain predictor improves held-out information substantially on BEIJING_PM25, whose hourly weather readings depend jointly on several others, and not on the other three tables. This suggests that additional conditioning helps the chosen learner on BEIJING_PM25; it does not identify the population information loss on the other tables. We report $\hat{G}$ beside FCU and do not fold it into the score.

Table 10: The full-versus-pruned gain $\hat{G}$. $\widehat{I}$ is the information captured by the real arm under each conditioning rule, in nats, and $\hat{G} = \widehat{I}_{\text{full}} - \widehat{I}_{\text{pruned}}$. Mean ± SD over the nine runs of a cell, averaged over entries.

Dataset	Factors	mean |Pa|	$\widehat{I}$ pruned	$\widehat{I}$ full	$\hat{G}$ (nats)
DIABETES_130US	42	1.38	3.18	2.90	$\mathbf{-0.278 \pm 0.085}$
ADULT_CENSUS	14	1.71	4.43	4.30	$\mathbf{-0.132 \pm 0.018}$
PETFINDER_TAB	17	1.51	2.78	2.75	-0.023 ± 0.066
BEIJING_PM25	11	1.91	3.24	4.89	1.654 ± 0.200

Pruning cost and sample size. $\hat{G}$ is negative on three of the four tables, and on ADULT_CENSUS and DIABETES_130US in every run. Conditioning on all predecessors captures *less* information on held-out rows than the pruned chain does. This does not contradict Theorem 1. The theorem concerns population distributions, where extra conditioning never loses information. With finite data, estimating a wide conditional has a statistical cost, and that cost can exceed the information gained. On those tables the pruned chain is both the cheaper and the better estimator.

Table 11: The full-versus-pruned gain $\hat{G}$ where the deficit is known. $\widehat{I}$ is the information the real arm captures under each conditioning rule, $\hat{G} = \widehat{I}_{\text{full}} - \widehat{I}_{\text{pruned}}$, and $\Delta(P)$ is the deficit of the recovered chain, exact on the twelve-column network and by Monte Carlo on the twenty-column one. Nats, mean over five seeds; ± is the SD of $\hat{G}$ over seeds. The synthetic arm is an oracle sample from P in every cell.

Network	k	n	$\widehat{I}$ pruned	$\widehat{I}$ full	$\hat{G}$	$\Delta(P)$	$\hat{G} - \Delta(P)$
sparse DAG, $d = 12$	2	1,000	1.541	2.298	0.757 ± 0.085	1.021	-0.264
	2	4,000	1.599	2.441	0.842 ± 0.025	0.962	-0.120
	2	16,000	1.686	2.529	0.843 ± 0.052	0.924	-0.081
	5	1,000	1.816	2.298	0.482 ± 0.154	0.720	-0.238
	5	4,000	1.941	2.441	0.500 ± 0.078	0.601	-0.100
	5	16,000	2.201	2.529	0.328 ± 0.107	0.387	-0.060
sparse DAG, $d = 20$	2	1,000	4.080	4.518	0.437 ± 0.115	1.207	-0.769
	2	4,000	4.321	5.048	0.727 ± 0.031	1.095	-0.367
	2	16,000	4.293	5.220	0.927 ± 0.066	1.091	-0.164
	5	1,000	4.378	4.518	0.140 ± 0.128	0.829	-0.689
	5	4,000	4.578	5.048	0.470 ± 0.080	0.819	-0.349
	5	16,000	4.638	5.220	0.582 ± 0.162	0.720	-0.138

Calibration where the deficit is known. On the two random sparse networks of Appendix C.5, with an oracle sample from P as the synthetic arm, $\hat{G}$ falls short of $\Delta(P)$ in every cell (Table 11): by 0.06 to 0.26 nats on the twelve-column network and by 0.14 to 0.77 on the twenty-column one, and the shortfall roughly halves for every fourfold increase in n. The pruned $\widehat{I}$ is within 0.01 nats of its exact value at $n = 16{,}000$. The shortfall belongs to the full chain, which captures 2.30 of 2.60 nats at $n = 1{,}000$ and 2.53 at $n = 16{,}000$ on the twelve-column network. $\hat{G}$ is never negative on these networks, whose conditionals are small tables. The negative values on real tables are therefore a cost of wide conditioning that the oracle networks do not exhibit, and $\hat{G}$ reads as a downward-biased diagnostic of $\Delta(P)$ rather than as an estimate of it.

D.5 HELD-OUT COLUMNS AND THE ALL-MODEL UTILITY COMPARATOR

Held-out columns. For each dataset and repeat, we split the columns in half using the repeat's seed. FCU is computed on half A alone, with its own association matrix, ordering, and parent sets. It never scores a column of half B. The held-out tasks predict each column of half B from all other columns. They use the same arms, calibration, scores, and aggregation. The experiment ran on four tables, with 432 runs in total; Table 12 gives the Spearman correlation with held-out utility, per table and pooled.

Pooled over the 297 runs on which all three are defined, FCU on the scored half predicts utility on the unscored half at 0.72, benchmark TSTR at 0.60, and allMLU at 0.94; over all 432 runs, where TSTR AUC is undefined on BEIJING_PM25, the numbers are 0.71, 0.60, and 0.93. Within single tables the picture is mixed. TSTR predicts better than FCU on ADULT_CENSUS, where the benchmark target is one of the most connected columns, and worse on PETFINDER_TAB; pooled superiority is therefore not uniform superiority. allMLU predicts best everywhere, and its form explains why. The held-out tasks predict each column of B from every other column, which is exactly the task that allMLU scores, and they condition on the half-A columns that FCU's structure was

Table 12: Spearman correlation with utility on the held-out half B, per table and pooled over runs. TSTR AUC is undefined on the regression table and on multiclass runs where a class is absent, so the first pooled row mixes denominators; the second restricts every column to the 297 runs on which all three are defined.

Dataset	runs	FCU on half A	benchmark TSTR AUC	allMLU
ADULT_CENSUS	108	0.595	0.777	0.898
BEIJING_PM25	108	0.643	n/a	0.956
DIABETES_130US	108	0.691	0.664	0.926
PETFINDER_TAB	108	0.784	0.726	0.969
pooled	432	0.707	0.595	0.928
pooled, common runs	297	0.717	0.595	0.936

estimated on. The B tasks are a subset of allMLU's own factors, so its 0.93 is partly a part-whole comparison rather than an independent prediction. The mean skill on the B tasks is 0.332 for the real arm and 0.166 for the synthetic arm. We also considered the root columns of the chain as a held-out task family. They do not form one, because the chain has a single root in practice.

allMLU on the full tables. We implement allMLU, the global utility of Jiang et al. (2026), directly. Each column in turn is the target, and all other columns are the conditioning set; this is not the all-predecessors chain of Appendix D.4, which is an ordered chain and has the identity of Theorem 1. Both arms are otherwise scored exactly as FCU scores its own factors, so the conditioning set is the only difference. It ran on the same four tables with twelve entries each; Table 13 gives the rank agreement.

Table 13: FCU against allMLU, twelve entries per dataset (cell means over nine runs). The rank agreement is high and the leading generator is the same on every table. The conditioning sets differ in size by up to an order of magnitude.

Dataset	d	Spearman ρ	ranks moved	max shift	mean $\lvert\mathrm{Pa}\rvert$ vs $d-1$
ADULT_CENSUS	15	0.986	4/12	1	1.71 vs 14
BEIJING_PM25	12	0.923	8/12	3	1.91 vs 11
DIABETES_130US	43	0.991	2/12	1	1.38 vs 42
PETFINDER_TAB	18	1.000	0/12	0	1.51 vs 17

Pooled over the 48 cells, the two metrics agree at Spearman 0.976. They agree least on BEIJING_PM25, which is also the table with the largest full-versus-pruned gain (Table 10). This is the situation the gain is meant to flag. A reader who only needs a ranking of generators on tables of this width may use either metric. FCU adds three things: a lower cost, a diagnostic of what its pruning discards, and the ability to trace a failure to a named conditional with a small parent set.

Conditioning width on all twelve tables. The cost difference is arithmetic on the stored factorization, not a timing measurement (Table 14). FCU conditions each factor on 1.2 to 3.3 parents on average, 1.75 over the corpus. allMLU and single-task TSTR both condition every task on the other $d-1$ columns, and the discriminator reads all d; the three differ from one another in task count, not in width. The ratio of widths runs from 3.0 on MAGIC, whose columns are densely dependent (mean 3.3 parents), to 30.4 on the 43-column DIABETES_130US, with mean 11.5 and median 10.4. Two caveats apply. The number of fitted models is the same order for FCU and allMLU ($|\mathcal{C}|$ against d tasks, two arms each), so the saving is in the width of each model, not in how many are fitted. FCU also pays once per split for the association matrix and the parent screen (Appendix B.1), which the comparators do not; that cost is shared by every generator scored on the split. Wall-clock time per method was not measured. The campaign cost in Appendix F.4 bundles FCU with every baseline in one job.

The categorical factors alone. Theorem 1 is exact for categorical factors and an approximation for numeric ones (Appendix A.5). To see how much of the benchmark the exact part of the identity carries, we rebuild FCU from the categorical factors alone, from the per-factor records of the same runs, and compare it with the full score (Table 15). The categorical factors carry 7% of the captured information on MAGIC and 96% on PETFINDER_TAB. Within tables the two scores rank the generators at Spearman 0.81 to 1.00, pooled 0.93 over 128 cells, and the leading generator is the same on nine of twelve tables. The three that change, BEIJING_PM25, CREDIT_DEFAULT_TAIWAN, and HOME_CREDIT, are tables where the numeric factors carry 60% to 70% of the information, so the conclusions of Section 6 do not rest on the numeric approximation, and where the numeric factors decide a table they do so as the regression utility of Section 3.2.2, not as a divergence.

Table 14: Conditioning width of FCU against the comparators, per dataset. $|\mathcal{C}|$ is the number of scored factors and mean $|\mathrm{Pa}|$ their mean parent count, averaged over the cells of the benchmark grid. allMLU and TSTR both condition every task on the other $d-1$ columns and differ only in task count (d and 1); the real-versus-synthetic discriminator reads all d. The ratio is $(d-1)$ over mean $|\mathrm{Pa}|$. The number of fitted models is the same order for FCU and allMLU ($|\mathcal{C}|$ against d tasks, two arms each), so the saving is per-model width, not model count; and FCU pays once per split for the association matrix and the parent screen, which the comparators do not. No wall-clock time was measured per method.

Dataset	d	$\lvert\mathcal{C}\rvert$	mean $\lvert\mathrm{Pa}\rvert$	allMLU, TSTR	discriminator	ratio
ADULT_CENSUS	15	14	1.71	14	15	8.2
BANK_MARKETING	15	14	1.50	14	15	9.3
BEIJING_PM25	12	11	1.91	11	12	5.8
CDC_DIABETES	22	21	2.14	21	22	9.8
CREDIT_DEFAULT_TAIWAN	24	23	1.90	23	24	12.1
CRITEO_1M	17	16	1.46	16	17	11.0
DIABETES_130US	43	42	1.38	42	43	30.4
HOME_CREDIT	17	16	1.44	16	17	11.1
MAGIC	11	10	3.30	10	11	3.0
MIMIC_III	11	10	1.23	10	11	8.1
PETFINDER_TAB	18	17	1.51	17	18	11.3
SUPPORT2	30	29	1.57	29	30	18.4
mean			1.75			11.5
median			1.54			10.4

Table 15: FCU from the categorical factors alone against the full score, per table. Both are rebuilt from the per-factor records of the same runs (3 runs per cell), with the control excluded. *share* is the fraction of the captured information $\widehat{I}$ that the categorical factors carry; ρ is the Spearman rank correlation of the generators under the two scores; *shift* is the largest change of a cell's FCU. Pooled over the 128 cells, $\rho = 0.93$ and the leading generator is the same on 9 of 12 tables.

Dataset	cat. / num. factors	share	ρ	shift	leader	categorical leader
ADULT_CENSUS	9 / 5	0.90	0.852	0.35	CDTD	CDTD
BANK_MARKETING	7 / 7	0.29	0.948	0.59	CDTD	CDTD
BEIJING_PM25	3 / 8	0.40	0.975	0.23	ForestDiffusion	**TabSyn**
CDC_DIABETES	18 / 3	0.76	0.939	0.04	TabDiff	TabDiff
CREDIT_DEFAULT_TAIWAN	10 / 13	0.38	0.936	0.33	CDTD	**ForestDiffusion**
CRITEO_1M	11 / 5	0.89	0.962	0.07	TabSyn	TabSyn
DIABETES_130US	37 / 5	0.88	1.000	0.09	TabDiff	TabDiff
HOME_CREDIT	13 / 3	0.30	0.914	0.55	TabSyn	**ForestDiffusion**
MAGIC	1 / 9	0.07	0.900	0.48	CDTD	CDTD
MIMIC_III	8 / 2	0.52	0.990	0.30	TabDiff	TabDiff
PETFINDER_TAB	14 / 3	0.96	0.962	0.19	TabDiff	TabDiff
SUPPORT2	10 / 19	0.33	0.809	0.33	CDTD	CDTD

D.6 Ablations

Every arm differs from the default in exactly one respect. Each is run on ADULT_CENSUS and PETFINDER_TAB under the full protocol: three seeds, three repeats, and twelve entries, for 24 cells. Table 16 reports the rank correlation of each arm's per-cell mean FCU with the default, and the mean and largest absolute shift. Every arm is a pinned configuration of the released library.

Structure. The association measure, the ordering, and the parent screen do not drive the results. Every structure arm ranks the cells at $\rho \geq 0.989$ and shifts them by 0.01 to 0.02 on average. The *type-pair association blocks* arm uses a different measure for each type pair: $|\rho|$ for numeric pairs, the symmetrized Theil coefficient for categorical pairs, and the correlation ratio η for mixed pairs. Its ordering is by mean association, and its parents are the top-k by association.

Parent budget. A larger fixed budget lowers the level of FCU monotonically. The mean FCU goes from 0.402 at k=1 through 0.386, 0.370, and 0.353 to 0.330 at k=10, against 0.394 for adaptive k (mean parent count 1.6). On these tables, wider conditionals cost the real arm more than they capture, and the synthetic arm

Table 16: Ablations on ADULT_CENSUS and PETFINDER_TAB, 24 cells per arm. CatBoost and linear are compared on the 22 cells they completed (see the note on failures). Spearman is against the per-cell FCU of the default arm.

Arm	Spearman vs default	mean $\lvert\Delta\text{FCU}\rvert$	max $\lvert\Delta\text{FCU}\rvert$
Structure			
distance correlation	1.000	0.008	0.031
tree-gain parent screen	0.996	0.011	0.035
type-pair association blocks	0.989	0.018	0.086
random-ordering ensemble	0.992	0.013	0.084
Fixed parent budget (default: adaptive k)			
k=1	1.000	0.008	0.034
k=2	1.000	0.008	0.055
k=3	0.999	0.024	0.070
k=5	0.998	0.041	0.138
k=10	0.996	0.065	0.203
Model family (default: LightGBM)			
XGBoost	0.999	0.009	0.081
CatBoost	0.999	0.015	0.141
random forest	0.996	0.020	0.286
linear	0.942	0.103	**0.336**
Scores and aggregation (default: log score, sums in nats)			
quantile-binned regression	0.962	0.051	0.311
gated mean of per-factor ratios	0.936	0.131	0.560
skill scores, per-factor clip	0.907	0.170	**0.642**

loses more than the real one, so FCU falls. The ranking is preserved ($\rho \geq 0.996$). Adaptive k is the default because it ties the number of parents of each factor to what the real data support.

Model family. The three tree families agree with LightGBM at $\rho \geq 0.996$ and shift the cells by at most 0.02 on average. The linear arm reorders the cells ($\rho = 0.942$) and shifts them by up to 0.34. Conclusions are therefore stable across tree families, while a linear estimator measures something else.

Scores and aggregation. These arms change the metric the most. The *skill-score arm* scores each factor with the Brier skill score or R^2 and clips per factor, $\min(U_j^{\text{syn}}, U_j^{\text{real}})$. It ranks the cells at $\rho = 0.907$ against the default and shifts one cell by 0.64. Its mean level is 0.563 against 0.394, because the per-factor clip and the skill normalization both favor a generator at low signal. The *gated mean of per-factor ratios* is the ratio form of Appendix A.6. It ranks at 0.936, and its level shift is the instability shown there. *Quantile-binned regression* replaces the Gaussian log score of numeric targets with a classification over bins. It ranks at 0.962. Binning keeps only part of the information: at $K = 16$ the real arm keeps 0.36 of the 0.93 nats on the five-column copula ladder (Appendix C.3). The numeric factors then carry less weight, and the ranking changes where those factors decide it.

Failures. Every structure, budget, and aggregation arm completed all 216 runs. The linear and CatBoost arms lost 24 and 25 runs to synthetic targets with a single class. That is a property of the *generator* (TVAE, TABDDPM, TABSYN, and STASY on some seeds). The XGBoost arm now completes all 216 runs; the encoding error in our integration that previously dropped runs on a category that exists only in the synthetic table has been fixed. Each arm is compared on the cells it completed.

D.7 COMPARISON WITH SAMPLE-LEVEL FIDELITY METRICS

α-precision and β-recall correlate only moderately with FCU (pooled Spearman 0.70 and 0.64), and the disagreement is systematic. 27 **of the** 140 **cells (**19%**) have α-precision above** 0.8 **and FCU below** 0.5**.** The sharpest case is CDTD on MIMIC_III: α-precision 0.97, β-recall 0.93, MF $= 0.94$, and FCU $= 0.00$. The synthetic points lie in dense regions of the real distribution, and yet a model cannot learn the table's conditionals from them. Authenticity correlates with FCU at -0.53. This is expected. A generator maximizes authenticity by producing rows unlike any training row. The collapsed cells do exactly that, at authenticity 1.0 and FCU $= 0$.

Generator failures. CODI scores FCU $= 0.000$ on every table it completed (CRITEO_1M absent), at MF $= 0.65$ and discrimination AUC 1.00. STASY scores 0 on five of its eleven completed tables (0.11 on average, MF $= 0.59$, down to 0.15 on MIMIC_III). TABDDPM averages 0.19 at MF $= 0.60$ and α-precision

Table 17: Copying diagnostics on the 7 tables of the memorization runs. Every quantity is averaged over the runs of a table first. *Near-copy*: the share of synthetic rows closer to a training row than the calibrated cutoff, in percent. Held-out real rows score about 5. *Above null*: tables on which every run exceeds the rate measured on held-out real rows. *MemIndex*: median over tables. *Excess*: MemIndex minus the value of fresh real rows on the same table, as the median over tables with the minimum in parentheses. *Rand.−novel*: FCU on a random subset minus FCU on the rows that are not near-copies, at equal size, with the largest absolute value over tables in parentheses. DCR: 1 is no closer than real rows. The last row scores real rows that no generator saw as if they were synthetic.

Entry	FCU	Near-copy	Above null	MemIndex	Excess	Rand.−novel	DCR
TVAE	0.360	8.1	5/7	+0.037	+0.044 (−0.007)	−0.002 (0.016)	0.77
CDTD	0.902	4.3	1/7	+0.038	+0.022 (−0.008)	−0.001 (0.008)	1.00
ForestDiffusion	0.801	4.0	1/7	+0.017	+0.020 (−0.017)	+0.005 (0.009)	1.00
TabDiff	0.787	2.9	0/7	+0.024	+0.017 (−0.010)	−0.002 (0.013)	1.00
TabSyn	0.572	1.9	0/7	+0.012	+0.003 (−0.040)	+0.003 (0.008)	1.00
CopulaGAN	0.481	0.8	0/7	−0.005	+0.012 (−0.030)	+0.000 (0.008)	1.00
CTGAN	0.409	0.8	0/7	+0.002	+0.011 (−0.058)	+0.000 (0.007)	1.00
TabDDPM	0.199	0.7	0/7	+0.002	−0.006 (−0.071)	+0.000 (0.004)	1.00
STaSy	0.089	0.2	0/7	+0.000	+0.025 (−0.052)	+0.002 (0.009)	1.00
GaussianCopula	0.211	0.1	0/7	+0.001	−0.002 (−0.050)	+0.000 (0.004)	1.00
CoDi	0.000	0.0	0/7	−0.002	+0.013 (−0.054)	+0.000 (0.000)	1.00
SMOTENC (control)	0.481	12.7	6/7	+0.050	+0.083 (+0.032)	−0.001 (0.017)	0.24
Fresh real rows	0.997	5.2	–	+0.015	–	+0.001 (0.007)	1.00

0.27 across twelve tables, yet reaches 0.94 on CDC_DIABETES and 0.90 on BANK_MARKETING. CDTD leads five tables among the learned generators and scores 0 on three (CDC_DIABETES, MIMIC_III, DIABETES_130US), where its discrimination AUC ranges 0.90–1.00. We treat these as failures of the emitted tables, not artifacts of FCU, since the independent columns agree in every case; they do not by themselves distinguish the published generator from our integration (the XGBoost encoding fault of Appendix D.6 shows how an integration error can look like a collapse). We audited CODI directly (`scripts/synthetic_marginal_audit.py`): its numeric columns collapse toward a uniform spread over the real range (SD 0.65–1.14 times a uniform's, against 0.08–0.73 for TABSYN) with means shifted by 1.5–47 real SDs, and no column is missing or has a novel level, so the output decoded correctly and the collapse is a configuration failure of our run (`epochs`=100 against the reference default 20,000), not a claim about the published generator; the row stays in the grid as a record of what was run.

Robustness to dataset effects. The correlations above pool cells across twelve datasets. Pooling could reflect differences between datasets instead of the association of interest. We therefore recomputed each correlation as the mean of the within-dataset correlations. Every estimate moves *away* from zero: from 0.752 to 0.823 for 1− discrimination AUC, from 0.697 to 0.730 for α-precision, from 0.639 to 0.650 for β-recall, and from −0.534 to −0.577 for authenticity. A sample-weighted meta-analytic average and a correlation on within-dataset ranks agree with these values to within 0.005. The pooled values quoted in Section 6 are therefore conservative.

D.8 COPYING DIAGNOSTICS

FCU cannot tell a relationship that was learned from one that was copied. A verbatim copy of the training rows scores FCU = 1 (Table 3). We therefore report the near-copy share beside FCU (Appendix B.4). It never enters FCU or the leaderboard and is not a privacy guarantee. We run it on the seven smallest tables (Table 17). Every cell has nine runs, and all eleven generators and the control are included. These runs use the same corrected sample-size matching as Table 25, and their FCU values agree with it exactly on the matched cells.

The diagnostic against a known answer. We build synthetic tables with a known share of copied rows. We permute every column of the training split independently, which keeps the marginals and removes all dependence, so this base table scores FCU ≈ 0. We then replace a fraction c of its rows with verbatim training rows, on the seven tables, five seeds, $c \in \{0, 0.25, 0.5, 0.75, 1\}$. The near-copy share recovers c to within 0.008 on all 175 rungs, and FCU rises from 0 to 0.33, 0.62, 0.84, and 1: the copies carry the whole score. The calibration check, the rate at which held-out real rows are themselves flagged, reads 0.050 to 0.056 on six tables and 0.074 to 0.085 on PETFINDER_TAB, above the 0.05 target; that table is mostly categorical, with many tied distances. The ladder contains exact copies only. It does not test a generator that places rows just outside the cutoff.

Control vs. held-out rows. Table 17 gives the results. The oversampling control has a near-copy share of 12.7% on average (4.2%–25.0% across tables), against 5.2% for real rows no generator saw; it exceeds the held-out rate in every run on six of the seven tables and has the highest or second-highest share of all entries everywhere. TVAE follows at 8.1%; every other generator is below 5% on average. DCR agrees (0.24 for the control, 0.77 for TVAE, 1.00 for all others on these seven tables; 0.22 for the control over the full grid, Table 24).

Limits. A distance criterion can miss leakage that a membership-inference attack would find (Yao et al., 2025): an interpolated row is not a near-copy under the cutoff. The bootstrap intervals of Appendix B.3 resample test rows only; for a difference between two separately fitted arms they are too narrow, so this section uses the spread over runs. The near-copy share uses the first ordering of the ensemble, while the FCU column is the mean over orderings.

D.9 Sources of variance

Every cell has three sampling seeds, drawn from one trained model, crossed with three split repeats. For each dataset, averaged over generators, Table 18 reports two standard deviations of FCU: across sampling seeds with the repeat fixed, and across repeats with the seed fixed (`scripts/analyze_results.py variance`).

Table 18: Spread of FCU between sampling seeds and between split repeats, averaged over generators, all twelve tables. The ratio is the between-seed SD divided by the between-repeat SD. Above 1, the sampling seed matters more than the split repeat.

Dataset	SD, between seeds	SD, between repeats	Ratio (seed / repeat)
ADULT_CENSUS	0.050	0.006	8.11
BANK_MARKETING	0.054	0.007	8.10
BEIJING_PM25	0.024	0.028	0.86
CDC_DIABETES	0.012	0.004	3.19
CREDIT_DEFAULT_TAIWAN	0.054	0.020	2.63
CRITEO_1M	0.036	0.001	31.93
DIABETES_130US	0.027	0.013	2.07
HOME_CREDIT	0.055	0.004	12.44
MAGIC	0.113	0.006	18.55
MIMIC_III	0.021	0.023	0.88
PETFINDER_TAB	0.016	0.009	1.85
SUPPORT2	0.084	0.010	8.05

The ratio ranges from 0.86 to 31.93. On most tables the sampling seed matters far more than the split repeat. On BEIJING_PM25 and MIMIC_III the two are comparable, so neither source can be dropped. A campaign with one seed cannot report this layer. A crossed random-effects model, sampling seed by split repeat fitted by the method of moments, gives the standard error of a cell mean directly. On every table it is 1.85 to 2.0 times SD/$\sqrt{9}$, because a seed effect is shared by three runs, so nine runs carry about three seeds' worth of information (`scripts/analyze_results.py variance`). The descriptive spread in Table 18 should be read with that factor, and every ranking claim in Section 6.2 against this layer.

D.10 How many factors carry the score

FCU weights each factor by the information that the real arm captured. If one or two factors dominate, "the share of the captured dependence information" describes those few factors and not the whole table. We therefore report the effective number of factors,

$$N_{\text{eff}} \;=\; \frac{\left(\sum_j \widehat{I}_j\right)^2}{\sum_j \widehat{I}_j^{\,2}},$$

where $\widehat{I}_j$ is the contribution of factor j to $\widehat{I}$, clipped at 0 before the formula. A factor whose real arm scored below climatology has a negative $\widehat{I}_j$ (Appendix A.5); with signed weights the formula could fall below 1 and the top-three share could exceed 1, so such factors get no weight here, although they still enter $\widehat{I}$ itself with their sign. N_{eff} equals the number of factors when all weights are equal, and 1 when a single factor carries all the weight. Table 19 reports it per dataset.

No denominator is close to zero, so the undefined case of Section 3.2.2 does not arise on this corpus. Concentration varies by a factor of five. On HOME_CREDIT, MIMIC_III, and BEIJING_PM25, three factors carry three quarters or more of the weight, so FCU there is mostly a statement about those three. On ADULT_CENSUS, the single factor `education-num` given `education`, a deterministic recoding, carries 2.0 of 4.3 nats. This

Table 19: Concentration of FCU's denominator, per dataset, all twelve tables. $\widehat{I}_j$ depends on the real arm only, so it is shared by every generator on a dataset. Computed from the per-factor records of one run, so it is a diagnostic without an interval.

Dataset	Factors scored	$\widehat{I}$ (nats)	N_{eff}	Top-3 share
HOME_CREDIT	16	4.21	3.2	0.76
MIMIC_III	10	3.40	3.4	0.84
BANK_MARKETING	14	2.40	3.5	0.70
ADULT_CENSUS	14	4.28	3.7	0.71
CRITEO_1M	16	14.27	4.4	0.74
BEIJING_PM25	11	2.86	4.7	0.76
MAGIC	10	4.78	4.9	0.71
PETFINDER_TAB	17	2.58	7.0	0.55
SUPPORT2	29	7.71	8.8	0.52
DIABETES_130US	42	2.80	9.7	0.47
CDC_DIABETES	21	1.19	13.1	0.38
CREDIT_DEFAULT_TAIWAN	23	11.02	16.7	0.23

reflects what the joint distribution contains, and weighting by information is what Theorem 1 prescribes. The consequence is that generator rankings on ADULT_CENSUS reflect the preservation of this recoding more strongly than the preservation of many weaker relationships. The number of factors and their concentration are different things: DIABETES_130US scores 42 factors, but their weight is equivalent to about 10.

D.11 Full grid, paired leaderboard, and completion

Completion. Across all twelve tables, 1,260 of the $12 \times 12 \times 9 = 1{,}296$ possible runs produced a scored row (97%). No summary of the main grid is therefore conditional on success (`scripts/failure_aware_summary.py`); the ablation arms of Appendix D.6 are compared on the cells they completed. The gap is entirely CRITEO_1M: CTGAN, CopulaGAN, CoDi, and STaSy produce no usable output at that scale, and those cells are absent rather than recorded as zeros. A missing cell must stay distinct from a zero, because the grid also contains true collapses (CoDi at FCU $= 0.000$ on every table it completed).

Both tables below are long, so they are placed with the rest of Appendix H.

Full grid. Table 25 holds every cell. Tables 25 and 26 are generated from the same per-cell summary records by `scripts/analyze_results.py`.

Paired leaderboard. Table 20 ranks the generators per dataset by mean FCU over the nine runs of each cell. Each step between neighboring ranks shows the mean paired difference and the share of paired runs whose paired 95% interval excludes zero (Section 6.2). The paired rows pool every pair of generator seeds within a repeat, so a step averages up to 27 paired differences (3 repeats $\times$ 3 $\times$ 3 seed pairings), and the share is over those 27; the *runs* column counts the nine runs of the cell. The intervals are conditional on the fitted models and the split, and a sorted list evaluated at adjacent ranks is a selected comparison, so the step columns are descriptive sensitivity summaries. The *rank set* column carries the simultaneous claim: a 95% confidence set for the generator's rank on that table (Neuhof & Benjamini, 2024). Every pair of generators is compared by a paired t-test over the nine runs both completed, the p-values are corrected by Holm over all pairs, and a rank is excluded only by a rejected pair, so the sets cover every true rank at once with probability at least 0.95 under the paired test's assumptions on the run-to-run differences. Table 20 applies the same construction pooled over the corpus, paired by table, repeat, and seed: TabDiff, ForestDiffusion, CDTD, and TabSyn share the set 1 to 4 and are confidently above every other generator, and CoDi is confidently last. Per table, CDTD is confidently first on two and cannot be excluded from first on six; TabSyn is never confidently first but cannot be excluded from first on seven. The oversampling control is listed separately in each block and excluded from the rank sets.

E Limitations

Section 8 lists these points briefly. We expand each one here.

Structure is a heuristic. The ordering and the parent sets come from pairwise associations. The Chow–Liu tree is optimal among trees, not among all orderings, and a pairwise screen can miss interactions that are only informative jointly (Appendix C.4). Theorem 1 states what is lost, and the random-ordering ensemble measures the variation over sampled orderings on two tables. The structure remains a modeling choice.

Table 20: Generators ranked by mean FCU over the corpus, with a simultaneous 95% confidence set for each rank (Neuhof & Benjamini, 2024). Every pair of generators is compared by a paired t-test over the runs both completed, paired by table, repeat and sampling seed; the p-values are Holm-corrected over all pairs, and a rank is excluded only by a rejected pair. The last columns count the tables on which the generator's per-table rank set is $\{1\}$ (confidently first), on which it contains 1 (first not excluded), and on which its point rank is 1. The oversampling control is excluded.

Generator	FCU	rank	rank set	confidently first	first not excluded	point first
TabDiff	0.653	1	1–4	2/12	4/12	3/12
ForestDiffusion	0.635	2	1–4	0/12	4/12	2/12
CDTD	0.620	3	1–4	2/12	6/12	5/12
TabSyn	0.575	4	1–4	0/12	7/12	2/12
CopulaGAN	0.368	5	5–6	0/11	0/11	0/11
CTGAN	0.284	6	6–9	0/11	0/11	0/11
TVAE	0.210	7	6–9	0/12	0/12	0/12
TabDDPM	0.194	8	5–10	0/12	1/12	0/12
GaussianCopula	0.186	9	6–10	0/12	0/12	0/12
STaSy	0.114	10	8–10	0/11	0/11	0/11
CoDi	0.000	11	11	0/11	0/11	0/11

The deficit describes P, not Q. $\Delta(P)$ measures the dependence in P that the chosen factorization omits. It does not measure discrepancies that Q introduces outside the scored tasks. Let A and B be independent fair bits under P, and let Q give the two equal-bit outcomes probability $(1-\epsilon)/2$ each and the two unequal ones $\epsilon/2$ each. Both marginals match P. Under an empty parent graph $\Delta(P) = 0$ and every scored quantity agrees, while $D_{\mathrm{KL}}(P \,\|\, Q) = \frac{1}{2}\log\{1/[4\epsilon(1-\epsilon)]\}$ grows without bound as $\epsilon \to 0$. A known deficit is therefore a statement about how well the chain approximates P, not a certificate that every important difference between P and Q was inspected.

Reading the diagnostics together. $\Delta(P)$ (exact, where P is known; Appendix C.5), $\hat{G}$ (a finite-sample diagnostic of whether wider conditioning would help on this sample, calibrated against but not an estimate of $\Delta(P)$; Appendix D.4), and the parent-budget ablation (Appendix D.6) are not interchangeable. In practice: a high FCU with a small $\hat{G}$ and a budget-stable ranking supports the chain's fidelity claim; a large positive $\hat{G}$ calls for a wider chain or allMLU on that table; and a high FCU with a poor copying diagnostic (Appendix D.8) leaves the copying question open, since FCU does not measure it.

Not a privacy measure. A generator that copies training rows scores $\mathrm{FCU} = 1$, at a discrimination AUC below 0.5 (Section 5.1). Appendix D.8 reports three copying diagnostics beside FCU. Any privacy claim needs membership-inference checks in addition.

Bounded range. FCU is a clipped ratio of two sums. Once the synthetic excess exceeds the information the chain captured, FCU is 0 and no longer separates generators. The mean-shift ladder reaches this bound at its last rung (Appendix C.3). The two sums in nats, reported beside every score, still separate such generators.

Numeric-factor approximation. A numeric factor is scored by a Gaussian log score at a scale profiled on the test rows, not by a fitted predictive density. Its excess targets a divergence of conditional means, is blind to a change in conditional spread alone, and responds logarithmically to a large mean shift (Appendix A.5). Theorem 1 is therefore exact for categorical factors and an approximation for numeric ones. Scoring quantile-binned numeric targets as categorical factors removes the approximation and registers the spread, but at $n = 4{,}000$ it keeps under half the information the Gaussian score keeps and ranks the rungs worse (Appendix C.3).

Configuration dependence. FCU depends on the estimated structure, the ordering, the parent rule, the class cap, the model family, the calibration, and the sample size. Two FCU values are comparable only under the same configuration. Every result row therefore records its configuration.

Dependence on the model class. Per-factor scores are relative to a fixed model class, gradient-boosted trees by default. Treating both arms alike cancels part of the shared estimation error. It does not remove model approximation error, and the metric cannot see a conditional that the model class cannot represent.

Missing data. Every table in our corpus is complete, because the preparation removed rows with missing values. This is a restriction of the study, not a property of the data. The per-factor arms fit and score directly on the supplied columns, and the preprocessor does not impute. Complete-case preparation is also not neutral. It can favor the simpler records of a table, where conditional structure is easiest to reproduce, so a generator may receive more credit than it would on the raw table. Extending FCU to incomplete data is future work. The options are to treat missingness as a level, to score each factor on its own complete cases, or to evaluate the missingness mechanism as a factor in its own right.

Coverage of uncertainty. Split repeats capture split noise. The three sampling seeds capture the sampling noise of one trained generator (Appendix D.9). The paired bootstrap captures test-set noise, including the correlation between factors that share parents and test rows. We vary neither the training seed of the generators nor the model seed of the evaluator. Crossing in either would cost several times the present grid.

Work we leave open.

- *Named downstream tasks.* Section 6.2 holds out half of the columns. It does not define the tasks a practitioner would run on a given table and test whether FCU predicts those. That study needs task definitions for each domain.
- *A privacy attack suite.* DCR and the copying diagnostics of Appendix D.8 detect copying. They are not attacks. Membership inference, attribute inference, and differentially private baselines would be needed to study the trade-off between privacy and utility, which this paper does not claim to measure.
- *Temporal and grouped splits.* Our splits are random, and Appendix F.4 checks that the three splits are exchangeable. Random assignment does not make rows independent: BEIJING_PM25 holds hourly observations and DIABETES_130US repeated patient encounters, so their paired intervals are optimistic, and tables with a time order or repeated subjects need grouped splits in a separate campaign.
- *Very wide vocabularies.* Our widest categorical column has $4{,}600$ levels, and the class cap applies to it (Appendix B.2). Columns with tens of thousands of levels, as in recommendation data, would test whether the cap remains a mild approximation or becomes the dominant modeling choice.

F DATASETS, GENERATORS, AND COMPUTE

F.1 DATASETS

Table 21 gives the measured shape of each table, its source, and whether it may be redistributed.

Table 21: Datasets, measured on the training split. #cat counts string and boolean columns, plus integer columns with at most 20 distinct values. Max card. is the largest number of levels in any column. *Rel.*: whether the released artifact contains the prepared file (*under Criteo's terms). All twelve tables are complete; we verified that none contains a null.

Dataset	n	d	#cat	max card.	Target	Task	Source	Rel.
CRITEO_1M	671,398	17	12	4,600	`target`	binary	Criteo (display ads)	yes*
MIMIC_III	389,631	11	9	571	`DISCHARGE_LOCATION`	multiclass	PhysioNet	**no**
CDC_DIABETES	177,576	22	19	13	`Diabetes_binary`	binary	UCI / CDC BRFSS 2015	yes
HOME_CREDIT	147,140	17	13	57	`TARGET`	binary	Kaggle (Home Credit)	**no**
DIABETES_130US	62,847	43	38	632	`readmitted`	multiclass	UCI (Diabetes 130-US)	yes
ADULT_CENSUS	33,334	15	10	42	`high_income`	binary	UCI (Adult)	yes
BANK_MARKETING	30,235	15	8	12	`y`	binary	UCI (Bank Marketing)	yes
BEIJING_PM25	29,229	12	3	12	`pm2.5`	regression	UCI (Beijing PM2.5)	yes
CREDIT_DEFAULT_TAIWAN	21,000	24	10	11	`default`	binary	UCI (Credit Card Default)	yes
MAGIC	13,314	11	1	2	`class`	binary	UCI (MAGIC Telescope)	yes
PETFINDER_TAB	10,483	18	15	164	`adopted`	binary	Kaggle (PetFinder.my)	**no**
SUPPORT2	5,471	30	11	10	`death`	binary	UCI (SUPPORT2)	yes

Targets. Ten targets are native columns of their source. Two are derived: `high_income` in ADULT_CENSUS is the indicator of an income above \$50k, and `adopted` in PETFINDER_TAB is derived from `AdoptionSpeed`. CRITEO_1M is a subsample of one million rows. HOME_CREDIT uses the application table only, and PETFINDER_TAB uses the tabular columns only.

Access and redistribution. Nine of the twelve tables are public benchmarks that may be redistributed, and we release their prepared files. Three are not included in the released artifact. MIMIC_III requires credentialed PhysioNet access, with a signed data-use agreement and human-subjects training. HOME_CREDIT and PETFINDER_TAB are Kaggle competition data whose terms allow use but not redistribution. For these three we release the preparation scripts and the exact column and split specifications, so a reader who obtains the data through the proper channel can reproduce our tables.

Preparation. Every table went through the same pipeline before any experiment. We select the modeling columns, derive the target where it is not native, drop every row with a missing value, and split $70/30$ into train and test with a fixed seed. No column was dropped for missingness and no value was imputed. This is why all twelve tables are free of nulls. It is a limitation and not a property of the data (Appendix E). SUPPORT2 is the clearest case: $5{,}471$ rows remain of about $9{,}100$.

F.2 GENERATORS

The synthetic tables were produced by a separate orchestration codebase, released with this work. It wraps the reference implementation of each generator behind one interface. Every generator was trained once per dataset on the training split, under one fixed training seed. The trained model was then sampled under three sampling seeds (420, 421, 422), which are the three seeds of Section 4. There was *no hyperparameter search*. Each generator ran at the settings recommended by its authors or at its library defaults (Table 22). Each sample has as many rows as the real training split, and numeric outputs are rounded to four decimals.

Table 22: Generator implementations and settings. Every generator is trained once and sampled under seeds 420 to 422. No hyperparameter optimization was performed. The SDV models are GAUSSIANCOPULA, TVAE, CTGAN, and COPULAGAN.

Generator	Implementation	Settings
GaussianCopula	`sdv` ≥ 1.36	defaults; min/max enforcement on
TVAE	`sdv` ≥ 1.36	`epochs`=300, `batch`=500, `embedding`=128, compress/decompress $(256, 256, 256)$, `l2scale`=10^{-3}
CTGAN	`sdv` ≥ 1.36	`epochs`=300, `batch`=500, generator/discriminator $(256, 256, 256)$, both lr $= 2\times10^{-4}$
CopulaGAN	`sdv` ≥ 1.36	`epochs`=300, `batch`=500, `embedding`=128, both lr $= 2\times10^{-4}$
STaSy	reference impl., `torchdiffeq` 0.2.5	VE-SDE, `epochs`=500, `batch`=1000, lr $= 2\times10^{-3}$, hidden $(256, 512, 1024, 1024, 512, 256)$, ODE sampler, EMA 0.9999
TabDDPM	reference impl.	`steps`=5000, `timesteps`=1000, `batch`=4096, lr $= 2\times10^{-3}$, $d_{\text{layers}} = (256, 512, 512, 256)$, quantile normalization, MSE Gaussian loss
CoDi	reference impl.	`epochs`=100 (both branches; the reference default is 20,000), $T = 1000$, `batch`=4096, lr $= 2\times10^{-3}$, $\lambda_{\text{con}} = \lambda_{\text{dis}} = 0.2$
TabSyn	reference impl.	VAE `epochs`=150, diffusion `epochs`=150, $\beta \in [10^{-5}, 10^{-2}]$, $\lambda = 0.7$
GReaT	`transformers` 5.5.4, `peft` 0.9.0	`distilgpt2`, `epochs`=15, `batch`=16, sampling $k = 100$
ForestDiffusion	`ForestDiffusion` $\geq 1.0.6$	flow matching, $n_t = 20$, 100 trees, `max_depth`=7, `duplicate_K`=20 (10 on CRITEO_1M), one-hot only below 64 levels
CDTD	reference impl. (Mueller et al., 2025)	`cat_emb_dim`=16, `mlp_layers`=5, `mlp_units`=1024, `steps`=10000, `warmup`=500, `batch`=4096, lr $= 10^{-3}$, 100 diffusion steps at sampling
TabDiff	reference impl. (Shi et al., 2024)	`layers`=2, `d_token`=4, `dim_t`=1024, `steps`=5000, `batch`=4096, lr $= 10^{-3}$
SMOTENC (control)	`imbalanced-learn` ≥ 0.11	$k = 5$ neighbors, min–max scaling, $1.0\times$ sampling fraction

GREAT is configured above but permanently excluded from the benchmark grid and every result in this paper. Its retry-and-parse loop hard-caps at 16 attempts and returns whatever partial sample it has collected when that cap is hit, regardless of the requested row count: every run request the correct number of rows, but every run returns only about 950 to 1,600 rows, from tables as small as CDC_DIABETES (38,052 requested) to as large as CRITEO_1M (143,871 requested). This is a fixed retry budget in the reference implementation, not a bug in our orchestration, and it does not scale with dataset size. We keep its row here as a record of what was run.

What can be reproduced. We release the evaluation code, the configurations, and the per-cell result records in full. Every number in this paper can be recomputed from the synthetic tables. Regenerating the synthetic tables also requires the orchestration codebase and the dependencies of each generator, whose versions are listed above as minimum versions or reference implementations rather than exact pins; the orchestration environment ran PyTorch 2.11.0 and, for STASY, `torchdiffeq` 0.2.5, and the remaining generator libraries are bounded below only. The contribution of this paper is the metric, and reproducing it does not require rerunning eleven third-party generators.

F.3 EVALUATION PROTOCOL

The eleven generators are TVAE and CTGAN (Xu et al., 2019), GAUSSIANCOPULA and COPULAGAN (Patki et al., 2016), CODI (Lee et al., 2023), STASY (Kim et al., 2022), TABDDPM (Kotelnikov et al., 2023),

FORESTDIFFUSION (Jolicoeur-Martineau et al., 2024), TABSYN (Zhang et al., 2023), CDTD (Mueller et al., 2025), and TABDIFF (Shi et al., 2024), with SMOTENC (Chawla et al., 2002), which interpolates between nearest neighbors of the training rows, as the memorization and oversampling control. Each is trained once per dataset on the same real training split with its recommended hyperparameters (Table 22); the training seed is not varied, and the trained model is sampled under three sampling seeds.

Every score is computed on the held-out split of each table, which no generator was trained on. The evaluator runs three *split repeats*: each draws a seeded 70% subsample of the real training split for structure estimation and the real arm, and subsamples the synthetic training set to the same size. The sampling seeds are crossed with the split repeats, so a *cell*, one dataset and one entry, carries $3 \times 3 = 9$ runs. Both arms of every factor use LightGBM (Ke et al., 2017) with 100 trees, native categorical handling, the calibration of Section 3.2.2, and one evaluator seed that is never varied. We report the mean over the nine runs with $\mathrm{SD}/\sqrt{9}$ as a descriptive spread; the runs share the test rows and, within a repeat, the real arm, so a crossed variance model (sampling seed $\times$ split repeat) gives standard errors roughly twice as wide (Appendix D.9). Per-run paired-bootstrap 95% intervals use 200 resamples of the shared test rows, conditional on the fitted models (Appendix D.11). Categorical factors are capped at $C_{\max} = 500$ classes, which binds on the three widest tables (Appendix B.2). The grid is 12 datasets $\times$ 12 entries $\times$ 9 runs, $1{,}296$ possible runs; completion is reported in Appendix D.11.

F.4 SPLIT VALIDITY AND COMPUTE

Split validity. Before the campaign we checked that the train, validation, and test splits of every dataset are exchangeable. All twelve pass. The check matters. An earlier version of this corpus contained splits that were sorted by the label and not shuffled. Such a split raises no error anywhere in the pipeline. It lowers the oracle ceiling, and the table simply looks as if it carried weak conditional signal. Both arms of every factor train on training rows and are scored on held-out real rows, so a non-exchangeable split silently invalidates every number. The check therefore gates the campaign.

Compute. All runs used CPU nodes with two AMD EPYC 7662 processors (128 cores) and $1{,}007$ GiB of memory. No GPU was used for evaluation. The figures below are the measured cost of the previous campaign of the same grid (one generator seed, five split repeats, eleven entries).

We ran the grid as SLURM array jobs, with one task per dataset and split repeat. The structure and the real arm are computed once per split and reused for all entries. Allocations were 8 cores and 32 GB for the seven smallest datasets, 16 cores and 64 GB for the three mid-sized ones, and 16 cores and 128 GB for MIMIC_III and CRITEO_1M. The wall-clock time per task ranges from about 80 s on SUPPORT2 to 15.6 h on CRITEO_1M. The 60 tasks total $1{,}632$ allocated CPU-hours. The cost is very uneven. The two largest tables account for 95% of the total and CRITEO_1M alone for 76%, while all thirty-five tasks of the smallest tier need 11.9 CPU-hours together. The $4{,}600$-level column of CRITEO_1M dominates the cost, which is the reason for the class cap (Appendix B.2). The validation suite runs separately. Its most expensive part, the five memorization tasks on CRITEO_1M, took $3{,}557$ allocated CPU-hours.

G EXTENDED RELATED WORK

This appendix adds detail to Section 7.

Evaluation frameworks. There is broad agreement that the goal is to approximate the joint distribution over all columns, subject to privacy constraints (Jordon et al., 2022). Frameworks split that goal into metrics that target different parts of the joint, from marginal and pairwise statistics to the distinguishability of whole rows (Patki et al., 2016; Qian et al., 2023; Dankar & Ibrahim, 2021). Sample-level methods measure fidelity, diversity, and generalization directly (Alaa et al., 2022). Discriminative two-sample tests train a classifier to tell real from synthetic rows. This work shows that matching low-order marginals is not enough. It still treats machine-learning utility as a separate axis and rarely connects it to a factorization of the joint. Its leaderboards are point orders, sometimes with a per-cell interval, and do not say which ranks a generator can hold at once. Neuhof & Benjamini (2024) build simultaneous confidence sets for the ranks of feature importances from pairwise tests with a multiplicity correction; Appendix D.11 applies the same construction to the generators, with the paired runs of a cell as the replicates.

Utility metrics. TSTR and TRTS (Esteban et al., 2017) compare models trained on synthetic data with models trained on real data for one chosen task. Many pipelines use them as the main utility indicator (Dankar & Ibrahim, 2021; Qian et al., 2023). Several studies score every column in turn from all the others: the all-models test of El Emam et al. (2022), the application-fidelity dimension of Dankar et al. (2022), and the global utility of Jiang et al. (2026). They average per-column accuracies or ratios and do not connect the resulting conditionals to the joint. Yang et al. (2024) do make that connection: they posit one objective, $Q = P$, and arrange metrics by the substructure of the joint each one tests, with machine-learning efficacy reinterpreted as a test of the

leave-one-out conditional $P(X_j \mid X_{-j})$. Their conditionals are leave-one-out, which do not multiply to a distribution, so no chain-rule identity such as Theorem 1 is available for their sum, and their evaluation has no pruning and no account of omitted dependence. Appendix D.5 compares allMLU with FCU empirically. In the same line, Senane et al. (2024) benchmark high-order causal structure on synthetic structural causal models.

H SUPPLEMENTARY TABLES

The four tables below are long enough that placing them inline would break up the discussion above, so they are collected here instead. Each is still cited by its label at the point in the text that discusses it.

Table 23: Error of the summed excess by rung. Each line is one perturbation level t of one family, averaged over seeds and, for the wide families, over the two parent budgets. KL is the exact (or Monte Carlo) full divergence at that rung and $\widehat{D}$ the mean summed excess. The two error columns are $\widehat{D}$ minus the full KL and $\widehat{D}$ minus its estimand $D_{\mathrm{KL}}(P \,\|\, Q^{\mathrm{Pa}}) - \Delta(P)$; a dash marks a family without a per-cell estimand. On the shift ladder the error grows with the shift because the numeric score responds as $\frac{1}{2}\log(1 + 2\,D_{\mathrm{KL}})$ (Appendix A.5).

Family	t	KL	$\widehat{D}$	vs. full KL	vs. estimand	cells
Gaussian copula, $d = 5$	0.00	0.000	0.001	+0.001	+0.001	5
	0.10	0.026	0.044	+0.018	+0.020	5
	0.20	0.083	0.099	+0.016	+0.017	5
	0.30	0.155	0.177	+0.022	+0.019	5
	0.40	0.235	0.291	+0.056	+0.045	5
	0.50	0.323	0.405	+0.082	+0.061	5
	0.60	0.417	0.550	+0.133	+0.102	5
	0.70	0.522	0.677	+0.155	+0.116	5
	0.80	0.641	0.803	+0.162	+0.120	5
	0.90	0.781	0.859	+0.078	+0.046	5
	1.00	0.956	0.875	-0.082	-0.082	5
Gaussian copula, mean shift	0.00	0.000	0.001	+0.001	+0.001	5
	0.25	0.199	0.152	-0.046	-0.046	5
	0.50	0.795	0.454	-0.341	-0.341	5
	0.75	1.788	0.738	-1.051	-1.051	5
	1.00	3.179	0.972	-2.208	-2.208	5
Discrete network, $d = 4$	0.00	0.000	-0.001	-0.001	-0.001	5
	0.10	0.036	0.041	+0.005	+0.005	5
	0.20	0.089	0.091	+0.002	+0.002	5
	0.30	0.154	0.156	+0.002	+0.002	5
	0.40	0.230	0.233	+0.003	+0.003	5
	0.50	0.316	0.316	-0.000	+0.001	5
	0.60	0.413	0.417	+0.003	+0.005	5
	0.70	0.521	0.517	-0.005	-0.002	5
	0.80	0.641	0.647	+0.005	+0.010	5
	0.90	0.776	0.779	+0.003	+0.010	5
	1.00	0.926	0.887	-0.040	-0.031	5
Gaussian, sparse DAG, $d = 20$	0.00	0.000	0.012	+0.012	+0.012	10
	0.10	0.016	0.058	+0.041	+0.046	10
	0.20	0.059	0.128	+0.068	+0.084	10
	0.30	0.124	0.198	+0.074	+0.104	10
	0.40	0.207	0.287	+0.080	+0.126	10
	0.50	0.308	0.379	+0.071	+0.136	10
	0.60	0.427	0.459	+0.032	+0.119	10
	0.70	0.565	0.520	-0.045	+0.069	10
	0.80	0.727	0.565	-0.163	-0.012	10
	0.90	0.919	0.588	-0.331	-0.130	10
	1.00	1.149	0.605	-0.544	-0.272	10
Gaussian, sparse DAG, $d = 40$	0.00	0.000	0.022	+0.022	+0.022	10
	0.10	0.039	0.138	+0.099	+0.110	10
	0.20	0.143	0.301	+0.159	+0.197	10
	0.30	0.296	0.464	+0.168	+0.239	10
	0.40	0.493	0.694	+0.201	+0.310	10
	0.50	0.731	0.901	+0.170	+0.321	10
	0.60	1.013	1.078	+0.065	+0.268	10
	0.70	1.347	1.259	-0.088	+0.183	10
	0.80	1.748	1.365	-0.382	-0.011	10
	0.90	2.243	1.412	-0.831	-0.301	10
	1.00	2.885	1.449	-1.437	-0.640	10
Discrete network, sparse DAG, $d = 20$	0.00	0.000	0.010	+0.010	+0.010	10
	0.10	0.365	0.394	+0.029	+0.024	10
	0.20	0.870	0.950	+0.080	+0.037	10
	0.30	1.462	1.601	+0.140	+0.066	10
	0.40	2.132	2.272	+0.140	+0.055	10
	0.50	2.883	2.985	+0.102	+0.037	10
	0.60	3.722	3.805	+0.083	+0.079	10

Table 24: The benchmark condensed: mean FCU over the 12 tables, the number of tables an entry leads (the control counted against every entry, the generators against each other), mean marginal fidelity, discrimination AUC (0.5 is indistinguishable) and DCR score (1 is no closer than real rows). Per-dataset ranks with paired intervals: Table 26.

Entry	FCU	leads	MF	Disc. AUC	DCR
TabDiff	0.653	3/12	0.938	0.88	0.89
ForestDiffusion	0.635	2/12	0.916	0.92	0.88
CDTD	0.620	5/12	0.914	0.79	0.81
TabSyn	0.575	2/12	0.919	0.82	0.90
CopulaGAN	0.368	0/12	0.882	0.96	0.99
CTGAN	0.284	0/12	0.868	0.96	0.98
TVAE	0.210	0/12	0.865	0.97	0.65
TabDDPM	0.194	0/12	0.596	0.93	0.92
GaussianCopula	0.186	0/12	0.917	0.98	1.00
STaSy	0.114	0/12	0.594	0.97	1.00
CoDi	0.000	0/12	0.655	1.00	1.00
SMOTENC (control)	0.549	4/12	0.933	0.86	0.22

Table 23, continued.

Family	t	KL	$\widehat{D}$	vs. full KL	vs. estimand	cells
	0.70	4.661	4.596	-0.065	+0.047	10
	0.80	5.717	5.471	-0.247	+0.049	10
	0.90	6.921	6.367	-0.553	+0.016	10
	1.00	8.314	7.048	-1.266	-0.309	10
Mixed-type mixture, $d = 6$	0.00	0.000	-0.001	-0.001	–	5
	0.10	0.003	0.012	+0.009	–	5
	0.20	0.013	0.033	+0.021	–	5
	0.30	0.026	0.045	+0.019	–	5
	0.40	0.043	0.076	+0.033	–	5
	0.50	0.063	0.106	+0.043	–	5
	0.60	0.086	0.137	+0.051	–	5
	0.70	0.112	0.164	+0.053	–	5
	0.80	0.140	0.204	+0.064	–	5
	0.90	0.172	0.218	+0.046	–	5
	1.00	0.208	0.248	+0.040	–	5

Table 25: Full evaluation grid: 12 datasets × 12 entries (generators plus the oversampling control), 9 runs per cell (3 split repeats × 3 generator seeds) (FCU mean ± SEM; remaining columns are means). Cells whose synthetic file was unavailable are omitted rather than recorded as zeros. A dash marks a column that is undefined in at least one run of the cell. DCR is a bounded nearest-neighbor disclosure diagnostic, not a privacy guarantee: 1.0 means synthetic rows sit no closer to the training data than held-out real rows do, and values below 1 mean they sit closer, which is consistent with memorization under this metric.

Dataset	Generator	FCU	MF	TSTR AUC	Disc. AUC	DCR
adult_census	CDTD	0.976 ± 0.001	0.983	0.90	0.57	1.00
adult_census	TabSyn	0.850 ± 0.002	0.956	0.89	0.72	1.00
adult_census	CopulaGAN	0.820 ± 0.007	0.918	0.88	0.88	1.00
adult_census	SMOTENC	0.741 ± 0.007	0.952	0.90	0.83	0.27
adult_census	CTGAN	0.703 ± 0.012	0.892	0.88	0.94	1.00
adult_census	ForestDiffusion	0.668 ± 0.002	0.900	0.88	0.97	1.00
adult_census	TabDiff	0.595 ± 0.036	0.914	0.89	1.00	1.00
adult_census	TVAE	0.300 ± 0.072	0.871	0.87	0.97	0.72
adult_census	STaSy	0.074 ± 0.037	0.606	0.75	0.99	1.00
adult_census	GaussianCopula	0.000 ± 0.000	0.904	0.79	1.00	1.00
adult_census	CoDi	0.000 ± 0.000	0.654	0.65	1.00	1.00
adult_census	TabDDPM	0.000 ± 0.000	0.568	0.82	1.00	1.00
bank_marketing	CDTD	0.948 ± 0.003	0.982	0.89	0.62	1.00
bank_marketing	TabDiff	0.911 ± 0.006	0.968	0.90	0.74	1.00
bank_marketing	TabDDPM	0.896 ± 0.006	0.984	0.88	0.68	1.00
bank_marketing	ForestDiffusion	0.854 ± 0.003	0.934	0.87	0.85	1.00
bank_marketing	SMOTENC	0.741 ± 0.005	0.949	0.90	0.75	0.28

Continued on the next page.

Table 25, continued.

Dataset	Generator	FCU	MF	TSTR AUC	Disc. AUC	DCR
bank_marketing	TabSyn	0.593 ± 0.148	0.906	0.75	0.79	1.00
bank_marketing	CTGAN	0.396 ± 0.016	0.886	0.73	0.94	1.00
bank_marketing	CopulaGAN	0.074 ± 0.006	0.859	0.83	1.00	1.00
bank_marketing	TVAE	0.001 ± 0.001	0.878	0.81	0.96	0.78
bank_marketing	GaussianCopula	0.000 ± 0.000	0.855	0.68	1.00	1.00
bank_marketing	CoDi	0.000 ± 0.000	0.633	0.50	1.00	1.00
bank_marketing	STaSy	0.000 ± 0.000	0.766	0.81	0.98	1.00
beijing_pm25	ForestDiffusion	0.832 ± 0.012	0.953	–	0.90	1.00
beijing_pm25	TabSyn	0.832 ± 0.011	0.966	–	0.71	1.00
beijing_pm25	TabDiff	0.777 ± 0.013	0.950	–	0.92	1.00
beijing_pm25	CDTD	0.774 ± 0.007	0.950	–	0.76	1.00
beijing_pm25	TVAE	0.699 ± 0.014	0.914	–	0.95	1.00
beijing_pm25	CopulaGAN	0.545 ± 0.019	0.907	–	0.94	1.00
beijing_pm25	CTGAN	0.421 ± 0.005	0.893	–	0.97	1.00
beijing_pm25	SMOTENC	0.420 ± 0.005	0.948	–	0.90	0.54
beijing_pm25	GaussianCopula	0.154 ± 0.022	0.937	–	0.98	1.00
beijing_pm25	STaSy	0.067 ± 0.034	0.789	–	0.97	1.00
beijing_pm25	CoDi	0.000 ± 0.000	0.660	–	1.00	1.00
beijing_pm25	TabDDPM	0.000 ± 0.000	0.368	–	1.00	1.00
cdc_diabetes	ForestDiffusion	0.945 ± 0.001	0.983	0.82	0.65	0.00
cdc_diabetes	TabDiff	0.943 ± 0.004	0.983	0.82	0.68	0.33
cdc_diabetes	TabDDPM	0.938 ± 0.003	0.990	0.82	0.60	0.00
cdc_diabetes	TabSyn	0.874 ± 0.010	0.978	0.82	0.65	0.67
cdc_diabetes	SMOTENC	0.797 ± 0.003	0.971	0.81	0.70	0.00
cdc_diabetes	STaSy	0.377 ± 0.012	0.918	0.81	0.81	1.00
cdc_diabetes	GaussianCopula	0.027 ± 0.003	0.972	0.74	0.86	1.00
cdc_diabetes	CopulaGAN	0.020 ± 0.010	0.905	0.75	0.92	1.00
cdc_diabetes	CTGAN	0.000 ± 0.000	0.882	0.71	0.94	1.00
cdc_diabetes	TVAE	0.000 ± 0.000	0.890	0.75	0.97	0.00
cdc_diabetes	CDTD	0.000 ± 0.000	0.950	0.79	0.90	0.00
cdc_diabetes	CoDi	0.000 ± 0.000	0.774	0.46	0.99	1.00
credit_default_taiwan	SMOTENC	0.926 ± 0.005	0.938	0.72	0.93	0.29
credit_default_taiwan	CDTD	0.878 ± 0.015	0.957	0.73	0.75	1.00
credit_default_taiwan	TabSyn	0.856 ± 0.013	0.940	0.71	0.88	1.00
credit_default_taiwan	ForestDiffusion	0.852 ± 0.005	0.885	0.71	0.98	1.00
credit_default_taiwan	CopulaGAN	0.696 ± 0.010	0.915	0.69	0.99	1.00
credit_default_taiwan	CTGAN	0.619 ± 0.012	0.891	0.70	1.00	1.00
credit_default_taiwan	TabDiff	0.577 ± 0.018	0.881	0.72	0.99	1.00
credit_default_taiwan	TVAE	0.456 ± 0.021	0.872	0.73	0.99	1.00
credit_default_taiwan	STaSy	0.382 ± 0.107	0.790	0.67	0.98	1.00
credit_default_taiwan	GaussianCopula	0.045 ± 0.006	0.788	0.62	1.00	1.00
credit_default_taiwan	CoDi	0.000 ± 0.000	0.406	0.53	1.00	1.00
credit_default_taiwan	TabDDPM	0.000 ± 0.000	0.386	0.50	1.00	1.00
criteo_1m	SMOTENC	0.806 ± 0.001	0.910	0.72	0.84	0.01
criteo_1m	TabSyn	0.456 ± 0.057	0.929	0.69	0.94	1.00
criteo_1m	CDTD	0.431 ± 0.020	0.913	0.72	0.91	0.36
criteo_1m	GaussianCopula	0.052 ± 0.000	0.935	0.62	0.98	1.00
criteo_1m	TabDiff	0.012 ± 0.006	0.887	0.56	1.00	1.00
criteo_1m	ForestDiffusion	0.000 ± 0.000	0.734	0.65	1.00	1.00
criteo_1m	TVAE	0.000 ± 0.000	0.708	0.66	1.00	1.00
criteo_1m	TabDDPM	0.000 ± 0.000	0.444	0.45	1.00	1.00
diabetes_130us	TabDiff	0.627 ± 0.023	0.979	0.59	0.77	1.00
diabetes_130us	TabSyn	0.243 ± 0.030	0.959	0.57	0.89	1.00
diabetes_130us	ForestDiffusion	0.152 ± 0.010	0.965	0.61	0.92	1.00
diabetes_130us	SMOTENC	0.135 ± 0.035	0.950	0.62	0.90	0.66
diabetes_130us	CTGAN	0.032 ± 0.009	0.940	0.56	0.92	1.00
diabetes_130us	CopulaGAN	0.000 ± 0.000	0.905	0.53	1.00	1.00
diabetes_130us	GaussianCopula	0.000 ± 0.000	0.922	0.50	1.00	1.00

Continued on the next page.

Table 25, continued.

Dataset	Generator	FCU	MF	TSTR AUC	Disc. AUC	DCR
diabetes_130us	TVAE	0.000 ± 0.000	0.885	0.58	1.00	0.86
diabetes_130us	CDTD	0.000 ± 0.000	0.434	–	1.00	1.00
diabetes_130us	CoDi	0.000 ± 0.000	0.715	0.49	1.00	1.00
diabetes_130us	STaSy	0.000 ± 0.000	0.209	–	1.00	1.00
diabetes_130us	TabDDPM	0.000 ± 0.000	0.717	0.50	1.00	1.00
home_credit	SMOTENC	0.777 ± 0.003	0.957	0.60	0.96	0.21
home_credit	TabSyn	0.764 ± 0.003	0.976	0.59	0.78	1.00
home_credit	ForestDiffusion	0.752 ± 0.002	0.958	0.61	0.99	0.83
home_credit	CDTD	0.698 ± 0.009	0.971	0.60	0.79	0.92
home_credit	GaussianCopula	0.597 ± 0.001	0.982	0.54	0.99	1.00
home_credit	CopulaGAN	0.481 ± 0.012	0.905	0.56	1.00	1.00
home_credit	STaSy	0.254 ± 0.075	0.848	0.57	0.93	1.00
home_credit	TabDiff	0.139 ± 0.070	0.882	0.51	1.00	1.00
home_credit	CTGAN	0.105 ± 0.019	0.875	0.54	1.00	1.00
home_credit	TVAE	0.000 ± 0.000	0.859	0.50	1.00	0.51
home_credit	CoDi	0.000 ± 0.000	0.753	0.52	1.00	1.00
home_credit	TabDDPM	0.000 ± 0.000	0.569	0.51	1.00	1.00
magic	CDTD	0.984 ± 0.002	0.977	0.91	0.54	1.00
magic	ForestDiffusion	0.946 ± 0.004	0.945	0.90	0.82	0.99
magic	TabDiff	0.899 ± 0.011	0.956	0.90	0.74	1.00
magic	TVAE	0.766 ± 0.006	0.915	0.85	0.87	1.00
magic	GaussianCopula	0.755 ± 0.004	0.932	0.78	0.96	1.00
magic	TabSyn	0.620 ± 0.155	0.900	0.85	0.76	1.00
magic	CopulaGAN	0.539 ± 0.011	0.899	0.81	0.96	1.00
magic	CTGAN	0.321 ± 0.013	0.882	0.77	0.98	1.00
magic	TabDDPM	0.307 ± 0.153	0.533	0.63	0.88	1.00
magic	STaSy	0.098 ± 0.034	0.652	0.78	1.00	1.00
magic	SMOTENC	0.000 ± 0.000	0.898	0.50	0.79	0.07
magic	CoDi	0.000 ± 0.000	0.635	0.52	1.00	1.00
mimic_iii	SMOTENC	0.698 ± 0.005	0.960	0.66	0.91	0.05
mimic_iii	TabDiff	0.599 ± 0.010	0.974	0.76	0.95	0.38
mimic_iii	TabSyn	0.556 ± 0.031	0.966	0.54	0.76	0.10
mimic_iii	CopulaGAN	0.179 ± 0.026	0.674	0.99	0.97	0.90
mimic_iii	ForestDiffusion	0.169 ± 0.013	0.907	0.64	1.00	0.71
mimic_iii	CTGAN	0.124 ± 0.027	0.669	0.99	0.98	0.81
mimic_iii	GaussianCopula	0.077 ± 0.004	0.945	0.54	1.00	1.00
mimic_iii	TVAE	0.000 ± 0.000	0.838	–	1.00	0.00
mimic_iii	CDTD	0.000 ± 0.000	0.941	0.53	0.97	0.43
mimic_iii	CoDi	0.000 ± 0.000	0.721	0.41	1.00	1.00
mimic_iii	STaSy	0.000 ± 0.000	0.146	0.46	1.00	1.00
mimic_iii	TabDDPM	0.000 ± 0.000	0.479	0.45	1.00	1.00
petfinder_tab	TabDiff	0.893 ± 0.012	0.962	0.71	0.77	1.00
petfinder_tab	CDTD	0.773 ± 0.016	0.955	0.71	0.75	1.00
petfinder_tab	ForestDiffusion	0.562 ± 0.006	0.920	0.71	0.95	1.00
petfinder_tab	CopulaGAN	0.552 ± 0.007	0.929	0.61	0.86	1.00
petfinder_tab	SMOTENC	0.542 ± 0.009	0.913	0.72	0.84	0.00
petfinder_tab	CTGAN	0.401 ± 0.015	0.898	0.60	0.90	1.00
petfinder_tab	GaussianCopula	0.000 ± 0.000	0.915	0.59	0.99	1.00
petfinder_tab	TVAE	0.000 ± 0.000	0.884	0.64	0.97	0.00
petfinder_tab	CoDi	0.000 ± 0.000	0.620	0.46	1.00	1.00
petfinder_tab	STaSy	0.000 ± 0.000	0.428	0.48	1.00	1.00
petfinder_tab	TabDDPM	0.000 ± 0.000	0.529	0.56	1.00	1.00
petfinder_tab	TabSyn	0.000 ± 0.000	0.726	0.49	1.00	1.00
support2	CDTD	0.979 ± 0.003	0.957	0.83	0.95	1.00
support2	ForestDiffusion	0.890 ± 0.005	0.909	0.80	1.00	1.00
support2	TabDiff	0.858 ± 0.010	0.925	0.83	1.00	0.98
support2	GaussianCopula	0.523 ± 0.005	0.917	0.71	0.99	1.00
support2	TVAE	0.302 ± 0.040	0.860	0.79	1.00	0.87

Continued on the next page.

Table 25, continued.

Dataset	Generator	FCU	MF	TSTR AUC	Disc. AUC	DCR
support2	TabSyn	0.253 ± 0.126	0.823	0.69	0.99	1.00
support2	TabDDPM	0.191 ± 0.096	0.590	0.70	0.98	1.00
support2	CopulaGAN	0.142 ± 0.013	0.883	0.80	1.00	1.00
support2	CTGAN	0.000 ± 0.000	0.834	0.65	1.00	1.00
support2	SMOTENC	0.000 ± 0.000	0.848	0.50	0.99	0.20
support2	CoDi	0.000 ± 0.000	0.630	0.75	1.00	1.00
support2	STaSy	0.000 ± 0.000	0.382	0.48	1.00	1.00

Table 26: Paired leaderboard. Generators are ranked by mean FCU over every run (mean $\pm$ SD$/\sqrt{\text{runs}}$, a descriptive spread); Δ is the mean paired difference to the next rank. *sep.* is the share of paired runs whose paired 95% interval excluded zero, over all seed pairings within a repeat (*pairs*: 3×3 per repeat, 27 per step) and over the pairings with matched sampling seeds (*matched*: one per run, 9 per step). The intervals are conditional on the fitted models and the split and are not adjusted across steps, so they are descriptive. *rank set* is a simultaneous 95% confidence set for the generator's rank on the table (Neuhof & Benjamini, 2024): paired t-tests over the nine runs of every pair of generators, Holm-corrected over all pairs, and a rank is excluded only by a rejected pair. The oversampling control is listed apart: it reproduces training rows and is the scope demonstration, not a competitor.

Dataset	Generator	FCU	Δ to next	sep. pairs	sep. matched	rank set	runs
adult_census	CDTD	0.976 ± 0.001	+0.126	100%	100%	1	9
adult_census	TabSyn	0.850 ± 0.002	+0.030	81%	78%	2	9
adult_census	CopulaGAN	0.820 ± 0.007	+0.117	100%	100%	3	9
adult_census	CTGAN	0.703 ± 0.012	+0.035	85%	89%	4–6	9
adult_census	ForestDiffusion	0.668 ± 0.002	+0.073	100%	100%	4–6	9
adult_census	TabDiff	0.595 ± 0.036	+0.295	93%	100%	4–6	9
adult_census	TVAE	0.300 ± 0.072	+0.226	78%	67%	7–8	9
adult_census	STaSy	0.074 ± 0.037	+0.074	33%	33%	7–11	9
adult_census	GaussianCopula	0.000 ± 0.000	+0.000	0%	0%	8–11	9
adult_census	CoDi	0.000 ± 0.000	+0.000	0%	0%	8–11	9
adult_census	TabDDPM	0.000 ± 0.000	–	–	–	8–11	9
adult_census	*SMOTENC* (control)	0.741 ± 0.007	–	–	–	–	9
bank_marketing	CDTD	0.948 ± 0.003	+0.037	81%	78%	1–2	9
bank_marketing	TabDiff	0.911 ± 0.006	+0.015	44%	56%	2–4	9
bank_marketing	TabDDPM	0.896 ± 0.006	+0.041	89%	89%	2–4	9
bank_marketing	ForestDiffusion	0.854 ± 0.003	+0.262	96%	100%	4–5	9
bank_marketing	TabSyn	0.593 ± 0.148	+0.197	100%	100%	1–11	9
bank_marketing	CTGAN	0.396 ± 0.016	+0.322	100%	100%	5–6	9
bank_marketing	CopulaGAN	0.074 ± 0.006	+0.072	96%	89%	6–7	9
bank_marketing	TVAE	0.001 ± 0.001	+0.002	0%	0%	7–11	9
bank_marketing	GaussianCopula	0.000 ± 0.000	+0.000	0%	0%	7–11	9
bank_marketing	CoDi	0.000 ± 0.000	+0.000	0%	0%	7–11	9
bank_marketing	STaSy	0.000 ± 0.000	–	–	–	7–11	9
bank_marketing	*SMOTENC* (control)	0.741 ± 0.005	–	–	–	–	9
beijing_pm25	ForestDiffusion	0.832 ± 0.012	+0.000	56%	89%	1–2	9
beijing_pm25	TabSyn	0.832 ± 0.011	+0.054	96%	100%	1–2	9
beijing_pm25	TabDiff	0.777 ± 0.013	+0.003	70%	67%	3–4	9
beijing_pm25	CDTD	0.774 ± 0.007	+0.076	100%	100%	3–4	9
beijing_pm25	TVAE	0.699 ± 0.014	+0.154	100%	100%	5	9
beijing_pm25	CopulaGAN	0.545 ± 0.019	+0.124	100%	100%	6	9
beijing_pm25	CTGAN	0.421 ± 0.005	+0.268	100%	100%	7	9
beijing_pm25	GaussianCopula	0.154 ± 0.022	+0.086	89%	89%	8–9	9
beijing_pm25	STaSy	0.067 ± 0.034	+0.067	33%	33%	8–11	9
beijing_pm25	CoDi	0.000 ± 0.000	+0.000	0%	0%	9–11	9
beijing_pm25	TabDDPM	0.000 ± 0.000	–	–	–	9–11	9
beijing_pm25	*SMOTENC* (control)	0.420 ± 0.005	–	–	–	–	9

Continued on the next page.

Table 26, continued.

Dataset	Generator	FCU	Δ to next	sep. pairs	sep. matched	rank set	runs
cdc_diabetes	ForestDiffusion	0.945 ± 0.001	$+0.002$	78%	78%	1–2	9
cdc_diabetes	TabDiff	0.943 ± 0.004	$+0.005$	74%	100%	1–3	9
cdc_diabetes	TabDDPM	0.938 ± 0.003	$+0.063$	100%	100%	2–3	9
cdc_diabetes	TabSyn	0.874 ± 0.010	$+0.497$	100%	100%	4	9
cdc_diabetes	STaSy	0.377 ± 0.012	$+0.350$	100%	100%	5	9
cdc_diabetes	GaussianCopula	0.027 ± 0.003	$+0.006$	85%	89%	6–7	9
cdc_diabetes	CopulaGAN	0.020 ± 0.010	$+0.020$	33%	33%	6–11	9
cdc_diabetes	CTGAN	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
cdc_diabetes	TVAE	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
cdc_diabetes	CDTD	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
cdc_diabetes	CoDi	0.000 ± 0.000	–	–	–	7–11	9
cdc_diabetes	*SMOTENC* (control)	0.797 ± 0.003	–	–	–	–	9
credit_default_taiwan	CDTD	0.878 ± 0.015	$+0.022$	89%	89%	1–3	9
credit_default_taiwan	TabSyn	0.856 ± 0.013	$+0.003$	96%	89%	1–3	9
credit_default_taiwan	ForestDiffusion	0.852 ± 0.005	$+0.156$	100%	100%	1–3	9
credit_default_taiwan	CopulaGAN	0.696 ± 0.010	$+0.077$	100%	100%	4–5	9
credit_default_taiwan	CTGAN	0.619 ± 0.012	$+0.041$	96%	100%	5–7	9
credit_default_taiwan	TabDiff	0.577 ± 0.018	$+0.122$	100%	100%	5–7	9
credit_default_taiwan	TVAE	0.456 ± 0.021	$+0.074$	96%	89%	7–8	9
credit_default_taiwan	STaSy	0.382 ± 0.107	$+0.337$	96%	100%	4–11	9
credit_default_taiwan	GaussianCopula	0.045 ± 0.006	$+0.045$	89%	89%	8–9	9
credit_default_taiwan	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	9–11	9
credit_default_taiwan	TabDDPM	0.000 ± 0.000	–	–	–	9–11	9
credit_default_taiwan	*SMOTENC* (control)	0.926 ± 0.005	–	–	–	–	9
criteo_1m	TabSyn	0.456 ± 0.057	$+0.025$	100%	100%	1–2	9
criteo_1m	CDTD	0.431 ± 0.020	$+0.378$	100%	100%	1–2	9
criteo_1m	GaussianCopula	0.052 ± 0.000	$+0.040$	100%	100%	3	9
criteo_1m	TabDiff	0.012 ± 0.006	$+0.012$	33%	33%	4–7	9
criteo_1m	ForestDiffusion	0.000 ± 0.000	$+0.000$	0%	0%	4–7	9
criteo_1m	TVAE	0.000 ± 0.000	$+0.000$	0%	0%	4–7	9
criteo_1m	TabDDPM	0.000 ± 0.000	–	–	–	4–7	9
criteo_1m	*SMOTENC* (control)	0.806 ± 0.001	–	–	–	–	9
diabetes_130us	TabDiff	0.627 ± 0.023	$+0.384$	100%	100%	1	9
diabetes_130us	TabSyn	0.243 ± 0.030	$+0.092$	85%	89%	2–3	9
diabetes_130us	ForestDiffusion	0.152 ± 0.010	$+0.120$	100%	100%	2–3	9
diabetes_130us	CTGAN	0.032 ± 0.009	$+0.032$	56%	56%	4–11	9
diabetes_130us	CopulaGAN	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	GaussianCopula	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	TVAE	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	CDTD	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	STaSy	0.000 ± 0.000	$+0.000$	0%	0%	4–11	9
diabetes_130us	TabDDPM	0.000 ± 0.000	–	–	–	4–11	9
diabetes_130us	*SMOTENC* (control)	0.135 ± 0.035	–	–	–	–	9
home_credit	TabSyn	0.764 ± 0.003	$+0.012$	78%	67%	1–2	9
home_credit	ForestDiffusion	0.752 ± 0.002	$+0.054$	100%	100%	1–2	9
home_credit	CDTD	0.698 ± 0.009	$+0.102$	100%	100%	3	9
home_credit	GaussianCopula	0.597 ± 0.001	$+0.115$	100%	100%	4	9
home_credit	CopulaGAN	0.481 ± 0.012	$+0.228$	93%	100%	5–6	9
home_credit	STaSy	0.254 ± 0.075	$+0.114$	78%	67%	5–11	9
home_credit	TabDiff	0.139 ± 0.070	$+0.034$	100%	100%	6–11	9
home_credit	CTGAN	0.105 ± 0.019	$+0.105$	100%	100%	6–8	9
home_credit	TVAE	0.000 ± 0.000	-0.000	0%	0%	7–11	9
home_credit	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
home_credit	TabDDPM	0.000 ± 0.000	–	–	–	7–11	9
home_credit	*SMOTENC* (control)	0.777 ± 0.003	–	–	–	–	9

Continued on the next page.

Table 26, continued.

Dataset	Generator	FCU	Δ to next	sep. pairs	sep. matched	rank set	runs
magic	CDTD	0.984 ± 0.002	$+0.038$	100%	100%	1–3	9
magic	ForestDiffusion	0.946 ± 0.004	$+0.046$	81%	67%	2–5	9
magic	TabDiff	0.899 ± 0.011	$+0.134$	100%	100%	2–5	9
magic	TVAE	0.766 ± 0.006	$+0.011$	37%	22%	4–7	9
magic	GaussianCopula	0.755 ± 0.004	$+0.135$	100%	100%	4–7	9
magic	TabSyn	0.620 ± 0.155	$+0.082$	100%	100%	1–11	9
magic	CopulaGAN	0.539 ± 0.011	$+0.217$	100%	100%	6–8	9
magic	CTGAN	0.321 ± 0.013	$+0.014$	100%	100%	7–9	9
magic	TabDDPM	0.307 ± 0.153	$+0.209$	78%	100%	1–11	9
magic	STaSy	0.098 ± 0.034	$+0.098$	67%	67%	8–11	9
magic	CoDi	0.000 ± 0.000	–	–	–	8–11	9
magic	*SMOTENC* (control)	0.000 ± 0.000	–	–	–	–	9
mimic_iii	TabDiff	0.599 ± 0.010	$+0.043$	100%	100%	1–2	9
mimic_iii	TabSyn	0.556 ± 0.031	$+0.377$	100%	100%	1–2	9
mimic_iii	CopulaGAN	0.179 ± 0.026	$+0.009$	96%	100%	3–5	9
mimic_iii	ForestDiffusion	0.169 ± 0.013	$+0.046$	93%	100%	3–5	9
mimic_iii	CTGAN	0.124 ± 0.027	$+0.047$	100%	100%	4–6	9
mimic_iii	GaussianCopula	0.077 ± 0.004	$+0.077$	100%	100%	4–6	9
mimic_iii	TVAE	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
mimic_iii	CDTD	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
mimic_iii	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
mimic_iii	STaSy	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
mimic_iii	TabDDPM	0.000 ± 0.000	–	–	–	7–11	9
mimic_iii	*SMOTENC* (control)	0.698 ± 0.005	–	–	–	–	9
petfinder_tab	TabDiff	0.893 ± 0.012	$+0.120$	93%	100%	1	9
petfinder_tab	CDTD	0.773 ± 0.016	$+0.211$	100%	100%	2	9
petfinder_tab	ForestDiffusion	0.562 ± 0.006	$+0.010$	19%	22%	3–4	9
petfinder_tab	CopulaGAN	0.552 ± 0.007	$+0.151$	100%	100%	3–4	9
petfinder_tab	CTGAN	0.401 ± 0.015	$+0.401$	100%	100%	5	9
petfinder_tab	GaussianCopula	0.000 ± 0.000	$+0.000$	0%	0%	6–11	9
petfinder_tab	TVAE	0.000 ± 0.000	$+0.000$	0%	0%	6–11	9
petfinder_tab	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	6–11	9
petfinder_tab	STaSy	0.000 ± 0.000	$+0.000$	0%	0%	6–11	9
petfinder_tab	TabDDPM	0.000 ± 0.000	$+0.000$	0%	0%	6–11	9
petfinder_tab	TabSyn	0.000 ± 0.000	–	–	–	6–11	9
petfinder_tab	*SMOTENC* (control)	0.542 ± 0.009	–	–	–	–	9
support2	CDTD	0.979 ± 0.003	$+0.089$	100%	100%	1	9
support2	ForestDiffusion	0.890 ± 0.005	$+0.032$	63%	78%	2	9
support2	TabDiff	0.858 ± 0.010	$+0.334$	100%	100%	3	9
support2	GaussianCopula	0.523 ± 0.005	$+0.221$	96%	89%	4–6	9
support2	TVAE	0.302 ± 0.040	$+0.049$	100%	100%	5–8	9
support2	TabSyn	0.253 ± 0.126	$+0.062$	56%	67%	4–11	9
support2	TabDDPM	0.191 ± 0.096	$+0.050$	100%	100%	4–11	9
support2	CopulaGAN	0.142 ± 0.013	$+0.141$	100%	100%	5–8	9
support2	CTGAN	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
support2	CoDi	0.000 ± 0.000	$+0.000$	0%	0%	7–11	9
support2	STaSy	0.000 ± 0.000	–	–	–	7–11	9
support2	*SMOTENC* (control)	0.000 ± 0.000	–	–	–	–	9