LEADERBOARD FIDELITY: AUDITING SYNTHETIC RELEASES OF PRIVATE FINANCIAL DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

Regulated financial data cannot be shared, so synthetic releases are offered as public stand-ins for private fraud-detection benchmarks. A release is used to choose a model, yet it is judged by distributional fidelity and train-on-synthetic accuracy, which do not say whether the models that win on it also win on the private data. We introduce leaderboard fidelity: the rank agreement between the detector leaderboard a release induces and the leaderboard on the private data, calibrated against what resampling the private test period alone produces. Using transfers from a national interbank clearing network, we build a leakage-free benchmark and evaluate sixty releases, five from each of twelve generators, against eleven detectors under equal budgets. Among generators that do not collapse, rank agreement varies more between releases of one generator than between generators, and the releases that reach the calibrated band come from five different generators. None of the distributional criteria we test, nor a train-on-synthetic score, picks the better release of a generator distinguishably better than chance. Measuring rank agreement against the private data does, and only the data holder can do so. The choice carries over to held-out halves and to the later months of the private test period, so a holder can audit several releases and publish the best with the value it scored. When duplicated records cross a random split, most detectors score near perfectly while their order agrees only weakly with the private one. We publish the sixty releases, the protocol, and the code.

1 INTRODUCTION

Practitioners use public benchmarks to select models, adopting the method that wins on the leaderboard. When that benchmark is a synthetic release standing in for private data, the property that matters is therefore whether the ordering of models it induces matches the ordering on the private data, which we call its leaderboard fidelity. Synthetic data is usually evaluated by distributional fidelity, train-on-synthetic-test-on-real (TSTR) accuracy, and distance-based privacy metrics (Giomi et al., 2023; Karst et al., 2024), none of which measures this property. Whether the substitution preserves the decision has rarely been checked, because checking it requires the private data the release is meant to replace.

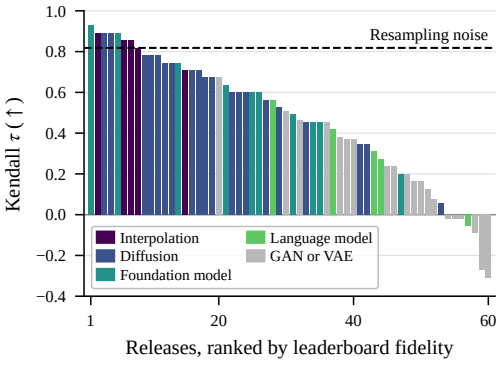


Figure 1: Releases of one generator differ widely in fidelity. The sixty releases ranked by their agreement with the private leaderboard, coloured by generator family. Eight reach the noise band, from five generators, and one generator's releases span τ from $+0.20$ to $+0.93$.

The substitution is common in financial fraud detection, and hard to verify there. Privacy rules and regulation keep real transaction data closed (Chen et al., 2025; Strobel & Shokri, 2022), and interbank transfers are aggregated at a clearing network on behalf of many banks (Altman et al., 2023), so no single institution can publish them. Public benchmarks

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.

in this domain are thus synthetic or heavily transformed (Jesus et al., 2022; Jensen et al., 2023; Alves et al., 2023), and the private counterpart that would validate them is not available to their users.

We can ask this question because we have access to transfers from a national interbank clearing network, where the private records and their bank-assigned suspicion labels are available to us as ground truth. We first build a leakage-free benchmark from these transfers, with account-history features computed only from strictly earlier transfers, a temporal split, and an explicit duplicate audit, and establish a reference leaderboard of eleven detectors tuned under a common budget. Because that leaderboard is itself estimated from a finite test period, we calibrate it by resampling the private test period, which moves its ranking by a known amount, and read every release against that scale instead of against perfect agreement. We then take sixty synthetic releases, five seeds from each of twelve generators, and ask how much of the reference ranking each preserves, both for a user who has only the public data and under the usual TSTR protocol.

Figure 1 summarizes what we find. On this benchmark, rank agreement is not determined by the generator that produced the release. Among generators that do not collapse, more than half the variation in agreement lies between runs of the same generator, and the eight releases that reach the calibrated band come from five different generators. None of the standard criteria we test reliably tells which release of a generator is the better one, and neither does TSTR, which only the data holder can compute in the first place. Measuring agreement against the private data does identify it. A carelessly built benchmark also fails in a way no public score reveals: when duplicated rows cross a random split, most detectors score near perfectly while their ranking drifts from the private one.

This paper makes four contributions.

- **Leaderboard fidelity.** We turn standard rank statistics (Kendall τ against the private leaderboard, preservation of separable detector pairs, and model-selection regret) into a release-level criterion by calibrating them against the sampling noise of the private test period.
- **Release-level evidence.** Sixty releases from twelve generators, showing that fidelity varies more within a generator than between generators once collapsed generators are set aside, and that none of the standard criteria we test reliably orders the releases of one generator.
- **An audit protocol.** Selecting a release by measured agreement on half of the private test period transfers to the other half and to its later months, so a holder can audit several releases and publish the best with its measured fidelity.
- **A leakage-free interbank benchmark.** Past-only history features verified by brute-force recomputation, a temporal split with no train–test duplicates, a calibrated reference leaderboard, and the releases.

2 Preliminaries and Related Work

This section reviews prior work on synthetic tabular data generation, fraud detection benchmarks, and synthetic data evaluation, and highlights the limitations that motivate our study.

2.1 Synthetic Tabular Data

Synthetic tabular data generation has been studied to enable data analysis and machine learning under privacy and data access constraints (Patki et al., 2016; Shi et al., 2025b; Cormode et al., 2025; Hassan et al., 2023), and approaches ranging from statistical oversampling to GAN-based models, diffusion models, and large language model-based techniques have been proposed (Chawla et al., 2002; Xu et al., 2019; Zhao et al., 2021; 2024; Lee et al., 2023; Sattarov et al., 2023; Kotelnikov et al., 2023; Borisov et al., 2023; Fonseca & Bacao, 2023; Kindji et al., 2025). However, existing methods often reproduce overall distributions while failing on rare events and label-conditional structure (Challagundla et al., 2025; Kindji et al., 2025), which is exactly the regime of financial transaction data with rare fraud (Altman et al., 2023; D'souza et al., 2025; Restrepo et al., 2025). Generators built for fraud data address that regime directly: FraudDiffuse conditions a diffusion model on the minority class (Roy et al., 2024), and EmDT clusters fraud patterns before denoising them with a transformer (Kuo & Motsch, 2026). Both aim to augment the minority class of one institution's own dataset rather than to produce a public release that stands in for it, the use we study.

2.2 Fraud Detection Benchmarks

Most public benchmarks for financial fraud detection are tabular and treat transactions as independent records, with credit card and e-commerce datasets as representative examples; they support reproducibility but miss relational structure and cross-institutional context (Shi et al., 2025b; Restrepo et al., 2025; Goldberg et al., 2025). Elliptic (Weber et al., 2019) supplies Bitcoin transaction networks, whose pseudonymous graph differs fundamentally from fiat interbank transfers, and agent-based simulators such as AMLworld (Altman et al., 2023), PaySim (Lopez-Rojas et al., 2016), and AMLgentex (Östman et al., 2025) produce controllable patterns without the distributional complexity of a real regulated payment system. Closest to our setting are releases derived from real institutional data, compared in Appendix A.3. Bank Account Fraud is generated with CTGAN from a real bank's applications and split by time (Jesus et al., 2022), SynthAML synthesizes alerts from a systemically important bank (Jensen et al., 2023), AMLworld publishes simulator output with a temporal protocol (Altman et al., 2023), and an anonymized banking release publishes real records directly (Mollaev et al., 2025). Only two of these four report any rank check: SynthAML compares six classifiers by ROC-AUC without a noise scale, and the anonymized release reports $\tau = 0.94$ against its own un-anonymized version rather than against a generated surrogate. None of these releases can be audited the way we audit ours, because the private source is not available to their users.

2.3 Evaluation of Synthetic Data

Synthetic data is usually judged on three axes at once, statistical utility, downstream performance, and privacy risk (Shi et al., 2025b; Cormode et al., 2025; Lautrup et al., 2025), yet high fidelity guarantees neither usefulness nor safety (Livieris et al., 2024; Restrepo et al., 2025; Fang et al., 2024). A parallel line of work warns that evaluation itself can be invalid. Duplicate records that straddle a train–test split are a recognised leakage type across empirical machine learning (Kapoor & Narayanan, 2023), and tabular benchmarks in particular have been found to contain leaky datasets whose rankings change once splits respect time (Rubachev et al., 2025). Memorizing generators, meanwhile, are rewarded by fidelity-style metrics while distance-based privacy measures fail to detect them (Fang et al., 2024; Ganev & De Cristofaro, 2025). Closest to our question, van Breugel et al. (2023) showed on public tables that model rankings computed on synthetic data can differ from the rankings on real data, and proposed training and evaluating across an ensemble of generator runs instead of a single one. Our study differs from theirs in three ways. We calibrate agreement against the sampling noise of the private test period, and we work from private financial records that the release is actually meant to replace. We then measure how much agreement varies between runs of one generator, and turn that variation into an audit the holder can run before publishing, which Section 5.4 shows recovers about twice as much of the available gain as averaging or their cross-release evaluation.

3 Interbank Transfers and the Benchmark

This section describes the payment system the transfers come from, the benchmark and labels we build from them, and the constraints on releasing such records.

3.1 Interbank Transaction System

A national interbank clearing network mediates account transfers between banks. A sending bank turns a customer request into a standardized message carrying the amount, timestamp, media type (for example mobile or PC banking), fund type, and the two bank identifiers; the network validates and routes it; the receiving bank decides whether to credit the destination account (Figure 2). Each bank may raise a fraud suspicion flag during this process according to its own operational criteria, and no participant observes the full record.

3.2 Data and Labels

The benchmark consists of 94,901 distinct interbank transfers sent by 2,733 accounts to 65,432 receiving accounts between September 2021 and December 2024, of which 1,317 are flagged. A

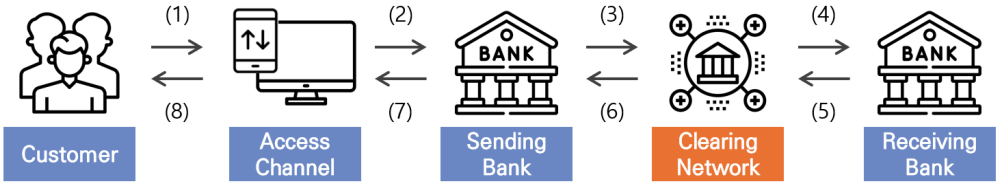


Figure 2: Interbank transfer flow. Steps (1) to (4) carry the transfer request from the customer through an access channel and the sending bank to the clearing network, which validates and routes the message to the receiving bank. Steps (5) to (8) return the credit decision along the same path. Transfers are observed at the network, so no single bank sees the full record.

label is a suspicion flag that a participating bank attaches to an individual transfer when it suspects financial fraud in the course of operations; it is an operational signal rather than an adjudicated fraud outcome, and we treat it as such throughout. Media type and fund type follow standardized code systems (Appendix A) that capture the access channel and the legal character of the transferred funds. Two properties of the sample bound what it can show. Its accounts are not a random sample of the network, since 44% of the sending accounts send at least one flagged transfer, so the flag rate describes the benchmark rather than the network. Receiving accounts are observed only through those senders: for 95% of transfers the receiving account has at most one distinct earlier payer on record, which makes the receiver-side features a truncated view of what the network holds.

3.2.1 Benchmark rows: a case-control sample

The transfers follow a case-control design (King & Zeng, 2001): benign transfers are sampled from the account neighbourhood in which the flagged transfers occur, which keeps the benchmark small enough to train many generators and detectors repeatedly. Its prevalence, 1.28% in the training period, is therefore higher than the 0.37% of the test period before that sampling, and absolute scores should be read accordingly: precision-based metrics depend on prevalence (Davis & Goadrich, 2006; Saito & Rehmsmeier, 2015), and probabilities transfer to the population rate only after prior correction (King & Zeng, 2001; Elkan, 2001). Because our claims concern the ranking of detectors rather than their absolute scores, Section 5.6 verifies that the reference ranking survives on that naturally distributed test set.

3.2.2 Leakage-free features

Raw identifiers are neither released nor used as features. Each transfer is instead described by 24 features in four groups (listed in Appendix A): transaction attributes; bank codes and bank-pair volume; sender history, eleven counts and amounts summarizing the sending account's past activity; and receiver cross-bank history, five features summarizing incoming transfers to the destination account. Only the last group needs records the sending bank does not hold. All history features are computed from transfers strictly earlier than the one being described, over the accounts' full record at the clearing network rather than over the benchmark rows alone. Transfers that share a (date, time band) are excluded because their order is unknown. The features were verified against a brute-force recomputation on 300 random transfers, and none is derived from labels.

3.2.3 Temporal split and duplicates

We split by time: training (September 2021 to December 2023; 63,707 transfers, 1.28% flagged), validation (January to June 2024; 15,427, 2.13%), and test (July to December 2024; 15,767, 1.11%). No test transfer is identical to a training transfer, and 98.9 to 99.4% of rows within each split are unique. Section 5.6 shows why this matters.

3.3 Privacy and Release Constraints

Interbank transaction data contain account identifiers, institutional affiliations, amounts, and timestamps that can reveal individual financial behaviour, so releasing raw records is restricted under privacy law and financial regulation. The applicable credit-information rules permit use only once

re-identification risk is eliminated, without specifying when it is. Before synthesis we therefore replaced account and institution identifiers by consistent one-to-one mappings held only inside the institution and truncated amounts to units; the released artefact carries no identifier column, and we report how many synthetic rows coincide exactly with a private training row (Table 2).

4 Leaderboard Fidelity

This section fixes what we compare and how: the twelve generators, what counts as a single release, and the three leaderboards with the calibrated measure of agreement between them.

4.1 Generative Models

We consider a diverse set of synthetic data generation models that represent different generative paradigms and are suitable for financial interbank transaction data. They are interpolation-based oversampling (SMOTE) (Chawla et al., 2002); four GAN- and VAE-based models (CTGAN, TVAE, CTAB-GAN, CTAB-GAN+) (Xu et al., 2019; Zhao et al., 2021; 2024); four diffusion models (TabDDPM, TabSyn, TabDiff, and the finance-specific FinDiff) (Kotelnikov et al., 2023; Zhang et al., 2024; Shi et al., 2025a; Sattarov et al., 2023; Li et al., 2025); and a language model fine-tuned on the target table (GReaT) (Borisov et al., 2023). We add a tabular foundation model sampled with SGLD (TabPFGen) and a variant of it that matches the private class ratio (Ma et al., 2024), for twelve generators in total. Appendix C lists the configurations.

4.2 Releases

A release consists of one synthetic dataset per temporal split, each produced by a generator trained only on the corresponding private split and sampled to the same size, so that a user can reproduce the temporal protocol from the public data alone. Because a release is one draw from a generator, we produce five of them per generator, one per seed, and report every result at the level of the individual release. To avoid favouring generators we could tune more carefully, every generator uses its official default configuration where one exists; Appendix C lists every deviation, chiefly fewer epochs or steps for the three most expensive methods to fit about three GPU-hours per split. The budget bounds training only, and the GPU was shared, so 21 generation runs exceeded three hours in wall-clock. Appendix C documents every deviation and the realized cost of each generator.

Table 1: Reference leaderboard on the private test period. Test PR-AUC, seed mean, with a 95% bootstrap CI. Recall and ROC-AUC are in Appendix H.

Detector	PR-AUC	95% CI
LightGBM	0.922	[0.890, 0.949]
XGBoost	0.916	[0.883, 0.944]
Random forest	0.915	[0.882, 0.944]
CatBoost	0.912	[0.880, 0.942]
HistGB	0.907	[0.874, 0.940]
Extra trees	0.882	[0.840, 0.918]
MLP	0.833	[0.787, 0.876]
Logistic regression	0.774	[0.721, 0.828]
Decision tree	0.769	[0.715, 0.818]
k-NN	0.732	[0.670, 0.795]
Naive Bayes	0.459	[0.402, 0.524]

4.3 Leaderboards, settings, and metrics

Suspicion flags are strongly associated with amount and channel: a gradient-boosted model using only the five transaction features reaches ROC-AUC 0.990 while its PR-AUC is only 0.432, so ROC-AUC is uninformative for ranking detectors here and we use PR-AUC throughout (Davis & Goadrich, 2006; Saito & Rehmsmeier, 2015).

A leaderboard is the vector of test PR-AUC scores of a fixed set of $M = 11$ detectors: Gaussian naive Bayes, decision tree, logistic regression, k-nearest neighbours, a multilayer perceptron, random forest, extra trees, histogram gradient boosting, LightGBM, XGBoost, and CatBoost. Every detector receives the same budget: 20 trials of TPE search maximizing PR-AUC on the validation split of the data it is trained on, followed by five retrainings with different seeds; we report the seed mean, and gradient-boosted models use the validation split for early stopping. We compare three leaderboards. The reference leaderboard L^{real} trains, tunes, and tests on the private data. For a synthetic release S, the public-user leaderboard $L^{S \to S}$ trains, tunes, and tests only on S, which is

Table 2: Among generators that do not collapse, leaderboard fidelity varies more between runs of one generator than between generators. $\tau_{S\to S}$ and $\tau_{S\to R}$ are means over each generator's five releases, one per seed, and [min, max] is the range of $\tau_{S\to S}$; the other columns use the seed-0 release. Arrows give the better direction; DCR near 1.0 means as far from the private data as a fresh real sample, and bold marks the highest $\tau_{S\to S}$. TVAE keeps one positive row per split, so its τ is not interpretable. Definitions are in Appendix B; separable pairs, regret and the remaining metrics are in Table 11.

Generator	Standard metrics		Leaderboard fidelity			Privacy	
	KS ↓	TSTR ↑	$\tau_{S\to S}$ ↑	[min, max]	$\tau_{S\to R}$ ↑	DCR → 1	MIA → .5
SMOTE	0.04	0.87	**+0.83**	[+0.71, +0.89]	+0.76	0.42	0.58
TabSyn	0.01	0.86	+0.72	[+0.53, +0.89]	+0.82	1.12	0.52
TabDiff	0.04	0.77	+0.69	[+0.35, +0.89]	+0.78	1.59	0.50
TabPFGen	0.27	0.80	+0.67	[+0.45, +0.89]	+0.75	4.64	0.43
FinDiff	0.16	0.77	+0.67	[+0.60, +0.78]	+0.65	4.87	0.51
TabPFGen-prior	0.15	0.82	+0.53	[+0.20, +0.93]	+0.72	4.70	0.43
TabDDPM	0.03	0.40	+0.43	[+0.05, +0.71]	+0.54	0.96	0.52
TVAE	0.30	0.05	+0.36	[+0.08, +0.51]	+0.25	1.66	0.50
GReaT	0.12	0.47	+0.30	[−0.05, +0.56]	+0.38	1.03	0.49
CTGAN	0.25	0.01	+0.27	[−0.27, +0.67]	−0.11	3.92	0.45
CTAB-GAN	0.32	0.03	+0.13	[−0.02, +0.38]	+0.16	4.89	0.45
CTAB-GAN+	0.25	0.01	−0.02	[−0.31, +0.20]	−0.08	4.34	0.46

all an outside researcher can do, and the TSTR leaderboard $L^{S\to R}$ trains and tunes on S but tests on the private test period, the protocol under which synthetic fraud data is usually reported (Mahrous & Di Pietro, 2025).

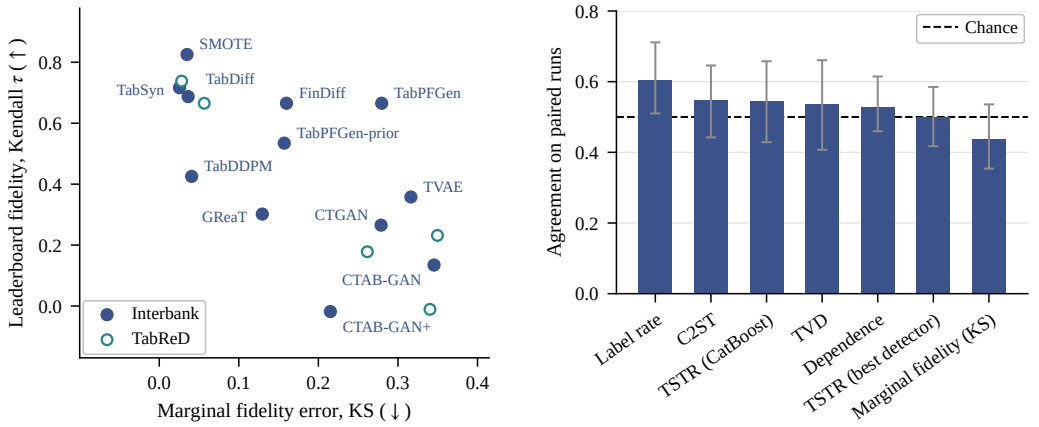
Write $L^{\text{real}} = (s_1, \ldots, s_M)$ for the reference scores and L for a candidate leaderboard over the same M detectors. Rank agreement is Kendall's τ_b, which accounts for the ties that degenerate releases produce in abundance. Because L^{real} is itself estimated from a finite test period, we calibrate τ_b against what that estimation noise alone produces: over $B = 1{,}000$ class-stratified bootstrap resamples of the private test period we recompute the reference leaderboard, each score again as the mean PR-AUC of the five seeds on the resample, and its τ_b against the full-sample one. The mean of that distribution is the noise ceiling and its 5th percentile is the lower edge of the noise band. The ceiling is an expected agreement, not an upper bound, and one of our releases exceeds it. The separable pairs $\mathcal{P}$ are the detector pairs whose order in L^{real} holds in at least 97.5% of the resamples, and separable-pair preservation is the fraction of $\mathcal{P}$ whose order L keeps. Selection regret is $s_{i^\star} - s_{\hat{\imath}}$, the private PR-AUC lost by deploying the detector $\hat{\imath}$ that tops L instead of the reference winner $i^\star$. All three are properties of a single release; a generator-level number is the mean over its five releases. We also report the standard criteria used to choose generators (Appendix B).

5 Experiments

We first fix the reference leaderboard and its noise scale, then read the sixty releases against it.

5.1 The reference leaderboard and its noise scale

Table 1 reports the reference leaderboard. The detectors form tiers, with five tree ensembles on top and naive Bayes last. Of the 55 detector pairs, 44 are separable; the rest lie almost entirely inside the top tier, whose four leaders cannot be told apart on this test period. Resampling the test period moves the ranking more than retraining does (bootstrap $\tau = 0.923$ against seed-to-seed 0.971), so a release with τ near 0.92 agrees with the reference as closely as a resample of the private test period does. Resampling accounts or weeks instead of transfers leaves this scale nearly unchanged (Appendix F).



(a) Across generators: one point per generator, its mean marginal fidelity error against its mean leaderboard fidelity, on the interbank benchmark (filled) and in the public replication of Appendix E (open).

(b) Within a generator: the share of same-generator release pairs in which each criterion prefers the release with the higher τ, with descriptive 95% intervals from resampling generators.

Figure 3: Distributional criteria separate generators but not reliably the releases of one generator.

5.2 Fidelity varies between releases of one generator

Table 2 and Figure 1 report leaderboard fidelity for the sixty releases. At the generator level the ordering is as expected, with interpolation first, diffusion and foundation models next, and GAN-, VAE-, and language-model-based generators last. At the level of individual releases, eight releases reach the noise band, and they come from five different generators, four from SMOTE and one each from TabSyn, TabDiff, TabPFGen, and its prior-matched variant. A modern generator can therefore produce a release whose leaderboard is as faithful as a resample of the private data, but it does not do so reliably. Across the sixty releases τ tracks the other two criteria, at $\rho = 0.95$ with separable-pair preservation and $\rho = -0.70$ with selection regret, and the eight releases in the band cost at most 0.006 of private PR-AUC when the detector they crown is deployed, against 0.05 on average and 0.19 at worst for the thirty releases below $\tau = 0.5$. The generator explains only part of the variation in τ across releases. Decomposing the sum of squares of τ over the sixty releases, the share lying within generators is 0.35, and it rises to 0.53 once the four collapsed GAN and VAE generators are excluded. Within one generator the spread is wide enough to reverse the conclusion drawn from a single run. TabPFGen-prior ranges from $+0.20$ to $+0.93$ across its five seeds, and TabDiff from $+0.35$ to $+0.89$.

The GAN- and VAE-based releases fail in a way that their mean τ does not show. Their leaderboards are degenerate: in every release CTAB-GAN assigns all eleven detectors a PR-AUC within 0.006 of each other, CTAB-GAN+ within 0.007, and CTGAN within 0.021, while every TVAE release contains exactly one positive row per split, so most detectors tie at the prevalence floor. When the scores that induce a ranking differ by a few thousandths, as for CTAB-GAN and CTAB-GAN+, the ranking is mostly noise. A public benchmark should therefore be identified by its release, not only by its generator.

5.3 No standard criterion reliably identifies the better release

Across the twelve generators (Figure 3a) the criteria are associated with agreement: Spearman ρ is 0.68 for marginal fidelity, 0.86 for dependence-structure error, and 0.90 for TSTR with CatBoost. The choice a holder actually faces, however, is which release to publish. Figure 3(b) takes every pair of releases from one generator, 112 pairs in all, and asks how often a criterion prefers the release with the higher τ. Marginal fidelity is right in 44% of the pairs, below the 50% of a coin flip, and the best criterion, label-rate error, reaches 60%. Since each release enters up to four pairs, we test by shuffling a criterion's values among the releases of each generator, and no criterion separates from chance: the smallest two-sided p is 0.14, for label-rate error, before any correction for testing seven

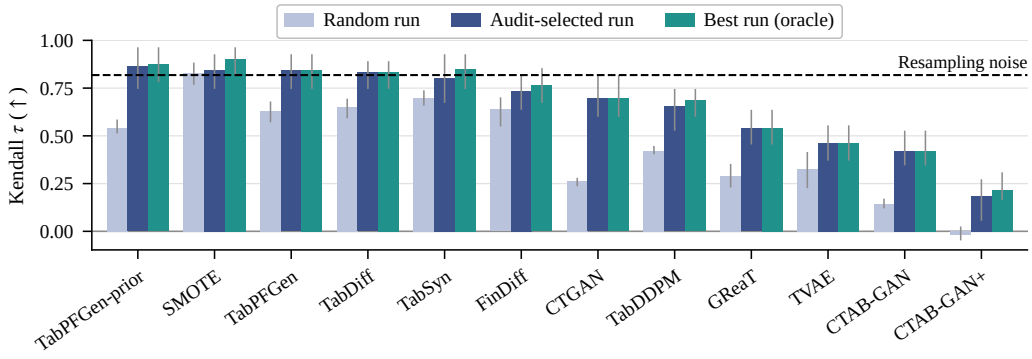


Figure 4: Measuring a release against the private data recovers most of the agreement a random release loses. Over 200 stratified half-splits of the private test period, each generator's release with the highest τ on one half is scored on the other. Bars are means and whiskers 5th to 95th percentiles; the dashed line is the band edge of Figure 1, shown for scale.

of them. The test would usually miss an agreement near 0.6 (Appendix F), so this is an absence of evidence, not evidence of no signal. In practice the signal is small. Publishing the release with the lowest marginal fidelity error costs 0.22 of τ relative to the best release of the same generator, barely different from the 0.23 that choosing at random costs, and even label-rate error costs 0.16. TSTR does not solve the problem. It ranks generators well but reaches 54% within a generator, and it needs the private test period, so an outside user cannot compute it either. The same holds on a public dataset (Appendix E): the criteria are positively associated with agreement across generators, but within one none separates from chance.

5.4 An audit turns the variance into a choice

The remaining option is for the holder to measure τ directly. Figure 4 tests whether such an audit generalizes, selecting on one stratified half of the private test period and evaluating on the other. Selecting this way among all sixty releases yields $\tau = 0.83$ on the held-out half, against 0.45 for a release taken at random and 0.92 for the best release in hindsight. The gain is largest where a generator's releases disagree most, $+0.32$ for TabPFGen-prior and $+0.25$ for GReaT, and it disappears where they already agree, $+0.01$ for SMOTE. Two remedies need no private data. Averaging a generator's five leaderboards lifts agreement on the held-out half from 0.45 to 0.55 over the twelve generators; the cross-release evaluation of van Breugel et al. (2023), training on one release and testing on the other four, raises it to 0.50 when the four scores are averaged and to 0.54 when the four test splits are pooled. Selecting on the measured half reaches 0.66, within 0.02 of the best release averaged over generators, so measuring recovers 92% of what separates a random draw from the best one, against 44% for averaging and 22% or 41% for cross-evaluation, an advantage that comes from access to the private data. Selection also carries over from July–September to October–December, to a split whose scored half was kept out of generation, and weakly to unused detectors (Appendix F).

5.5 Proximity to the private data is a property of interpolation

A distance to the closest private record is hard to interpret on its own, so we calibrate it against real transfers of the same period that were not sampled into the benchmark. A DCR ratio of 1.0 means the release sits no closer than a fresh real sample does. Table 2 reports that ratio with a nearest-neighbour membership-inference attack that sees only the release: five releases sit within a factor of two of the baseline and six sit three to five times beyond it, all with attack AUC between 0.43 and 0.52. SMOTE is the exception. Interpolation sometimes lands exactly on a real transfer, about 1% of every split, and we delete every such row before publication; even then its ratio is 0.42 and membership inference reaches 0.58. A half-split test, needed because the holdout is not exchangeable with the benchmark rows (Appendix F), confirms it: when one half of the private training period generates the release and the other supplies non-members, a real second half gives a

Table 3: Effect of dataset construction on the private-data leaderboard. Test PR-AUC of every detector, columns ordered by the reference leaderboard (first row); τ is each row's agreement with it. The last row also replicates 2,157 distinct rows to 91,005 before splitting. Bold marks each row's highest score and ties at this precision. Separable pairs: 44, 36, and 26 of 55.

Construction	LGBM	XGB	RF	CB	HGB	ET	MLP	LR	DT	kNN	NB	τ
Temporal split	**0.92**	**0.92**	0.91	0.91	0.91	0.88	0.83	0.77	0.77	0.73	0.46	1.00
Random split	0.92	**0.93**	0.92	0.92	0.92	0.91	0.86	0.77	0.83	0.78	0.37	0.82
Duplicated + random split	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.97	0.98	**1.00**	0.48	0.43

ratio of 1.00 and an attack AUC of 0.50, while SMOTE gives 0.61 and 0.58. This proximity comes from interpolation, and across generators it is not reliably associated with τ.

5.6 A DUPLICATED BENCHMARK LOOKS PERFECT BUT RANKS POORLY

Table 3 isolates how the benchmark itself is built. Splitting at random instead of by time leaves the ranking nearly intact ($\tau = 0.82$). Replicating a small pool of distinct transfers before splitting, a plausible outcome of resizing a dataset to a target size, is far more damaging. Almost every test row then also appears in training, eight of the eleven detectors saturate above 0.999 PR-AUC with six of them exactly tied at 1.000, and agreement collapses to $\tau = 0.43$. Unlike a collapsed generator, which shows itself with scores near 0.03, this leaderboard looks excellent while ranking the detectors poorly, and which of the eight tied at the top a user deploys is arbitrary, costing anywhere from nothing to 0.19 of private PR-AUC. The case-control sampling itself is benign: at the natural prevalence of 0.37% the reference ranking and every separable pair are preserved. Duplication is therefore a validity problem as well as a privacy problem, because it turns a fraud benchmark into a lookup task (Kapoor & Narayanan, 2023; Rubachev et al., 2025).

Appendix H reports one further use of a release, as extra training data, where the gain tracks leaderboard fidelity, and a feature ablation on the private data.

6 DISCUSSION AND CONCLUSION

Some synthetic releases preserve the private model ranking. Eight of sixty, from five generators, agree with it about as well as a resample of its test period does. Which release does so is not reliably predicted by its generator, the standard criteria we test, or TSTR, but measuring agreement against the private data identifies it, also on held-out halves and later months of the test period, so a holder can audit and disclose leaderboard fidelity instead of assuming it. Errors in how the benchmark is built are harder to see: when duplicated rows cross a random split, most detectors score near perfectly and their order agrees only weakly with the private one.

Six recommendations follow for a holder who plans to publish a synthetic benchmark.

- Generate several releases per generator rather than one, and treat each as a separate artefact.
- Audit them against the private leaderboard, and publish the measured fidelity alongside the data.
- Publish the release that the audit selected, not the one with the best distributional scores.
- Where no audit is possible, publish several releases so that users can average their leaderboards.
- Split by time, and delete rows identical to private records before publishing.
- Report proximity to the private data, because the most faithful release may also be the closest one.

Our study has limits. The labels are suspicion flags rather than confirmed fraud (Deprez et al., 2025; Alves et al., 2023), so absolute scores overstate deployment performance. The benchmark comes from one payment infrastructure, its benign class is a case-control sample, and the protocol is demonstrated on two datasets, too few to say how often a distributional screen misleads; on the public one the generator explains more of the variation (Appendix E). The audit is validated within one test period, and only weakly for an entirely new detector set (Appendix F). Transfers are treated as tabular records, generators ran at official defaults, and the release has no formal privacy guarantee (Abadi et al., 2016).

AI USE STATEMENT

We used a large language model in three ways: to search for and surface related work, which the authors then read and selected; to draft sections of this paper from results and analyses the authors had already produced; and to polish wording for clarity. Every AI-assisted passage was reviewed, edited, and checked against our experimental records and the cited sources by the authors before inclusion. The synthetic datasets studied here are the object of the paper rather than a writing aid: they are produced by the tabular generative models of Section 4 under the configurations of Appendix C. The authors take full responsibility for the content of this paper.

ETHICS STATEMENT

Data governance. The transfers underlying this work are operational records of a national interbank clearing network and are accessed under an institutional agreement; no author had access to customer identities. No raw record is released. Before any generator was trained, account and institution identifiers were replaced by consistent one-to-one mappings, held only inside the institution and never shared with the release, and amounts were truncated; the released artefact contains no identifier column at all, and the raw extract, the identifier-bearing intermediate file, and the mapping remain inside the institution. The access to these records and the publication of the synthetic datasets were approved by the data governance function of the institution that holds them, under the terms we follow here: no raw record leaves the institution, the released artefact carries no identifier column, and every synthetic row identical to a private record is deleted before publication. We name neither the institution nor the network, as double-blind review requires.

Privacy of the release. We do not claim a formal privacy guarantee, and we therefore measured the residual risk rather than asserting its absence (Section 5.5): exact-copy rate, distance to the closest private record calibrated against real transfers of the same period that are not in the benchmark, and a nearest-neighbour membership-inference attack that sees only the release. Every row identical to a private record is deleted from every release before publication, as in Jesus et al. (2022): this affected all fifteen SMOTE splits (about 1% of each, none of them flagged), three TabDDPM runs, and two GReaT runs, and every result we report is computed on the filtered releases. Every release except SMOTE sits near that baseline or beyond it: across all five seeds the nearest are TabDDPM and GReaT at 0.95 to 0.97, and attack AUC stays between 0.43 and 0.52. SMOTE sits more than twice as close to the private data as a fresh real sample, and membership inference against it reaches AUC 0.584. An exchangeable half-split test confirms that this proximity is real rather than an artefact of the comparison sample: a control built from a real second half of the training period gives a distance ratio of 1.00 and an attack AUC of 0.50, while SMOTE gives 0.61 and 0.58. Because it interpolates between real neighbours, this is a property of the method rather than of a particular run, and we disclose it rather than presenting the release as risk-free.

Labels and individuals. The labels are suspicion flags raised in the course of operations, not adjudicated fraud outcomes, and some flagged transfers are certainly legitimate. We use them as a prediction target and make no claim about any individual; because the release carries no identifiers, it cannot be used to attribute suspicion to a person or an account. Readers should not read the high absolute scores as evidence that fraud is easy to detect in deployment.

Intended use and dual use. The releases are intended for methodological research on fraud detection and on the evaluation of synthetic data. Detection methods can in principle inform evasion, but the released data are synthetic, aggregated feature summaries of a single payment system, and contain no operational thresholds or rules; we judge the marginal uplift to an adversary to be small relative to the benefit of a public benchmark in a domain that currently has none. The work involves no human subjects and required no IRB review.

REPRODUCIBILITY STATEMENT

The private transfers cannot be released, which is the premise of this paper rather than an obstacle to reproducing it: every claim we make about what a release looks like to its users can be rederived from the artefacts we publish, while the reference leaderboard, the privacy calibration, and the augmentation experiment require the private data and can only be re-run inside the institution.

We publish all sixty releases rather than the one an audit would select, each with its measured agreement in the manifest, so that the variation this paper reports can be re-examined and a user can pool or choose as they prefer; each is one dataset per temporal split with the feature schema of Section 3, so the public-user leaderboard ($S \to S$) of any release can be recomputed from the release alone, and a script checks that the published files hold exactly the values we evaluated. The standard metrics compare a release with the private training period, so we publish them as measured values rather than as something a reader can recompute. The evaluation protocol is released as code: dataset construction with the past-only history rules and the duplicate audit, the leaderboard and tuning procedure, the bootstrap noise ceiling, the leaderboard-fidelity metrics, the feature ablation, and the privacy evaluation, together with the configuration of every generator (Appendix C), the search space and budget of every detector (Appendix D), fixed seeds, and pinned library versions. The result tables and the three result figures are emitted directly by that code from the stored result files, so the numbers they carry are not transcribed by hand; the configuration tables in the appendix are written by hand from the same configuration files. For the private data we report what is needed to interpret the results without disclosing records: period, size, prevalence, split boundaries, feature definitions, and the duplicate and leakage audits (Section 3, Appendix A). A datasheet for the releases is given in Appendix G, and the releases and the code are available at `https://anonymous.4open.science/r/Fin-Syn-0CD4`.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019.

Erik Altman, Jovan Blanuša, Luc Von Niederhäusern, Béni Egressy, Andreea Anghel, and Kubilay Atasu. Realistic synthetic financial transactions for anti-money laundering models. *Advances in Neural Information Processing Systems*, 36:29851–29874, 2023.

Jean V Alves, Diogo Leitão, Sérgio Jesus, Marco OP Sampaio, Pedro Saleiro, Mário AT Figueiredo, and Pedro Bizarro. FiFAR: A fraud detection dataset for learning to defer. *arXiv preprint arXiv:2312.13218*, 2023.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *International Conference on Learning Representations*, 2023.

Raju Challagundla, Mohsen Dorodchi, Pu Wang, and Minwoo Lee. Synthetic tabular data generation: A comparative survey for modern techniques. *arXiv preprint arXiv:2507.11590*, 2025.

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.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

Yisong Chen, Chuqing Zhao, Yixin Xu, Chuanhao Nie, and Yixin Zhang. Year-over-year developments in financial fraud detection via deep learning: A systematic literature review. *arXiv preprint arXiv:2502.00201*, 2025.

Graham Cormode, Samuel Maddock, Enayat Ullah, and Shripad Gade. Synthetic tabular data: Methods, attacks and defenses. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5989–5998, 2025.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 233–240, 2006.

Bruno Deprez, Wei Wei, Wouter Verbeke, Bart Baesens, Kevin Mets, and Tim Verdonck. Advances in continual graph learning for anti-money laundering systems: A comprehensive review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 17(3):e70040, 2025.

Annie D'souza, M Swetha, and Sunita Sarawagi. Synthetic tabular data generation for imbalanced classification: The surprising effectiveness of an overlap class. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16127–16134, 2025.

Charles Elkan. The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence*, volume 17, pp. 973–978. Lawrence Erlbaum Associates Ltd, 2001.

Zhengyu Fang, Zhimeng Jiang, Huiyuan Chen, Xiao Li, and Jing Li. Understanding and mitigating memorization in diffusion models for tabular data. *arXiv preprint arXiv:2412.11044*, 2024.

Joao Fonseca and Fernando Bacao. Tabular and latent space synthetic data generation: a literature review. *Journal of Big Data*, 10(1):115, 2023.

Georgi Ganev and Emiliano De Cristofaro. The inadequacy of similarity-based privacy metrics: Privacy attacks against "truly anonymous" synthetic datasets. In *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 4007–4025. IEEE, 2025.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64 (12):86–92, 2021.

Matteo Giomi, Franziska Boenisch, Christoph Wehmeyer, and Borbála Tasnádi. A unified framework for quantifying privacy risk in synthetic data. *Proceedings on Privacy Enhancing Technologies*, 2023(2):312–328, 2023.

Alexander Goldberg, Giulia Fanti, Nihar Shah, and Steven Wu. Benchmarking fraud detectors on private graph data. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 378–389, 2025.

Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In *International Conference on Learning Representations*, volume 2025, pp. 77899–77935, 2025.

Conor Hassan, Robert Salomone, and Kerrie Mengersen. Deep generative models, synthetic tabular data, and differential privacy: An overview and synthesis. *arXiv preprint arXiv:2307.15424*, 2023.

Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen, and Michalis Vazirgiannis. DGraph: A large-scale financial dataset for graph anomaly detection. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22765–22777, 2022.

Rasmus Ingemann Tuffveson Jensen, Joras Ferwerda, Kristian Sand Jørgensen, Erik Rathje Jensen, Martin Borg, Morten Persson Krogh, Jonas Brunholm Jensen, and Alexandros Iosifidis. A synthetic data set to benchmark anti-money laundering methods. *Scientific Data*, 10(1):661, 2023.

Sérgio Jesus, José Pombal, Duarte Alves, André Cruz, Pedro Saleiro, Rita Ribeiro, João Gama, and Pedro Bizarro. Turning the tables: Biased, imbalanced, dynamic tabular datasets for ML evaluation. *Advances in Neural Information Processing Systems*, 35:33563–33575, 2022.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9), 2023.

Fabian Sven Karst, Sook-Yee Chong, Abigail A Antenor, Enyu Lin, Mahei Manhai Li, and Jan Marco Leimeister. Generative AI for banks: Benchmarks and algorithms for synthetic financial transaction data. *arXiv preprint arXiv:2412.14730*, 2024.

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. *Advances in Neural Information Processing Systems*, 30, 2017.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

G Charbel N Kindji, Lina M Rojas-Barahona, Elisa Fromont, and Tanguy Urvoy. Tabular data generation models: An in-depth survey and performance benchmarks with extensive tuning. *Neurocomputing*, pp. 131655, 2025.

Gary King and Langche Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2): 137–163, 2001.

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.

En-Ya Kuo and Sebastien Motsch. EmDT: Embedding diffusion transformer for tabular data generation in fraud detection. *arXiv preprint arXiv:2603.13566*, 2026.

Anton D Lautrup, Tobias Hyrup, Arthur Zimek, and Peter Schneider-Kamp. SynthEval: a framework for detailed utility and privacy evaluation of tabular synthetic data. *Data Mining and Knowledge Discovery*, 39(1):6, 2025.

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.

Zhong Li, Qi Huang, Lincen Yang, Jiayang Shi, Zhao Yang, Niki van Stein, Thomas Bäck, and Matthijs van Leeuwen. Diffusion models for tabular data: Challenges, current progress, and future directions. *arXiv preprint arXiv:2502.17119*, 2025.

Ioannis E Livieris, Nikos Alimpertis, George Domalis, and Dimitris Tsakalidis. An evaluation framework for synthetic data generation models. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pp. 320–335. Springer, 2024.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *International Conference on Learning Representations*, 2017.

Edgar Lopez-Rojas, Ahmad Elmir, and Stefan Axelsson. PaySim: A financial mobile money simulator for fraud detection. In *28th European Modeling and Simulation Symposium (EMSS)*, pp. 249–255. Dime University of Genoa, 2016.

Junwei Ma, Apoorv Dankar, George Stein, Guangwei Yu, and Anthony Caterini. TabPFGen–tabular data generation with TabPFN. *arXiv preprint arXiv:2406.05216*, 2024.

Ahmed Mahrous and Roberto Di Pietro. TSTR for financial fraud: Learning to detect manipulation without real data. In *Proceedings of the 6th ACM International Conference on AI in Finance*, pp. 71–79, 2025.

Dzhambulat Mollaev, Ivan Kireev, Mikhail Orlov, Alexander Kostin, Ivan Karpukhin, Maria Postnova, Gleb Gusev, and Andrey Savchenko. Multimodal banking dataset: Understanding client needs through event sequences. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 6476–6480, 2025.

Johan Östman, Edvin Callisen, Anton Chen, Kristiina Ausmees, Emanuel Gårdh, Jovan Zamac, Jolanta Goldsteine, Hugo Wefer, Simon Whelan, and Markus Reimegård. AMLgentex: Mobilizing data-driven research to combat money laundering. *arXiv preprint arXiv:2506.13989*, 2025.

Inkit Padhi, Yair Schiff, Igor Melnyk, Mattia Rigotti, Youssef Mroueh, Pierre Dognin, Jerret Ross, Ravi Nair, and Erik Altman. Tabular transformers for modeling multivariate time series. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3565–3569. IEEE, 2021.

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

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 2018.

Maria Fernanda Davila Restrepo, Benjamin Wollmer, Fabian Panse, and Wolfram Wingerath. Benchmarking tabular data synthesis: evaluating tools, metrics, and datasets on prosumer hardware for end-users. *Data Science Journal*, 24:37–37, 2025.

Ruma Roy, Darshika Tiwari, and Anubha Pandey. FraudDiffuse: Diffusion-aided synthetic fraud augmentation for improved fraud detection. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 90–98, 2024.

Ivan Rubachev, Nikolay Kartashev, Yury Gorishniy, and Artem Babenko. TabReD: Analyzing pitfalls and filling the gaps in tabular deep learning benchmarks. In *International Conference on Learning Representations*, volume 2025, pp. 35166–35202, 2025.

Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*, 10(3):e0118432, 2015.

Timur Sattarov, Marco Schreyer, and Damian Borth. FinDiff: Diffusion models for financial tabular data generation. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, pp. 64–72, 2023.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. TabDiff: a mixed-type diffusion model for tabular data generation. In *International Conference on Learning Representations*, volume 2025, pp. 37353–37375, 2025a.

Ruxue Shi, Yili Wang, Mengnan Du, Xu Shen, Yi Chang, and Xin Wang. A comprehensive survey of synthetic tabular data generation. *arXiv preprint arXiv:2504.16506*, 2025b.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017.

Martin Strobel and Reza Shokri. Data privacy and trustworthy machine learning. *IEEE Security & Privacy*, 20(5):44–49, 2022.

Boris van Breugel, Zhaozhi Qian, and Mihaela van der Schaar. Synthetic data, real errors: how (not) to publish and use synthetic data. In *International Conference on Machine Learning*, pp. 34793–34808. PMLR, 2023.

Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591*, 2019.

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

Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *International Conference on Learning Representations*, volume 2024, pp. 52829–52857, 2024.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. CTAB-GAN: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pp. 97–112. PMLR, 2021.

Zilong Zhao, Aditya Kunar, Robert Birke, Hiek Van der Scheer, and Lydia Y Chen. CTAB-GAN+: Enhancing tabular data synthesis. *Frontiers in Big Data*, 6:1296508, 2024.

APPENDIX OVERVIEW

Appendix A documents the data, the code systems, the 24 features, and the dataset statistics, and compares the benchmark with the public financial datasets discussed in Section 2. Appendix B defines the standard metrics, and Appendix C and Appendix D give the configuration and hardware of every generator and the search space of every detector. Appendix E replicates the whole protocol on public data. Appendix F reports the robustness checks, Appendix G is the datasheet for the releases, and Appendix H collects the remaining result tables.

A DATA AND FEATURES

Table 4: Media type and fund type code definitions in the interbank transaction dataset

Type	Code	Description
Media	01	PC banking
	02	Internet banking
	03	Telephone banking
	04	Mobile banking
	05	Single transfer
	06	Others
	07	Bulk transfer
Fund	00, 03, 04	General-purpose funds
	01, 02	Salary or dividend-related funds
	11–30	Legally protected (non-seizable) benefits

Table 4 summarizes the media type and fund type codes in the interbank transaction dataset. Media type codes indicate the transaction channel through which a transfer was initiated, such as PC banking, mobile banking, or bulk transfer, providing contextual signals related to customer access channels and transaction initiation patterns. Fund type categories reflect institutional and legal characteristics of transferred funds, distinguishing general-purpose transfers from income-related funds and protected social benefits. In particular, protected fund categories correspond to non-seizable benefits defined by regulation, which impose structural constraints on transaction behavior and may influence observed fraud risk patterns.

A.1 FEATURE DEFINITIONS

Table 5 lists the 24 features. Write q for the transfer being described, t_q for its timestamp in hours, $p(q)$ for its sending account, $r(q)$ for its receiving account, and $\mathcal{H}(q) = \{x : t_x < t_q\}$ for the transfers that occur strictly earlier in the accounts' full record at the clearing network. All history features are computed over $\mathcal{H}(q)$ only. Timestamps are resolved to (date, time band); transfers sharing a band are not ordered, so a transfer never enters its own history nor that of any transfer in the same band. Features that are undefined because an account has no prior activity take the value -1, which tree-based detectors can separate from any genuine count.

Five of the 24 features, the receiver history, can only be computed at the clearing network: a sending bank does not observe what other institutions send to an account it does not hold. Bank-pair volume and the sender's distinct destination institutions consist of transfers the sending bank itself originates, so it can compute them. The ablation of Table 13 isolates the five network-only features; in this benchmark they see a receiver only through the sending accounts it contains (Section 3.2).

A.1.1 VERIFICATION

We drew 300 transfers at random and recomputed every history feature by brute-force scanning of the accounts' full record, with the strict inequality on time applied directly; all values matched. We also checked that no transfer appears in its own history, that no feature is a function of the label, and that the training, validation, and test periods are disjoint.

Table 5: Feature definitions. The Visible to column indicates which party can compute the feature from its own records: the sending bank (S) or only the clearing network (N).

Group	Feature	Visible to	Definition
Transaction	amount	S	transferred amount (unit-truncated)
	hour_band	S	time band of the transfer
	dow	S	day of week
	media_type	S	access channel code (Table 4)
	fund_type	S	fund category code (Table 4)
Bank	withdraw_bank	S	sending institution code
	deposit_bank	S	receiving institution code
	bankpair_n_30d	S	bank-pair transfers in the previous 30 days
Sender history	s_n_prev	S	$\lvert\{x \in \mathcal{H}(q) : p(x) = p(q)\}\rvert$
	s_n_7d, s_n_30d	S	the same over the previous 7 and 30 days
	s_amt_sum_30d	S	sum of amounts over the previous 30 days
	s_amt_mean_prev	S	mean prior amount of the sending account
	s_amt_max_prev	S	maximum prior amount
	s_hours_since_last	S	$t_q - \max\{t_x : x \in \mathcal{H}(q), p(x) = p(q)\}$
	s_days_since_first	S	days since the account's first transfer
	s_n_payees_prev	S	distinct prior payees of the sending account
	s_n_dbanks_prev	S	distinct prior destination institutions
	pair_n_prev	S	prior transfers from $p(q)$ to $r(q)$
Receiver cross-bank history	r_n_prev_in	N	$\lvert\{x \in \mathcal{H}(q) : r(x) = r(q)\}\rvert$
	r_n_in_30d	N	the same, restricted to the previous 30 days
	r_n_payers_prev	N	distinct prior payers of the receiving account
	r_n_sendbanks_prev	N	distinct prior sending institutions
	r_days_since_first	N	days since the first incoming transfer

A.2 Dataset statistics

Table 6 places the benchmark among the tabular datasets commonly used to evaluate tabular generators. Its scale is unremarkable, mid-sized by split with a moderate categorical cardinality dominated by the two institution codes, and that is deliberate: the difficulty here comes from a 1.28% positive rate under a temporal split, not from dimensionality. The class imbalance is what distinguishes it from the datasets in the upper block, none of which is a rare-event detection task.

Table 6: Summary statistics of tabular datasets commonly used to benchmark generative models, with ours. Num, Cat, and Feat are the numbers of numerical, categorical, and total features; Card is the total categorical cardinality. Splits of the upper block are random; ours are consecutive time periods, with positive rates of 1.28%, 2.13%, and 1.11%.

Dataset	Train	Valid	Test	Num	Cat	Feat	Card
fb-comments	157,638	19,722	19,720	36	15	51	109
miniboone	83,240	20,811	26,013	50	0	50	0
higgs-small	62,751	15,688	19,610	28	0	28	0
cardio	44,800	14,000	11,200	5	6	11	14
adult	26,048	6,513	16,281	6	8	14	102
house	14,581	3,646	4,557	16	0	16	0
king	13,832	3,458	4,323	17	3	20	12
california	13,209	3,303	4,128	8	0	8	0
buddy	12,053	3,014	3,767	4	5	9	93
churn2	6,400	1,600	2,000	7	4	11	9
gesture	6,318	1,580	1,975	32	0	32	0
wilt	3,096	775	968	5	0	5	0
abalone	2,672	669	836	7	1	8	3
insurance	856	214	268	3	3	6	8
diabetes	491	123	154	8	0	8	0
Ours	63,707	15,427	15,767	18	6	24	113

A.3 Comparison with public financial benchmarks

Table 7 places the benchmark among the public financial datasets of Section 2, along the four properties that decide whether a release can stand in for private records: whether a record spans two institutions, whether the split respects time, what the label means, and whether the paper checks the model ranking.

Table 7: Public financial benchmarks: BAF (Jesus et al., 2022), SynthAML (Jensen et al., 2023), MBD (Mollaev et al., 2025), AMLworld (Altman et al., 2023), AMLgentex (Östman et al., 2025), PaySim (Lopez-Rojas et al., 2016), TabFormer (Padhi et al., 2021), Elliptic (Weber et al., 2019), and DGraph (Huang et al., 2022). Cross-inst.: does one record involve two institutions, with both sides released? Temporal: does the paper specify a time-based split? Rank check: does the paper compare a ranking of models on the released data against real data? ✓ yes, × no, (✓) partial, – not applicable or unspecified. The partial entries are the two simulators, whose two sides are simulated rather than real; SynthAML, which compares six classifiers without a noise scale; and MBD, which reports $\tau = 0.94$ against its own un-anonymized data rather than against a generated surrogate.

Dataset	Source	Cross-inst.	Temporal	Label	Rank check
BAF	CTGAN on applications	×	✓	confirmed fraud	×
SynthAML	copula on real alerts	×	✓	alert disposition	(✓)
MBD	real, anonymized	×	×	product purchase	(✓)
AMLworld	agent-based simulation	(✓)	✓	simulation truth	×
AMLgentex	agent-based simulation	(✓)	×	simulation truth	×
PaySim	agent-based simulation	×	–	none in the paper	×
TabFormer	rule-based simulation	×	–	simulation truth	×
Elliptic	real blockchain graph	–	✓	vendor attribution	–
DGraph	real, masked	×	×	loan default	×
Ours	synthesized transfers	✓	✓	bank suspicion flag	✓

B Detailed Evaluation Metrics

Every metric compares a synthetic training split S with the private training split R it was generated from. Let d index the numeric columns and c the categorical ones. Rank agreement itself is Kendall's τ_b (Kendall, 1938), defined in Section 4.

B.1 Distributional fidelity

KS is the mean over numeric columns of the Kolmogorov–Smirnov statistic $\sup_x |F_d^R(x) - F_d^S(x)|$ between the two empirical distributions. TVD is the mean over categorical columns of the total variation distance $\frac{1}{2}\sum_v |P_c^R(v) - P_c^S(v)|$, with categories absent from one side counted at probability zero. Both are zero for a perfect release and one in the worst case. Dependence error (Corr) is the root-mean-square difference between the Spearman rank-correlation matrices of R and S over the numeric columns, which captures the cross-variable structure that marginal statistics ignore.

B.2 Distinguishability

C2ST is the classifier two-sample test: we label the private rows 0 and the synthetic rows 1, fit a LightGBM classifier (300 trees, learning rate 0.05, 31 leaves) on up to 20,000 rows of each under five-fold stratified cross-validation, and report the out-of-fold ROC-AUC (Lopez-Paz & Oquab, 2017). A value of 0.5 means the two samples are indistinguishable and 1.0 that they are trivially separable. Unlike KS and TVD it is sensitive to joint structure, and unlike TSTR it does not depend on the label.

B.3 Label and downstream utility

The positive rate of S is compared with the private prevalence of 1.28%; label-rate error is the absolute difference in percentage points. TSTR is the test PR-AUC on the private test period of a detector trained and tuned on S alone, reported for CatBoost and for the best of the eleven detectors.

B.4 Privacy

All distances use the same encoding: numeric columns are min-max scaled by the private 1st and 99th percentiles and clipped to $[-1, 2]$, categorical columns are one-hot encoded, and distances are Euclidean in that space. The exact-copy rate is the fraction of synthetic rows whose full feature vector matches a private training row exactly. The membership-inference attack follows the standard formulation (Shokri et al., 2017), in the distance-based form usual for synthetic tabular data. DCR is the median distance from a synthetic row to its nearest private training record; because that number is meaningless in isolation, we report it as a ratio to the same quantity computed for a holdout of 20,000 real transfers drawn from the training period but excluded from the benchmark, so that a ratio of 1.0 means the release sits no closer to the private data than a fresh real sample does. DCR share is the fraction of synthetic rows whose nearest neighbour, among the union of the private training split and an equally sized holdout, falls on the private side; 0.5 indicates no preference for training records. It is 0.650 for SMOTE and 0.547 for TabDDPM, and between 0.28 and 0.53 for the other ten releases, which agrees with the DCR ratios in Table 2. MIA is the ROC-AUC of a nearest-neighbour membership-inference attack that sees only the release: each real record is scored by the negative distance to its closest synthetic row, and the score is used to separate members (private training records) from non-members (the holdout), so 0.5 means the release reveals nothing about membership.

C Generator configurations

Every generator is trained on each temporal split separately, using that split only, and sampled to the size of the split it was trained on; each is run at seeds 0 to 4, and all five releases are published. We use each method's official default configuration wherever one exists. Where the defaults would not finish in reasonable time, we fixed a training budget of about three GPU-hours per split and set the epoch or step count from the method's measured per-epoch cost: GReaT trains 14 epochs instead of the default 100, TabSyn 1,300 VAE and 2,500 diffusion epochs instead of 4,000 and 10,001 on the training split, and TabDiff 1,100 steps instead of 8,000. The budget bounds training only. All generators and detectors of the main experiment ran on one workstation with an NVIDIA GB10 accelerator and twenty CPU cores. Sampling is not counted against the budget, and the GPU was shared with unrelated jobs, so the wall-clock time of a run varied widely: 21 runs exceeded three hours, the longest being GReaT on the training split, which ranged from 3.1 to 25.6 hours across its five seeds. One seed of the eleven generators that use the GPU cost 38.8 GPU-hours, of which TabDiff 12.8, GReaT 9.5, and TabSyn 8.0; SMOTE is the twelfth and runs on CPU in minutes. Table 8 lists the configurations and, in the last column, every deviation we had to make. We report the deviations because each of them is a judgement call that another author might make differently, and three of them materially change what a release looks like: CTAB-GAN trained for 150 rather than 10 epochs, GReaT trained for 14 of the default 100 epochs, and TabPFGen released with its default class balancing (45.8% positives).

Two entries in that column are not budget decisions. TabDDPM has no published default configuration, so we fix the median of the search space of its paper and keep the MLP backbone, and the prior-matched variant of TabPFGen is ours (Appendix C.3).

C.1 GReaT

The reference pipeline passes column names to the language model as the strings `"0"..."23"`. Under those names the model produced no parseable row at all; passing the real feature names instead yields a $> 99.99\%$ valid-row rate (63,706 of 63,707 on the training split). We then set training to 14 epochs on every split, the largest count whose measured cost on an unshared GPU approached the three-hour budget, against the library default of 100; GReaT is thus the generator

Table 8: Generator configurations. Settings not listed are the defaults of the implementation we ran, whose commit is recorded in the released repository. The deviations are explained below the table.

Generator	Configuration	Deviation
SMOTE (SMOTENC)	$k = 5$ neighbours	none
CTGAN	300 epochs, batch 500, lr $2{\cdot}10^{-4}$	none
TVAE	300 epochs, batch 500, lr 10^{-3}	none
CTAB-GAN	150 epochs, batch 512, lr $2{\cdot}10^{-4}$	epochs $10 \rightarrow 150$
CTAB-GAN+	150 epochs, batch 500, lr $2{\cdot}10^{-4}$	none
TabDDPM	MLP $[512, 1024, 1024, 512]$, 30k steps, batch 4096	no published default
GReaT	DistilGPT2, 14 epochs, batch 32	epochs $100 \rightarrow 14$
TabPFGen	1,000 SGLD steps, step 0.01, noise 0.01	one-line code fix
TabPFGen-prior	as TabPFGen, subsampled to the class ratio	ours
TabSyn	1,300 VAE and 2,500 diffusion epochs	epochs reduced
TabDiff	1,100 steps on train, 4,400 on the others	steps $8{,}000 \rightarrow 1{,}100$
FinDiff	MLP, 100 epochs, batch 128, lr 10^{-4}	none

whose nominal schedule we cut the most. It is also among the most expensive to run, at 9.5 of those 38.8 GPU-hours, and at seed 0, on the shared GPU, its training split alone took 5.6 hours of fine-tuning followed by 1.0 hour of sampling.

C.2 Seeding

Four of the reference implementations (CTGAN, TVAE, CTAB-GAN, and CTAB-GAN+) seed only their sampling step, so their training starts from an unrecorded random initialization and a rerun cannot be reproduced. We added seeding of the Python, NumPy, and PyTorch generators at the start of training in each of them, leaving everything else untouched. On a short check (2,000 rows, two epochs, every run in a fresh process), two runs at the same seed produce identical numeric features, categorical features, and labels on the CPU for all four and on the GPU for CTGAN and TVAE, and a different seed changes all three. On the GPU, CTAB-GAN and CTAB-GAN+ reproduce the categorical features and labels but not the numeric ones, of which 72 to 77% differ, by at most 0.003 standard deviations; enabling cuDNN's deterministic mode removes the difference, so it comes from non-deterministic convolution kernels, which our runs left enabled. We did not repeat the check at full training length. TabDDPM, TabPFGen, and SMOTE were already seeded, and our run scripts seed TabSyn, TabDiff, FinDiff, and GReaT in the same way. The seed-0 releases of those four methods were regenerated under the seeded code. Because the CTAB-GAN and CTAB-GAN+ releases were trained on the GPU without deterministic kernels, rerunning them from their seed need not reproduce them bit for bit, which is one reason we publish the releases themselves and not only the code that generated them.

C.3 TabPFGen

With `balance_classes=False` the released implementation raises a type error when the labels live on the GPU; we fixed that single line and changed nothing else. The resulting samples, however, contain no positives at all, because after SGLD the implementation relabels every sample by the TabPFN argmax, which collapses to the majority class at a 1.28% prevalence. We therefore report the default `balance_classes=True` configuration, which yields a 45.8% positive rate, and add the prior-matched variant in the table: it draws twice as many samples and subsamples them to the private class ratio, changing the label distribution without touching the generator.

D Detector search spaces

Every detector receives the same budget on every leaderboard: 20 trials of Optuna's TPE sampler (Akiba et al., 2019) maximizing PR-AUC on the validation split of whichever data it is trained on, then five retrainings of the selected configuration with seeds $0 \ldots 4$, of which we report the mean. Categorical features are one-hot encoded for the scikit-learn models, passed natively to LightGBM (Ke et al., 2017), XGBoost (Chen & Guestrin, 2016), and CatBoost (Prokhorenkova

et al., 2018), whose own implementations we use, and numeric features are standardized for the distance- and gradient-based models (k-NN, logistic regression, MLP). The three gradient-boosting libraries are given 3,000 rounds with early stopping after 100 rounds without improvement on the same validation split; no other detector sees that split during fitting. The other eight detectors are the scikit-learn implementations. One consequence of the -1 placeholder for undefined history (Appendix A) is worth noting: a tree can split it away, but the distance- and gradient-based detectors see it as a numeric value after standardization, which places records with no prior activity at an arbitrary point in feature space and may cost those three detectors some accuracy. We keep the encoding uniform across detectors so that the leaderboard compares detectors rather than imputation schemes. Table 9 gives the search spaces.

Table 9: Detector search spaces. One row per tuned hyperparameter; log marks the ranges sampled logarithmically. Names are those of the implementation.

Detector	Hyperparameter	Range
Naive Bayes	`var_smoothing`	$[10^{-12}, 10^{-2}]$ (log)
Decision tree	`max_depth`	$[3, 20]$
	`min_samples_leaf`	$[1, 200]$ (log)
Logistic regression	C	$[10^{-3}, 10^{2}]$ (log)
	`class_weight`	{none, balanced}
k-NN	`n_neighbors`	$[5, 200]$ (log)
	`weights`	{uniform, distance}
MLP	hidden width	{64, 128, 256}
	depth	$[1, 3]$
	`alpha`	$[10^{-6}, 10^{-2}]$ (log)
	`learning_rate_init`	$[10^{-4}, 10^{-2}]$ (log)
Random forest, extra trees	`n_estimators`	300
	`max_depth`	$[6, 40]$
	`min_samples_leaf`	$[1, 50]$ (log)
	`max_features`	$[0.1, 1.0]$
	`class_weight`	{none, balanced_subsample}
HistGB	`learning_rate`	$[0.01, 0.3]$ (log)
	`max_leaf_nodes`	$[8, 128]$ (log)
	`min_samples_leaf`	$[5, 200]$ (log)
	`l2_regularization`	$[10^{-8}, 10]$ (log)
LightGBM	`learning_rate`	$[0.01, 0.2]$ (log)
	`num_leaves`	$[8, 256]$ (log)
	`min_child_samples`	$[5, 200]$ (log)
	`subsample`	$[0.5, 1.0]$
	`colsample_bytree`	$[0.3, 1.0]$
	`reg_lambda`	$[10^{-8}, 10]$ (log)
XGBoost	`learning_rate`	$[0.01, 0.2]$ (log)
	`max_depth`	$[3, 10]$
	`min_child_weight`	$[10^{-2}, 20]$ (log)
	`subsample`	$[0.5, 1.0]$
	`colsample_bytree`	$[0.3, 1.0]$
	`reg_lambda`	$[10^{-8}, 10]$ (log)
CatBoost	`learning_rate`	$[0.01, 0.2]$ (log)
	depth	$[4, 10]$
	`l2_leaf_reg`	$[10^{-2}, 30]$ (log)
	`bagging_temperature`	$[0, 1]$

D.1 Degenerate releases

A synthetic release can be so imbalanced that a split contains a single class; TVAE produced one positive row per split. We do not discard such a release, because a benchmark user could not dis-

card it either. Instead, a detector that cannot be fitted or tuned falls back to predicting the training prevalence for every row, and a metric that is undefined on a single-class split is recorded as missing rather than imputed. This keeps the affected release on the leaderboard with the score its degeneracy implies.

E External replication on public data

Our protocol is demonstrated on data nobody else can obtain, so we repeated it end to end on a public dataset: the TabReD homecredit-default slice (Rubachev et al., 2025), a real industrial credit-risk task with a time-based split (32,076 / 7,924 / 10,000 rows, 5.0 / 3.3 / 2.3% positive). We reduced it to the 24 most informative features using the training split only, so that the generators face a schema of the same width as ours, and ran the same pipeline at a reduced budget of ten tuning trials and three detector seeds, with every generator run at five seeds as on our benchmark. For seeds 1 to 4 we built only the public-user leaderboard, which is what the question of choosing a release concerns, so TSTR is available for seed 0 alone. Table 10 reports the result.

Four observations follow. First, the noise ceiling is a property of the dataset, not a constant: here the bootstrap ceiling is 0.740 (5th percentile 0.491) and only 18 of 55 detector pairs are separable, against 0.923 and 44 of 55 on our benchmark, so a τ of 0.6 means something different in the two settings, which is the reason we calibrate it at all. Second, two of the seven generators, CTAB-GAN+ and TabPFGen, produce no evaluable release at any of the five seeds: their synthetic test split contains no positive row, so no $S \to S$ score is defined for any detector, and a public user would discover this only after building the leaderboard. Third, the ordering of generators does not transfer. TabDDPM is first here and seventh on our benchmark, SMOTE is first on ours and second here, and TabPFGen is tied fourth on ours and unusable here, so a generator that tops a published comparison cannot be assumed to preserve the leaderboard of the data one actually holds. Fourth, no distributional criterion reliably picks the better release of a generator. Averaged over their releases, marginal fidelity and dependence-structure error are still positively associated with agreement across the five generators whose τ is defined ($\rho = 0.70$ and 0.80, exact $p = 0.23$ and 0.13 with five generators), but among the 47 same-generator pairs they can order, marginal fidelity prefers the release with the higher τ in 51% of them, TVD in 49%, dependence error in 55%, and C2ST in 43%. Label-rate error reaches 71% of the 28 pairs it can order, and under the permutation test of Section 5.3 no criterion separates from chance (smallest $p = 0.11$), although with so few pairs this is weak evidence in either direction; picking by label-rate error costs 0.07 of τ against 0.11 for a random pick. TVAE keeps one positive row per split here, as on our benchmark, so its τ is not interpretable; without it the four distributional criteria are right in 54%, 43%, 59%, and 46% of 37 pairs, again none distinguishable from chance.

One finding is weaker on this dataset. The share of the sum of squares of τ lying within generators is 0.14, against 0.35 on our benchmark, because the releases with a defined τ fall into two tiers far apart: the ten releases of SMOTE and TabDDPM lie between $+0.60$ and $+0.78$, all above the band edge, and the fifteen of the other three at or below $+0.37$. Within the upper tier each generator spans 0.11, less than half the width of the noise band (0.49 to 0.74), while in the lower tier CTAB-GAN spans -0.31 to $+0.20$ and CTGAN -0.02 to $+0.31$, both changing sign across seeds. Here the choice of generator matters more than the choice of release, and the choice of release is still one no standard criterion reliably informs. The seed-0 releases alone would suggest otherwise: at that seed marginal fidelity orders the five generators exactly ($\rho = 1.00$, exact $p = 0.017$), which shows how a comparison built on one release per generator can overstate what a criterion is worth.

F Robustness checks

This appendix collects the checks that the main text refers to: how robust the noise scale is, how far the audit carries beyond the half of the test period it was run on, whether it depends on the detectors, how much power the within-generator test has, and what the privacy metrics can and cannot establish.

Table 10: External replication of the protocol on public data. TabReD homecredit-default (32,076 / 7,924 / 10,000 rows by time, 5.0 / 3.3 / 2.3% positive), same protocol at a reduced budget (10 tuning trials, three detector seeds). Every generator produced five releases; $\tau_{S\to S}$ is the mean over them and [min, max] their range, and the other columns use the seed-0 release. The noise ceiling here is lower than on our benchmark: bootstrap τ ceiling 0.740 (5th percentile 0.491) against 0.923 (0.818), and only 18 of 55 detector pairs are separable. A generator marked undefined has no positive row in the synthetic test split of any of its five releases, so no $S \to S$ score, and hence no public-user ranking, is defined for it, although its training split has positives (Pos. % is the synthetic training split); TVAE keeps one positive row per split, as on our benchmark, so its τ is not interpretable. Bold marks the highest $\tau_{S\to S}$.

Generator	Standard metrics				Label / utility		Leaderboard fidelity		
	KS	TVD	Corr	C2ST	Pos. %	TSTR	$\tau_{S\to R}$	$\tau_{S\to S}$	[min, max]
TabDDPM	0.019	0.014	0.019	0.845	4.92	0.054	+0.53	**+0.74**	[+0.67, +0.78]
SMOTE	0.057	0.062	0.023	0.939	5.02	0.050	+0.82	+0.67	[+0.60, +0.71]
TVAE	0.344	0.317	0.202	1.000	0.00	0.028	+0.34	+0.23	[+0.14, +0.37]
CTGAN	0.264	0.142	0.215	1.000	10.22	0.025	+0.13	+0.18	[−0.02, +0.31]
CTAB-GAN	0.391	0.162	0.215	1.000	4.64	0.025	+0.27	−0.01	[−0.31, +0.20]
CTAB-GAN+	0.249	0.104	0.215	1.000	6.84	0.025	−0.02	undefined	
TabPFGen	0.229	0.038	0.038	1.000	0.03	0.023	+0.29	undefined	

F.1 How robust is the noise scale?

The bootstrap of Section 4 resamples transfers, but transfers from one sending account, or from one week, are not independent: in the test period the busiest account sends 13% of the transfers and the ten busiest send a quarter of them. We therefore repeated it with the sending account (2,133 units) and the calendar week (27 units) as the resampling unit, reading them from an identifier-bearing file that stays inside the institution. The noise ceiling is 0.920 by account and 0.928 by week against 0.923 by transfer, the lower edge of the band is 0.818 by account and 0.855 by week, and all 44 separable pairs remain separable under both; under the stricter week-level edge seven releases instead of eight reach the band, from the same five generators. The scale also degrades gracefully with less private data: from a stratified quarter of the test period (3,941 transfers, 44 flagged) the ceiling is 0.873, the band edge 0.782, and 39 of the 55 pairs are separable.

F.2 How far does the audit carry?

The audit of Section 5.4 selects on one random half of the test period (A) and scores on the other (B). The following checks probe how far that result extends. Splitting into halves of sending accounts or of weeks instead of transfers, selection within a generator recovers 91% and 93% of the gap between a random and the best release, against 92% for halves of transfers. Splitting by time, selecting on July to September (8,099 transfers, 70 flagged) and scoring on October to December (7,668, 105 flagged), it recovers 90% (0.43 to 0.60 against a best of 0.62), and 92% in the opposite direction, so the choice carries over to the later months of the period. With only a quarter of the test period available, averaged over 100 half-splits of that quarter, it recovers 89% against 91% on the same 100 splits of the full period, so an audit needs far fewer flagged transfers than the benchmark contains.

In every split above, half B is never used to select, but every synthetic test split was generated from the whole test period and so from B as well. We removed that overlap for the first of the 200 splits: every generator was retrained at every seed on half A alone (7,883 transfers, 87 flagged) with its original test-split configuration, the new synthetic test splits passed the same publication filter, and each release's detectors were refit from their stored tuning and scored on the new split. Within each generator the release with the highest τ on A is also the best one on B for all twelve generators, so selection recovers the whole gap between a random release ($\tau = 0.36$ on B, averaged over generators) and the best one (0.60); on the same split with the original test splits it recovers 97% (0.46 to 0.69). Agreement is lower on average, plausibly because each synthetic test split now imitates half as many transfers. This is one split rather than 200, and the eleven GPU generators ran on different hardware with the same library versions, so it suggests that, on this split, the audit's success does not depend on the generator having seen B; it does not show how often the audit

succeeds. All of these checks stay within one six-month test period; whether a release chosen today preserves the leaderboard of a later period is untested.

F.3 Does the conclusion depend on the detectors?

Our leaderboard fixes eleven detectors, and the audit selects with all of them. We check how the audit fares on detectors it did not use, and whether adding a twelfth detector changes the results.

The audit is weaker for detectors it did not use. Over all 462 ways of splitting the eleven detectors into six and five, we select on the 15 pairs among the six and score on the 40 pairs that involve any of the other five: within a generator the selected release scores 0.57 against 0.46 for a random one and 0.69 for the best, recovering 47% of the gap and beating a random pick in 97% of the splits, and among all sixty releases it recovers 65%. Scored only on the 10 pairs among the five unused detectors it recovers 28%, beating a random pick in 86% of the splits. Part of the loss is that 15 pairs select less precisely than 55, but the result supports transfer to comparisons that involve a new detector more than to an entirely new set of detectors.

For the second check we added TabM (Gorishniy et al., 2025), a recent parameter-efficient ensemble of MLPs, under the same tuning budget. On the private data it reaches PR-AUC 0.913 and ranks fourth, inside the top tier. Extending the leaderboard to twelve detectors lowers the noise ceiling, since there are more pairs to get wrong: the expected agreement falls from 0.923 to 0.894 and the 5th percentile from 0.818 to 0.788, and 50 of the 66 pairs are separable. For the twelve seed-0 releases, rank agreements computed over twelve detectors track the eleven-detector values at Spearman $\rho = 1.00$ ($S \to S$) and 0.99 ($S \to R$) and differ from them by 0.03 on average, at most 0.06 and 0.08. No pair of releases changes order: adding the detector only breaks one tie between releases under $S \to S$ and creates new ties among three under $S \to R$ (Kendall τ between the two orderings 0.99 and 0.98), so adding one detector leaves the ordering of releases largely intact, unlike scoring on an entirely new set of detectors (previous paragraph).

F.4 How much power does the within-generator test have?

With five releases per generator the permutation test of Section 5.3 can detect only a strong association. We simulated criteria correlated with each release's τ within its generator, on the actual τ values of our releases, and applied the same test: a criterion right in 58% of the pairs is detected at the 5% level in 24% of the simulations, one right in 66% in 74%, and one right in 76% always. The best criterion we observe, label-rate error at 60%, lies in the range the test would usually miss, which is why Section 5.3 reports an absence of evidence rather than an absence of signal.

F.5 What the privacy metrics show

The distance to the closest private record and the nearest-neighbour attack we report are proximity diagnostics, not privacy guarantees; both are known to miss attacks that a formal analysis would catch, and a release can score well on them while still leaking (Ganev & De Cristofaro, 2025; Fang et al., 2024). They are also sensitive to the comparison sample. We computed them against two external holdouts: a uniform random sample of real transfers from the training period, and a matched holdout drawn from outside the account neighbourhood of Section 3.2 and then put through the same case-control rule. The distance ratios agree on the ordering, with SMOTE closest under both (0.42 and 0.49) and every other release near the baseline or beyond it. The attack AUC does not agree: for five of the six releases far from the private data it moves from about 0.45 under the random holdout to between 0.54 and 0.60 under the matched one (FinDiff moves from 0.51 to 0.47), plausibly because widening that neighbourhood reaches a few very high-volume accounts and changes the composition of the sample. An external holdout is therefore not exchangeable with the benchmark rows, and we do not interpret its attack AUC for distant releases. The half-split test reported in Section 5.5 avoids the problem by construction, at the cost of applying only to a generator cheap enough to rerun on half the training period.

G Datasheet for the releases

We follow the structure of Gebru et al. (2021), condensed to the questions that bear on how the artefact should be used.

G.1 Motivation and composition

The releases exist so that fraud-detection methods can be developed and compared outside the institution that holds the transfers, and so that the question this paper asks, whether such a substitution preserves a model ranking, can be studied at all. Each instance is one interbank transfer described by 24 features and a binary suspicion label, with no identifier column. A release contains three datasets, one per temporal split, sampled at the sizes of the private splits they imitate, 63,707, 15,427, and 15,767 rows, less any row deleted for coinciding with a private record. We publish sixty releases: five seeds for each of twelve generators. No instance is an exact copy of a private transfer; rows that coincided exactly with a private record were deleted before publication.

G.2 Collection and processing

The private source is described in Section 3. Labels are suspicion flags raised by participating banks in the course of operations, not adjudicated fraud outcomes. History features are computed only from transfers strictly earlier than the one they describe, and were verified against a brute-force recomputation. Identifiers were replaced by one-to-one mappings held inside the institution before any generator was trained, and amounts were truncated to units. Each generator was then trained on the private split alone and sampled to the same size, at its official default configuration under the budget of Appendix C.

G.3 Uses

The releases are intended for methodological research on fraud detection and on the evaluation of synthetic data. They support the public-user leaderboard and distributional comparisons among releases, and the leaderboards obtained from the five releases of one generator can be averaged, which Section 5.4 shows recovers about half of what an audit against the private data would; reproducing our augmentation experiment additionally requires the private data, which we do not release. They do not support conclusions about the absolute difficulty of fraud detection in deployment, because the benign class is a case-control sample and the labels are suspicion flags; nor about any individual or institution, because no identifier survives. A user who needs a release with known rank fidelity should read the measured value we publish for it and not assume one.

G.4 Distribution and maintenance

The releases are distributed as Parquet files with the schema of Appendix A, under CC BY 4.0, together with a manifest giving each release its measured $\tau_{S\to S}$ and $\tau_{S\to R}$. The evaluation code is distributed under the MIT License. The private transfers are not distributed and will not be. Errata and any further releases will be published at the same location.

H Additional results

This appendix collects the tables the main text points to: the remaining standard criteria per release, the reference leaderboard under every metric we computed, one further use of a release and its sensitivity to the mixing ratio, a feature ablation on the private data, and the per-detector leaderboards behind the rank agreements of Table 2.

Table 12 repeats the reference leaderboard with the columns Table 1 omits, and two of them explain choices made elsewhere in the paper. ROC-AUC is 0.995 or above for eight of the eleven detectors and spans 0.956 to 0.998 in total, so it barely separates them; PR-AUC spans 0.459 to 0.922 over the same detectors and recall at a 0.1% false-positive rate spans 0.119 to 0.826, which is why we rank on PR-AUC and treat ROC-AUC as uninformative here. The seed standard deviation is at most 0.015, for the multilayer perceptron, and is zero for the four detectors whose fitting is deterministic,

Table 11: Remaining standard metrics of each generator. Separable-pair preservation and selection regret are means over that generator's releases; the remaining columns use the seed-0 release, and TSTR here is the best of the eleven detectors, where Table 2 reports CatBoost. The Copies column gives the share of rows identical to a private record, deleted before publication. Across the sixty releases this affected all fifteen SMOTE splits, three TabDDPM runs and two GReaT runs; at seed 0, which this table reports, only SMOTE produced them. The private prevalence is 1.28%.

Generator	Pairs % ↑	Regret ↓	TVD ↓	Dep. ↓	Pos. %	TSTR (best) ↑	Copies %	DCR sh. → .5
SMOTE	99	0.002	0.036	0.024	1.29	0.870	0.89	0.650
TabSyn	92	0.021	0.010	0.007	1.17	0.858	0.00	0.529
TabDiff	90	0.035	0.029	0.063	1.07	0.779	0.00	0.528
TabPFGen	89	0.009	0.654	0.193	45.79	0.837	0.00	0.284
FinDiff	87	0.025	0.327	0.101	1.28	0.771	0.00	0.480
TabPFGen-prior	86	0.009	0.638	0.053	1.28	0.832	0.00	0.306
TabDDPM	76	0.025	0.022	0.061	1.26	0.554	0.00	0.547
TVAE	85	0.006	0.378	0.275	0.00	0.130	0.00	0.493
GReaT	68	0.027	0.081	0.059	0.65	0.655	0.00	0.495
CTGAN	65	0.059	0.085	0.422	10.04	0.014	0.00	0.469
CTAB-GAN	60	0.059	0.226	0.421	2.47	0.073	0.00	0.473
CTAB-GAN+	50	0.080	0.143	0.421	3.37	0.024	0.00	0.488

Table 12: Reference leaderboard, all metrics. Seed mean ± standard deviation over five runs; CI from 1,000 stratified bootstrap resamples of the test period. The columns omitted from Table 1 are included here.

Detector	PR-AUC	95% CI	R@0.1%FPR	R@1%FPR	ROC-AUC
LightGBM	0.922 ± 0.003	[0.890, 0.949]	0.814	0.950	0.998
XGBoost	0.916 ± 0.002	[0.883, 0.944]	0.795	0.944	0.998
Random forest	0.915 ± 0.001	[0.882, 0.944]	0.826	0.944	0.998
CatBoost	0.912 ± 0.002	[0.880, 0.942]	0.771	0.951	0.998
HistGB	0.907 ± 0.002	[0.874, 0.940]	0.782	0.938	0.998
Extra trees	0.882 ± 0.002	[0.840, 0.918]	0.674	0.920	0.995
MLP	0.833 ± 0.015	[0.787, 0.876]	0.584	0.913	0.997
Logistic regression	0.774 ± 0.000	[0.721, 0.828]	0.440	0.891	0.995
Decision tree	0.769 ± 0.000	[0.715, 0.818]	0.527	0.875	0.977
k-NN	0.732 ± 0.000	[0.670, 0.795]	0.423	0.834	0.956
Naive Bayes	0.459 ± 0.000	[0.402, 0.524]	0.119	0.703	0.969

so a leaderboard estimated from five seeds is stable; the width of the bootstrap interval, about ± 0.03 in the top tier and up to ± 0.06 below it, therefore reflects the finite test period rather than retraining, and it is that width the noise ceiling of Section 4 turns into a scale for τ.

Table 13 asks what the five features that only the clearing network can compute are worth, since they are the part of the benchmark no single bank could assemble. Starting from the five transaction features alone (PR-AUC 0.432), the receiver's cross-bank history raises the score to 0.552 and the bank codes with the bank-pair volume raise it to 0.635; together they reach 0.770. The sender's own history, which the sending bank does hold, is worth more than either: it alone reaches 0.904. Once it is present, the network-only features add 0.003 (from 0.913 to 0.916) and nothing at 0.1% false-positive rate. What a receiving account looks like across banks is thus largely redundant with what the sending bank already records about its own customer, in this benchmark, whose rows see a receiver only through the sending accounts it contains.

Table 14 uses a release the other way round, as extra training data for a holder who has few labels rather than as a benchmark. With 5% of the real training labels a detector reaches PR-AUC 0.822, against 0.917 on all of them, and appending a release helps in four cases out of twelve: SMOTE by $+0.057$, TabPFGen by $+0.029$, TabSyn by $+0.020$, and the prior-matched variant by $+0.017$. The other eight hurt, TVAE by -0.198. The effect shrinks as real labels accumulate and is gone by half of them, where no release helps and several still hurt, so augmentation is worth considering only in the scarce-label regime. Which releases help is close to which releases preserve the leaderboard:

Table 13: Feature-group ablation: value of cross-institution information. LightGBM with a fixed configuration, five seeds, private test period. Only the receiver's cross-bank history needs records the sending bank does not hold; bank codes and the sender's own history do not.

Feature groups	# feat.	PR-AUC	R@0.1%FPR	ROC-AUC
Transaction	5	0.432 ± 0.006	0.091	0.990
+ Bank codes, bank-pair volume	8	0.635 ± 0.014	0.257	0.995
+ Receiver cross-bank history	10	0.552 ± 0.009	0.146	0.992
+ Bank, receiver cross-bank	13	0.770 ± 0.009	0.453	0.996
+ Sender history	16	0.904 ± 0.004	0.777	0.997
+ Bank, sender history	19	0.913 ± 0.005	0.777	0.998
All groups	24	0.916 ± 0.005	0.775	0.998

Table 14: Augmenting scarce real labels with a synthetic release. Test PR-AUC on the private test period; first row absolute, the rest change against it. LightGBM and XGBoost at fixed configurations, three stratified subsamples each; $\tau_{S\to S}$ is the seed-mean value of Table 2. At 5% of labels, Spearman ρ between gain and $\tau_{S\to S}$ is 0.74 (exact permutation $p = 0.007$, $n = 12$).

Training data	Fraction of real training labels					$\tau_{S\to S}$
	5%	10%	25%	50%	100%	
Real labels only	0.822	0.859	0.883	0.904	0.917	none
+ SMOTE	+0.057	+0.029	+0.012	−0.002	−0.008	+0.83
+ TabPFGen	+0.029	+0.012	+0.005	−0.004	−0.004	+0.67
+ TabSyn	+0.020	+0.001	−0.020	−0.028	−0.027	+0.72
+ TabPFGen-prior	+0.017	+0.003	−0.010	−0.009	−0.010	+0.53
+ TabDDPM	−0.019	−0.022	−0.021	−0.019	−0.016	+0.43
+ FinDiff	−0.027	−0.042	−0.033	−0.031	−0.018	+0.67
+ TabDiff	−0.041	−0.077	−0.073	−0.064	−0.056	+0.69
+ CTAB-GAN+	−0.063	−0.055	−0.041	−0.025	−0.014	−0.02
+ CTAB-GAN	−0.121	−0.103	−0.044	−0.032	−0.025	+0.13
+ CTGAN	−0.142	−0.086	−0.052	−0.040	−0.030	+0.27
+ GReaT	−0.142	−0.120	−0.106	−0.083	−0.043	+0.30
+ TVAE	−0.198	−0.083	−0.044	−0.016	−0.011	+0.36

the Spearman correlation between the gain at 5% and $\tau_{S\to S}$ is 0.74 ($p = 0.007$). That is a further argument for measuring rank agreement, but not a criterion a public user could apply, because this experiment scores against the private test period.

The augmentation experiment of Table 14 appends the whole synthetic training split, which is about twenty times the size of the 5% real subset. Table 15 repeats it with the synthetic rows capped at one and at four times the number of real rows. The size of the gain depends on the ratio, as one would expect, but the four releases that help do so at every ratio, the six that hurt most hurt at every ratio, and the two that change sign, TabDDPM and FinDiff, stay within 0.008 of zero at 1:1. The finding that the gain tracks leaderboard fidelity is therefore not an artefact of the mixing ratio.

Tables 16 and 17 give the leaderboards behind the rank agreements of Table 2: the test PR-AUC of every detector on every release, ordered so that a rank-preserving release reproduces the ordering of the private row. Reading across a row shows what a benchmark user would conclude from that release alone, and the rows make concrete what a Kendall τ near or below zero looks like. The SMOTE row tracks the private ordering except inside the top tier, where the detectors are not separable anyway, at scores 0.02–0.08 higher for every detector but naive Bayes (range 0.426–0.951, against 0.459–0.922 on the private data). The TabPFGen row is higher still (0.890–0.995 for every detector) because its 45.8% positive rate inflates PR-AUC, which is precisely why absolute scores on a release cannot be compared with private ones. The GAN-based rows are flat: CTAB-GAN+ assigns every detector a score between 0.028 and 0.034, CTAB-GAN between 0.044 and 0.048, and CTGAN between 0.075 and 0.084. On the CTAB-GAN+ release a spread of 0.006 across eleven

Table 15: The releases that help do so at every mixing ratio. Change in test PR-AUC against training on the 5% real subset alone (LightGBM and XGBoost, absolute baseline 0.822), when the synthetic rows appended are capped at one and four times the number of real rows, and when the whole synthetic training split is appended (about twenty times, the setting of Table 14). The four releases that help unrestricted help at every ratio, and the six that hurt most unrestricted hurt at every ratio; TabDDPM and FinDiff, the two that change sign, are within 0.008 of zero at 1:1.

Release	1:1	4:1	Unrestricted
SMOTE	+0.019	+0.047	+0.057
TabPFGen	+0.015	+0.023	+0.029
TabSyn	+0.014	+0.010	+0.020
TabPFGen-prior	+0.001	+0.007	+0.017
TabDDPM	+0.005	+0.005	−0.019
FinDiff	+0.008	−0.002	−0.027
TabDiff	−0.017	−0.025	−0.041
CTAB-GAN+	−0.005	−0.039	−0.063
CTAB-GAN	−0.003	−0.043	−0.121
CTGAN	−0.028	−0.044	−0.142
GReaT	−0.030	−0.075	−0.142
TVAE	−0.022	−0.052	−0.198

Table 16: Per-detector results, trained, tuned, and tested on the release ($S \to S$). Test PR-AUC of the seed-0 release. Columns are ordered by the private leaderboard and rows by $\tau_{S \to S}$; a release preserves the leaderboard when its row orders the columns as the first row does. Bold marks each row's highest score and ties at this precision; rows in which every detector ties carry no bold.

Release	LGBM	XGB	RF	CB	HGB	ET	MLP	LR	DT	kNN	NB
Private data	**0.92**	**0.92**	0.91	0.91	0.91	0.88	0.83	0.77	0.77	0.73	0.46
SMOTE	**0.95**	0.94	0.94	**0.95**	0.94	0.91	0.90	0.83	0.85	0.79	0.43
TabSyn	0.75	0.76	0.75	**0.77**	0.75	0.73	0.68	0.70	0.68	0.63	0.42
TabDiff	0.58	**0.60**	0.59	0.58	0.57	0.57	0.56	0.54	0.52	0.51	0.33
FinDiff	0.49	**0.50**	0.45	0.46	0.46	0.45	0.33	0.35	0.33	0.23	0.05
TabPFGen	0.95	0.98	**0.99**	0.98	**0.99**	0.98	0.98	0.98	0.96	0.95	0.89
TabPFGen-prior	0.54	0.56	**0.74**	0.72	0.72	0.45	0.55	0.32	0.58	0.40	0.11
TabDDPM	0.20	**0.27**	0.16	0.21	0.11	0.07	0.26	0.23	0.18	0.07	0.10
TVAE	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GReaT	0.12	0.13	**0.15**	0.13	0.11	0.13	0.11	0.14	0.10	0.13	0.09
CTGAN	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08
CTAB-GAN	**0.05**	**0.05**	**0.05**	0.04	**0.05**	**0.05**	**0.05**	**0.05**	0.04	**0.05**	**0.05**
CTAB-GAN+	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03

detectors is no wider than the bootstrap interval of a single detector (0.005 to 0.012), and only 10 of the 55 pairs are separable, so the ordering is largely arbitrary. This is the mechanism behind the sign changes in Table 2 and the reason we report a seed range rather than a point estimate. TVAE is the degenerate case discussed in Appendix D: with a single positive row among 15,767 test rows, seven of the eleven detectors score exactly the prevalence floor (6.3×10^{-5}) and the other four stay below 10^{-3}, so the ranking carries no information and the τ reported for TVAE is not interpretable.

Table 17 repeats the exercise under TSTR, where every detector is tested on the same private test period, so the rows are comparable with the private one and with each other. The scores are accordingly much closer to the private ones, and the failure they expose is different: a release can train detectors that score well and still order them wrongly, which is what a $\tau_{S \to R}$ far from one means in Table 2.

Table 18 is the release-level form of the TSTR result of Section 5.3, and it shows why a good TSTR score does not reliably identify a good release. For some generators the TSTR score barely moves while $\tau_{S \to S}$ moves a great deal: the five SMOTE runs span 0.015 in TSTR and 0.18 in τ, and the five prior-matched TabPFGen runs span 0.065 in TSTR and 0.73 in τ. Selecting on TSTR therefore buys little: it returns $\tau = +0.49$ for that generator where the best of its runs reaches $+0.93$, and $+0.24$

Table 17: Per-detector results, trained and tuned on the release, tested on the private test period ($S \to R$). Test PR-AUC of the seed-0 release. Columns are ordered by the private leaderboard and rows by $\tau_{S\to S}$; a release preserves the leaderboard when its row orders the columns as the first row does. Bold marks each row's highest score and ties at this precision; rows in which every detector ties carry no bold.

Release	LGBM	XGB	RF	CB	HGB	ET	MLP	LR	DT	kNN	NB
Private data	**0.92**	**0.92**	0.91	0.91	0.91	0.88	0.83	0.77	0.77	0.73	0.46
SMOTE	**0.87**	**0.87**	0.86	**0.87**	0.86	0.85	0.80	0.72	0.71	0.72	0.44
TabSyn	0.85	0.85	**0.86**	**0.86**	0.85	0.83	0.80	0.76	0.79	0.69	0.46
TabDiff	0.77	**0.78**	0.76	0.77	0.76	0.75	0.73	0.72	0.65	0.62	0.43
FinDiff	0.75	**0.77**	0.74	**0.77**	**0.77**	0.63	0.70	0.69	0.65	0.38	0.23
TabPFGen	0.82	0.81	0.81	0.80	**0.84**	0.75	0.66	0.70	0.63	0.45	0.08
TabPFGen-prior	**0.83**	0.82	0.82	0.82	**0.83**	0.75	0.67	0.51	0.68	0.44	0.08
TabDDPM	0.34	0.51	0.48	0.40	0.19	0.29	**0.55**	0.42	0.31	0.19	0.15
TVAE	0.01	0.03	0.01	0.05	0.01	0.01	0.01	**0.13**	0.01	0.01	0.01
GReaT	0.57	0.50	0.55	0.47	**0.65**	0.51	0.59	0.50	0.60	0.49	0.44
CTGAN	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
CTAB-GAN	0.02	0.01	**0.07**	0.03	0.04	0.01	0.01	0.01	0.03	0.02	0.01
CTAB-GAN+	0.01	0.01	0.01	0.01	**0.02**	0.01	0.01	**0.02**	**0.02**	0.01	**0.02**

Table 18: TSTR does not reliably tell a good run of a generator from a bad one. Runs of one generator differ in $\tau_{S\to S}$ far more than in TSTR. Across the 112 pairs of runs of the same generator, the run with the higher TSTR also has the higher $\tau_{S\to S}$ in 54% of cases (62% of the 76 pairs among non-collapsed generators). The last two columns compare the run TSTR would select with the best run.

Generator	Runs	Range over runs		$\tau_{S\to S}$ of the run picked by	
		TSTR	$\tau_{S\to S}$	TSTR	Oracle
SMOTE	5	0.870 to 0.885	+0.71 to +0.89	+0.89	+0.89
TabPFGen	5	0.776 to 0.846	+0.45 to +0.89	+0.60	+0.89
TabPFGen−prior	5	0.807 to 0.872	+0.20 to +0.93	+0.49	+0.93
TabDDPM	5	0.396 to 0.813	+0.05 to +0.71	+0.56	+0.71
GReaT	5	0.470 to 0.655	−0.05 to +0.56	+0.56	+0.56
TVAE	5	0.022 to 0.047	+0.08 to +0.51	+0.37	+0.51
CTAB−GAN	5	0.008 to 0.027	−0.02 to +0.38	+0.16	+0.38
CTGAN	5	0.006 to 0.121	−0.27 to +0.67	+0.24	+0.67
CTAB−GAN+	5	0.011 to 0.030	−0.31 to +0.20	−0.02	+0.20
TabSyn	5	0.743 to 0.856	+0.53 to +0.89	+0.71	+0.89
TabDiff	5	0.755 to 0.796	+0.35 to +0.89	+0.67	+0.89
FinDiff	5	0.697 to 0.798	+0.60 to +0.78	+0.75	+0.78

against $+0.67$ for CTGAN. Over the 112 pairs of runs of one generator the run with the higher TSTR is the one with the higher τ in 54% of cases, and in 62% of the 76 pairs that exclude the collapsed generators, neither distinguishable from a coin under the permutation test of Section 5.3 ($p = 0.52$ and 0.14).