Title

Toward an Auto-Synthesizer: Stress-Aware Selection and Recommendation of Deep Tabular Generative Models

ISBN

9798190929652

Author

Son, Hochan

Publication Date

2026-09-10

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Toward an Auto-Synthesizer: Stress-Aware Selection and Recommendation of Deep Tabular Generative Models

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Applied Statistics & Data Science

by

Hochan Son

2026

ABSTRACT OF THE THESIS

Toward an Auto-Synthesizer: Stress-Aware Selection and Recommendation of Deep Tabular Generative Models

by

Hochan Son
Master of Applied Statistics & Data Science
University of California, Los Angeles, 2026
Professor Guang Cheng, Chair

Deep generative models for tabular data — GANs, variational autoencoders, diffusion models, and LLM-based generators — exhibit highly non-uniform behavior across datasets: the best-performing family depends on distributional stressors such as long-tailed marginals, high-cardinality categoricals, Zipfian imbalance, and small-sample regimes. While automated machine learning resolved the analogous selection problem for supervised learning a decade ago, tabular synthesis still lacks an "Auto-Synthesizer": its libraries standardize and benchmark generators but leave the question of which model to use to the practitioner, for whom evaluating every candidate — training, sampling, and scoring each under competing objectives of fidelity, privacy, and utility — is prohibitively expensive. This thesis studies intent-conditioned tabular synthesis selection and answers it with a two-system framework. Table-Synthesizers, the generation and measurement substrate, unifies 19 published generators behind a single interface and manufactures the empirical ground truth against which any selector must be judged. SYNTHONY, the selection layer, introduces stress profiling — a synthesis-specific, interpretable meta-feature representation quantifying dataset diffi-

culty along four failure-mode-aligned dimensions — and matches stress profiles against a measurement-grounded, Bayesian-calibrated capability registry to rank candidates without training a single generator at inference time. Across a benchmark of 7 datasets, 10 synthesizers, and 3 user intents, stress-based meta-features prove highly predictive of synthesizer performance: an oracle-informed nearest-neighbor reference attains 0.578 held-out Spearman rank correlation against intent-specific oracles, which this thesis independently reproduces (0.581) from the recovered evaluation matrix. The deployable zero-shot selector attains a reported 0.573 — within 0.005 of that reference at nine held-out pairs, a gap far smaller than the coarse granularity the sample size supports — while training no generator at inference time, remaining per-dimension inspectable and extensible to newly registered generators without benchmark reruns; independently reproducing that figure awaits recovery of the selector's calibrated parameters from the original pipeline. Ablations show dataset stress profiling is the load-bearing component, and that freeing all registry parameters overfits at current ground-truth scale — motivating the framework's re-runnable calibration pipeline. The reproduced selection signal is robust to the definition of the third intent, holding (Spearman 0.52–0.58) whether downstream quality is measured by latency or a train-on-synthetic–test-on-real utility oracle. The thesis closes by specifying, at task granularity, the remaining distance to a closed-loop Auto-Synthesizer that executes, evaluates, and learns from its own recommendations.

The thesis of Hochan Son is approved.

Michael Tsiang

Fredric Schoenberg

Guang Cheng, Committee Chair

University of California, Los Angeles

2026

I would like to express my deepest gratitude to my family for their unwavering love, support, and dedication throughout this journey. To my wife, Chahyun Hwang, and my son, Dali Son, thank you for being my constant source of inspiration and strength.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am profoundly grateful to my academic advisor, Prof. Guang Cheng, for your invaluable guidance, mentorship, and continued support. I would also like to thank the members of my committee, Prof. Mike Tsiang and Prof. Rick Schoenberg, for their time and insightful feedback. It has been a distinct honor to collaborate with my colleagues, Xiaofeng Lin and Jason Ni, as well as all my fellow researchers at the Trustworthy AI Lab; your camaraderie and intellectual contributions have made this experience truly rewarding. Finally, I wish to extend my sincere thanks to Laurie, MASDS advisor, and Chie Ryu, Chief Administrative Officer at the Department of Statistics, for their exceptional administrative support and guidance throughout my studies.

CHAPTER 1

Introduction

1.1 What is Synthetic Data Generation?

Synthetic data generation is the construction of artificial records that preserve the statistical structure of a real dataset — its marginal distributions, inter-feature dependencies, and predictive relationships — without reproducing the real records themselves [JSH22]. The practice today spans modalities — text and code synthesis via large language models form their own active line [NDT25], with their own evaluation challenges [CIL25] — but this thesis addresses the tabular case, where the stakes of consequential decision-making and regulation concentrate. Formally: given a private dataset D drawn from an unknown distribution $p(\mathbf{x})$, a generative model is fit to approximate p, and a synthetic dataset D' is sampled from the fitted model. The value proposition is that D' can stand in for D in analysis, model training, testing, and sharing, while the sensitive originals never leave custody.

The idea is older than the deep learning era that popularized it. It originates in *statistical disclosure control*: Rubin [Rub93] proposed, in the Journal of Official Statistics, releasing fully synthetic public-use microdata generated by multiple imputation — treating every value as "missing" and imputing all of it from a model fit on the confidential data — so that no original record is ever released. Little [Lit93] proposed the partially-synthetic variant (imputing only the sensitive variables), and a decade later Raghunathan, Reiter and Rubin [RRR03] and Reiter [Rei03] gave the approach the rigorous inferential foundation that let national statistical agencies actually deploy it. Synthetic data is thus not a by-product of

generative AI; generative AI is the latest — and most capable — instrument applied to a three-decade-old statistical problem.

It is worth distinguishing synthesis from its nearest neighbor, anonymization. Anonymization transforms real records (masking, generalization, suppression) and has repeatedly proven fragile to linkage and re-identification attacks; synthesis generates *new* records from a learned distribution, shifting the privacy question from "can this row be linked back?" to "how much does the model memorize?" [SOT22] — a question this thesis returns to repeatedly, since memorization risk is precisely what makes model *selection* consequential on small datasets. (Adversarial quantification of that risk — membership-inference and related attacks — is out of scope for this thesis, which uses the benchmark's distance-based privacy proxy throughout.)

1.2 What is the Purpose of Tabular Synthetic Data Generation?

Tabular data — rows of mixed continuous and categorical features — is the dominant data format of consequential decision-making: electronic health records, financial transactions, insurance claims, administrative registers, advertising and customer-relationship data. Recent surveys — general [BTS24] and tabular-specific [SWD25] — justify the tabular focus on exactly these grounds: widespread use in critical decisions, complex feature dependencies, and pressing privacy demands in regulated industries. The main purposes cluster into five:

1. **Privacy-compliant data sharing in regulated sectors.** Healthcare (HIPAA), finance (GLBA, PCI-DSS), government statistics, and any cross-border data flow under GDPR/CCPA. Synthetic counterparts let hospitals share EHR-like data for research (training predictive diagnostic models without exposing patients — [YZN24]), let statistical agencies publish census-like microdata (the original Rubin use case), and let advertisers and platforms share audience-level data without individual-level exposure.

2. **Data augmentation and class-imbalance repair.** Fraud detection, rare-disease cohorts, and manufacturing defect data are all label-starved in exactly the class that matters; conditional synthesis oversamples the minority regime (the lineage running from SMOTE, [CBH02], to conditional deep generators; surveyed for the rare-event case by [GZW25] — whose framing of long-tail generation failure is the survey-level counterpart of this thesis's Zipfian stress dimension, §3.3.1).

3. **Model stress-testing and scenario generation.** Financial institutions generate realistic transaction data to stress-test risk models and fraud systems against regimes absent from history — e.g., synthetic transaction series for fraud-model stress testing, retail analytics, and advertising "data clean rooms" for privacy-safe cross-party collaboration (use-case coverage: [BTS24]; [SWD25]).

4. **Software development and testing.** Realistic-but-fake data for development, staging, and QA environments where production data is contractually or legally off-limits.

5. **AI training-data scarcity.** As a supply of training data for machine learning where real data is scarce, siloed, or too expensive to label — increasingly relevant as data-hungry models meet data-protection law.

1.3 What is the Current Revolution in Tabular Synthetic Data Generation?

The field's history reads as four waves, each changing what "a generative model of a table" means:

Wave 1 — statistical synthesis (1993-2015). Multiple-imputation synthesis ([Rub93]), sequential CART synthesis ([Rei05]; `synthpop`, [NRD16]), Bayesian networks [ZCP14], and Gaussian copulas ([PWV16]; [Skl59]). Interpretable, fast, and — as this thesis's benchmark repeatedly finds — still highly competitive on moderate-sized data.

Wave 2 — deep generative adaptation (2017-2021). GANs and VAEs adapted to the pathologies of tables: CTGAN and TVAE [XSC19] introduced mode-specific normalization and training-by-sampling to survive multimodal numerics and imbalanced categoricals; PATE-GAN ([JYS19]) and PATECTGAN grafted differential privacy onto adversarial training.

Wave 3 — diffusion and latent-space generation (2022-2024). TabDDPM [KBR23] brought cascaded Gaussian/multinomial diffusion to mixed types; TabSyn [ZZS24] moved diffusion into a VAE-crafted latent space; TabDiff ([SXH25]) added per-column learnable noise schedules. Diffusion models currently define state-of-the-art fidelity on large mixed-type tables (perspective: [Zhu24]).

Wave 4 — language models and foundation models (2022-). GReaT [BSL23] serializes rows to text and fine-tunes a pretrained LLM; REaLTabFormer [SD23] extends the approach to relational pairs; and TabPFN-based generators ([HME23]; TabPFGen, [MDS24]) generate through a pretrained prior with no per-dataset gradient training at all. (This wave also brings new memorization surfaces of its own — noted here for completeness, but out of scope for this thesis.) In parallel, a differential-privacy line matured independently (AIM, [MMS22]).

And the accumulation has not stopped: reinforcement-learning-based synthesis is emerging as a fifth line — ReTabSyn [LKL26] optimizes realism as a reward signal — a reminder that any selection framework must expect its candidate pool to keep growing (a design requirement this thesis takes seriously; see the registry-extension property of §3.4.2).

The "revolution," then, is not any single architecture but the *accumulation* of them: a practitioner in 2026 faces at least five architecturally incommensurable families, each with regimes of genuine superiority and documented failure modes. Recent comparative surveys [BTS24, SWD25] confirm what this thesis's benchmark also finds: rankings are unstable across datasets and evaluation objectives, and no family dominates. The revolution created the selection problem.

1.4 What are the Tabular Synthetic Data Models?

(Condensed inventory; the full mechanism-level descriptions live in Methodology §3.2.2. This thesis's infrastructure layer, Table-Synthesizers, wraps 19 of these behind one interface; 15 are registered for selection; 10 constitute the evaluated benchmark pool.)

Table 1.1: Model families of the implemented pool with representative references.

Family	Models (this thesis's pool)	Representative reference
Baseline	Identity	—
Statistical / resampling	CART, SMOTE, GaussianCopula	[Rei05]; [CBH02]; [PWV16]
Probabilistic graphical	BayesianNetwork	[ZCP14]
GAN	CTGAN, PATECTGAN	[XSC19]; [JYS19]
VAE	TVAE, LTM-VAE	[XSC19]; (internal)
Diffusion	TabDDPM, TabDiff	[KBR23]; [SXH25]
Latent hybrid (VAE+diffusion)	TabSyn, AutoDiff	[ZZS24]; [SLH23]
Normalizing flow	NFlow	[PNR21]
Tree-based adversarial	ARF	[WBK23]
LLM-based	GReaT	[BSL23]
Foundation-model in-context	TabPFGen	[MDS24]
DP mechanisms	AIM, DPCART	[MMS22]

Implementation-status disclaimer. The table above lists only the models actually implemented in Table-Synthesizers (and, per the 19/15/10 note, not all of those are selectable or benchmarked). Every other generator named in this thesis — REaLTabFormer (§1.3) and ReTabSyn (§1.3, Chapter 6) — is cited **for reference and context only**: neither is implemented, wrapped, evaluated, or selectable in either Table-Synthesizers or Synthony as of this writing. No experimental claim in this thesis rests on either.

1.5 Why Build a Training and Evaluation Framework — Toward an Auto-Synthesizer

The central claim of this thesis, inherited from the SYNTHONY paper [SLN26]: **the bottleneck in tabular synthesis is no longer *access* to generative models but the *selection* of them.** Figure 1.1 states the situation the argument responds to; four observations then compose it:

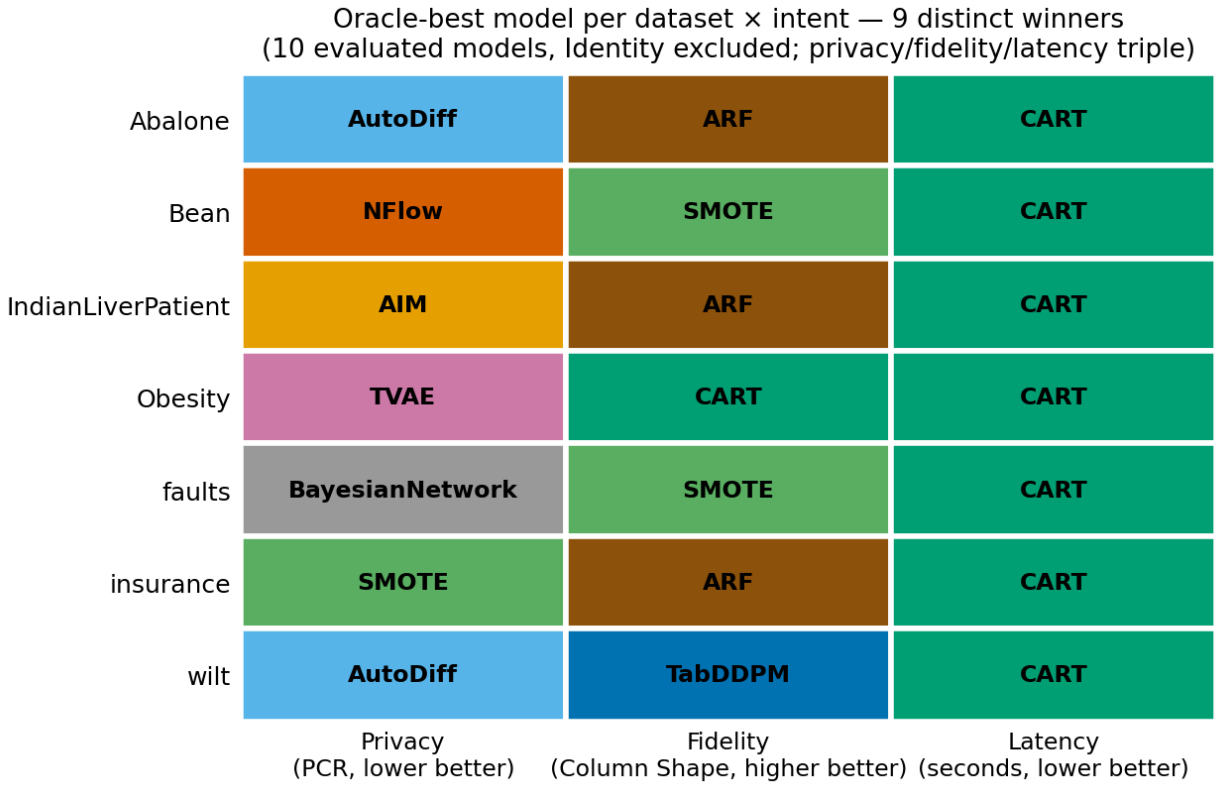

Figure 1.1: Oracle-best model per dataset and intent, computed from the benchmark evaluation matrix (ten evaluated models, Identity excluded; privacy/fidelity/latency triple). Nine distinct winners across the 21 cells: no single model dominates. The latency column is uniformly CART — it trains in near-zero time — whereas the privacy column is maximally diverse, the pattern that motivates intent-conditioned selection.

1. **Non-uniform behavior is systematic, not noise.** The best-performing family depends strongly on measurable distributional stressors — long-tailed marginals, high-cardinality categoricals, Zipfian imbalance, small-sample regimes — each of which names a documented failure mechanism (mode collapse: [MGN18]; tail-category erasure: [New05]; memorization/leakage: [SOT22]). Figure 1.1 makes this concrete: across seven datasets and three intents, nine distinct models are oracle-best, with none dominating. Selection is therefore *predictable* from dataset properties.

2. **Selection is risky when uninformed.** An untuned complex model (e.g., a GAN destabilized by mode collapse) can underperform a simple, well-tuned baseline — the practitioner's intuition that "newer is better" is exactly wrong often enough to be dangerous.

3. **Existing tooling standardizes but does not decide.** SDV [PWV16] and SynthCity [QCS23] unify APIs and benchmarking; tuning-integrated tools (SynMeter, [DL24]; STNG, [RAR24]) close the hyperparameter loop — yet all leave *which model* to the user. Supervised learning has had an answer to the analogous problem for a decade (auto-sklearn, [FKE15]; AutoGluon, [EMS20]); synthesis lacks an "Auto-Synthesizer" paradigm. The selection question has been posed before in a narrower form: Cheng et al. [CWP22] studied generative-model selection for synthetic training data in the specific setting of downstream fraud-detection utility — this thesis generalizes that question from one downstream task to arbitrary datasets and intents, and from post-hoc comparison to predictive, profile-driven selection.

4. **The evaluation cost structure forbids brute force.** The oracle answer — train every candidate, evaluate all of them, pick the winner — costs a full factorial sweep per dataset, and the "correct" answer additionally depends on the user's intent (privacy vs. fidelity vs. utility), which reweights the metric vector per deployment.

The response is the two-system framework this thesis builds and the title names. **Table-**

Synthesizers supplies the *training and evaluation* half: 19 generators behind one interface, measured by one code path, producing the empirical ground truth that any honest selector must be calibrated against. **SYNTHONY** supplies the *recommendation* half: stress profiling as a synthesis-specific meta-feature representation, a calibrated capability registry, and intent-conditioned scoring that approximates the oracle at zero generators trained per query. Together they are a step *toward* an Auto-Synthesizer — the qualifier is deliberate, and §1.6 states what remains.

1.6 What is the Future of the Synthetic Data Generation Framework?

The framework in this thesis selects, and its infrastructure trains and evaluates — but the loop between them is still closed by hand: benchmarks are run offline, the registry is calibrated in batch, and a recommendation, once issued, does not observe its own consequences. The natural end state is *agentic*: a selector that recommends, executes its recommendation (training under subprocess isolation with real cancellation), benchmarks the result, iterates on quality shortfall, and deposits every outcome into a store that continuously recalibrates the registry it recommends from — converting selection from a one-shot prediction into a bounded, self-improving search. Beyond the single loop, the same recommendation machinery generalizes to *multi-agent* settings, where a selection service advises other autonomous agents on their own decision points (which tool, which workflow, which model) — a different decision domain sharing the same capability-matching core. Two cautions accompany that future: closed loops that train on their own synthetic output risk recursive degradation — the model-collapse phenomenon ([SSZ24]) — and agentic tool-using systems introduce failure modes of their own, from silent mid-trace errors to adversarial exploitation of tool-call provenance, so an agentic Auto-Synthesizer must be evaluated as an agent, not just as a ranker. At the time of the underlying paper's publication, none of this was implemented;

the design, its preconditions, and the parts since prototyped are treated in the Future Work chapter (Chapter 6), which this introduction only gestures at. The thesis's contribution is the foundation that makes that future work well-posed: the components are built, measured, and calibrated — what remains is the loop.

1.7 Contributions and Thesis Organization

This thesis makes five contributions:

1. **Problem formulation and evaluation protocol.** We pose *intent-conditioned tabular synthesis selection* as an instance of the classical Algorithm Selection Problem [Ric76], formalize its oracle, and establish an evaluation protocol grounded in the algorithm-selection tradition — Top-K accuracy, Spearman rank correlation, and NDCG against intent-specific oracles — rather than the leaderboard tradition of synthesis benchmarking.

2. **Stress profiling as a synthesis-specific meta-feature representation.** Four interpretable dataset-difficulty dimensions, each tied to a documented generative failure mechanism, and empirical evidence that they carry learnable selection signal (a meta-feature kNN reaches 0.578 held-out Spearman against the oracle).

3. **A two-system, contract-coupled framework.** Table-Synthesizers — 19 published generators unified behind one interface and one evaluation code path — as the ground-truth-manufacturing substrate, and SYNTHONY — a measurement-grounded, Bayesian-calibrated capability registry with intent-conditioned scoring — as a zero-shot selector; the two coupled by a thin, auditable file contract so that the selector trains no generator at inference time and extends to new generators without benchmark reruns.

4. **Cross-family empirical analysis with a transparent validity audit.** A benchmark across 7 datasets, 10 synthesizers, and 3 intents showing that the calibrated selec-

tor nearly matches an oracle-informed reference (0.573 vs. 0.578 Spearman); ablations locating the load-bearing component (stress profiling) and an honest negative result (intent scaling overfits at current ground-truth scale); a consolidated threats-to-validity disposition (Experiments §5.10) that resolves, rather than inherits, the reporting issues identified in the underlying publication; and a split-sensitivity analysis (§5.12) that re-establishes the finding under leave-one-dataset-out cross-validation and across all 35 possible dataset-level splits.

5. **A task-granular roadmap toward the closed loop.** A future-work program (Chapter 6) specifying, with per-package tasks and validation gates, the remaining distance to an Auto-Synthesizer that executes, evaluates, and learns from its own recommendations.

Relationship to the prior publication. The empirical numbers in Chapters 3 and 5 — the benchmark tables, the SF-only/Joint comparison, the ablation and meta-predictor results — are transcribed from the underlying SYNTHONY paper [SLN26] (ICLR 2026 DeLTa Workshop) and are labeled as such throughout; this thesis makes no new claim about the published system beyond the clearly-labeled reproduction and robustness analyses it runs itself (§5.11, §5.12), which are reported as its own results and never substituted for the published ones.

What is original to this thesis is: (a) the consolidated threats-to-validity audit (Experiments §5.10) that *resolves*, rather than merely inherits, the paper's own reporting issues; (b) a deeper methodological exposition reconciling the published method description against the shipped implementation, including discrepancies the paper does not disclose (Methodology §3.3.2; §5.10 item I); (c) the live, end-to-end case-study captures of Appendix A, which the paper does not contain; (d) the reproduction of the oracle-derived rows from the recovered evaluation matrix, which settles the third-intent question in favour of latency (§5.11); (e) a leave-one-dataset-out cross-validation and an exhaustive enumeration of all 35 dataset-level

splits (§5.12), establishing that the published single-split result is not an artifact of its split; and (f) the task-granular future-work roadmap of Chapter 6.

This draft additionally closes three items that were open as of the most recent internal audit: the `faults` dataset's feature-count range is corrected to 6–34 by directly verifying column counts across every benchmark artifact (Methodology §3.2.4; §5.10 item D); the apparent 54-versus-78 parameter inconsistency and the Table-1-versus-Table-4 number mismatch are resolved by direct verification against the published PDF — the published paper contains only the 78-parameter configuration and its two tables agree exactly, both apparent inconsistencies tracing to superseded internal drafts (§5.10 items B–C); and a gap not previously stated in the paper or its critique — that the calibrated scale factors behind the headline SF-only result (0.573) are absent from the committed repository — is now disclosed explicitly as validity-audit item J. It also corrects three references carried over from the paper's reference list that do not match the publisher record, and re-points the two claims the thesis had rested on them (§5.10 item K).

Where a claim in this thesis differs from a claim in the paper — e.g., the third-intent identification of §5.11, corrected here to latency — the difference is stated explicitly and traced to its evidence, not silently substituted.

Organization. Chapter 2 positions the work against generative models, synthesis libraries, and algorithm selection. Chapter 3 presents the methodology: the two-system architecture's conceptual pipeline, stress profiling, capability scoring, and calibration. Chapter 4 details the system architecture and the data contract between the two systems. Chapter 5 reports the empirical evaluation, ablations, and validity audit. Chapter 6 specifies future work, and Chapter 7 concludes.

CHAPTER 2

Related Work

2.1 Generative Models for Tabular Data

Tabular synthesis predates deep learning by two decades. The statistical-disclosure-control lineage — fully synthetic microdata via multiple imputation [Rub93], partially synthetic variants [Lit93], and the inferential foundations that made them deployable at statistical agencies [RRR03, Rei03] — established both the problem framing and the first generation of methods: sequential CART synthesis [Rei05, NRD16], Bayesian networks [ZCP14], and copula models coupling per-column marginals through Sklar's theorem [Skl59, PWV16].

The deep generative era adapted general-purpose architectures to tabular pathologies. CTGAN and TVAE [XSC19] confronted multimodal numerics and imbalanced categoricals with mode-specific normalization and training-by-sampling; PATE-GAN [JYS19] composed adversarial training with differential privacy. Diffusion models [HJA20] arrived via TabDDPM's cascaded Gaussian/multinomial processes [KBR23], latent- space variants that decouple mixed-type encoding from score-based generation (TabSyn, [ZZS24]; AutoDiff, [SLH23]), per-column noise scheduling (TabDiff, [SXH25]), and normalizing-flow alternatives with exact likelihood [PNR21]. Tree-based adversarial methods (ARF; [WBK23]) achieve competitive density estimation with no neural network at all. Language-model-based generation serializes rows to text (GReaT, [BSL23]; REaLTabFormer, [SD23]), foundation-model approaches sample through pretrained priors without per-dataset training (TabPFN, [HME23]; TabPFGen, [MDS24]), and reinforcement-learning formulations are emerging (ReTabSyn,

[LKL26]). A separate differential-privacy line matured around workload-adaptive marginal measurement (AIM, [MMS22]), and SMOTE-style interpolation [CBH02] persists as a fast resampling baseline. Diffusion-centric and general overviews of this landscape appear in [Zhu24], [BTS24], and — tabular-specifically — [SWD25]; rare-event generation, the regime behind two of this thesis's four stress dimensions, is surveyed by [GZW25].

Positioning. This thesis contributes no new generator. Its claim is that the architectural diversity above *is itself the problem*: each family has regimes of genuine superiority and documented failure mechanisms — mode collapse under adversarial training [MGN18], tail-category erasure under power-law categoricals [New05], memorization in low-sample regimes [SOT22] — and no family dominates. Where a new-generator paper argues "our architecture wins," this thesis asks the orthogonal question: *given that no architecture wins everywhere, which one should this dataset and this intent get?*

2.2 Synthesis Libraries, Benchmarking, and Evaluation

A second body of work standardizes rather than generates. SDV [PWV16] established the unified-API pattern for tabular synthesis; SynthCity [QCS23] extended it across modalities with integrated benchmarking. Evaluation-focused efforts formalize what "good synthetic data" means: SynMeter [DL24] couples principled fidelity/privacy/utility assessment with tuning; STNG [RAR24] integrates hyperparameter search into the evaluation loop; domain-specific evaluation frameworks exist for health records [YZN24] and beyond tables entirely [CIL25]. The TSTR paradigm — train on synthetic, test on real [EHR17] — remains the standard utility instrument, and this thesis adopts it unchanged.

Positioning. The infrastructure half of this thesis (Table-Synthesizers, Methodology §3.2) belongs squarely to this family: 19 generators behind one `fit`/`sample` interface, measured by one code path. The difference is what sits on top. Every system above leaves the *selection* decision to the user — SDV and SynthCity offer catalogs, SynMeter and STNG

optimize a chosen model's hyperparameters, but none answers "which model?" This thesis treats the standardized library not as an end product but as the ground-truth factory for an automated selector: the benchmark output that other work presents as leaderboards becomes, here, calibration signal (Methodology §3.2.4). To our knowledge, no prior synthesis library closes this loop.

2.3 Meta-Learning and Algorithm Selection

Choosing an algorithm from instance features is a formalized problem with a fifty-year history [Ric76], made unavoidable in principle by no-free-lunch results [WM97], and made practical in supervised learning by AutoML: meta-feature-driven warm-starting in auto-sklearn [FKE15], robust ensembled defaults in AutoGluon [EMS20], and a mature meta-feature literature [Van19, RGS22] supported by shared experiment infrastructure [VRB13]. Model-based hyperparameter optimization [BBB11] — which this thesis repurposes for registry calibration — and bandit-based exploration [RVK18] — which Future Work §6.1 targets — come from the same tradition.

For *synthesis* selection specifically, the cupboard is nearly bare. Recent surveys consistently identify it as open: synthesizer rankings shift with compute budget and offer no predictive mechanism [KRF25], practitioners must manually interpret benchmark results [DWP25], and the best available guidance is qualitative [DGP25]. The closest prior work is that of Cheng et al. [CWP22], who compared generative models for synthetic training data under a single downstream objective (fraud-detection utility) — post-hoc comparison in one fixed setting, rather than predictive selection across datasets and intents.

Positioning. This thesis ports the Rice framework to synthesis with two adaptations the supervised setting does not need. First, *synthesis-specific meta-features*: generic AutoML meta-features predict supervised learnability, but synthesizer failure is driven by distributional stressors — hence stress profiling (Methodology §3.3.1), four interpretable dimensions

each tied to a documented failure mechanism. Second, *intent conditioning*: in supervised learning the objective is fixed by the task, whereas a synthesis "best model" is undefined until the user weights fidelity against privacy against utility — hence intent-conditioned scale factors (§3.3.3). The evaluation design follows the algorithm-selection tradition (oracle regret, rank correlation) rather than the leaderboard tradition of §2.2, which is what distinguishes a *selector* from a *benchmark*.

2.4 Scope Boundaries with Adjacent Lines

Three adjacent literatures are deliberately out of scope, cited only where their existence shapes a design decision. (i) *Adversarial privacy evaluation* — membership inference and related attacks — is excluded per the thesis scope; the privacy oracle uses the benchmark's distance-based proxy throughout (Methodology §3.2.4), with attack-grounded evaluation deferred to Future Work §6.5. (ii) *LLM text/code synthesis* [NDT25] and its evaluation [CIL25] proceed on a parallel track; this thesis is tabular-only. (iii) *Recursive-training dynamics* — model collapse when generators train on generated data [SSZ24] — becomes relevant only for the closed-loop future system (Chapter 6), not for the one-shot selection studied here.

Table 2.1: Positioning against adjacent lines: which systems generate, standardize, evaluate, tune, and select.

System / line	Generates	Standardizes	Evaluates	Tunes	**Selects**
Individual generators (§2.1)	yes	—	—	—	—
SDV / SynthCity	yes	yes	yes	—	—
SynMeter / STNG	—	yes	yes	yes	—
Supervised AutoML (§2.3)	—	yes	yes	yes	yes *(supervised only)*
Cheng et al. [CWP22]	—	—	yes	—	partial *(one task, post-hoc)*
This thesis (Table-Synthesizers + SYNTHONY)	yes	yes	yes	partial *(calibration)*	yes *(intent-conditioned, zero-shot)*

CHAPTER 3

Methodology

3.1 Overview and Theoretical Framing

3.1.1 The problem as algorithm selection

Our methodology is an instantiation of the Algorithm Selection Problem [Ric76], which formalizes selection as four spaces and two mappings: a problem space $\mathcal{P}$ (here: tabular datasets paired with user intents), a feature space $\mathcal{F}$ (here: stress profiles, §3.3.1), an algorithm space $\mathcal{A}$ (here: the generator pool $\mathcal{G}$ provided by Table-Synthesizers), and a performance space $\mathcal{Y}$ (here: intent-conditioned metric vectors). The selector is the mapping $S : \mathcal{F} \to \mathcal{A}$ that maximizes performance without exhaustively evaluating $\mathcal{A}$. The no-free-lunch results for search and optimization [WM97] supply the motivation: no single generator dominates across all data distributions, so *any* fixed choice is suboptimal on some inputs, and empirically the best model family varies with measurable dataset properties (§4 of the paper). Independent benchmarks reach a compatible conclusion by a different route: [KRF25] finds that synthesizer rankings are not budget-invariant — diffusion models lead in general, but the lead is not significant once tuning and training are held to a common compute budget — so even a fixed ranking is contingent on the conditions that produced it.

What distinguishes generative-model selection from the supervised AutoML setting (auto-sklearn, [FKE15]; AutoGluon, [EMS20]) is the cost structure of the performance signal. In supervised learning, evaluating a candidate costs one cross-validation run and the label is unambiguous. In synthesis, evaluating a candidate requires *training the generator,*

sampling from it, and scoring the samples along multiple partially conflicting dimensions (fidelity, privacy, utility) — and the "label" itself depends on which of those dimensions the user cares about. This is why our oracle (Eq. 1 below) is prohibitively expensive at inference time, why the ground-truth corpus (§3.2.4) is small and precious, and why the selector must be *zero-shot* with respect to the deployment dataset: it may consult offline measurements, but it must not train generators on the user's data before recommending.

3.1.2 Two systems, one dependency

The methodology therefore separates into two systems:

1. **Table-Synthesizers** — the *generation and measurement infrastructure*: the concrete algorithm space $\mathcal{A}$ behind a unified interface, and the factory that manufactures the performance space $\mathcal{Y}$ (the ground truth) offline.
2. **Synthony** — the *selection layer*: the feature extraction $\phi : \mathcal{P} \to \mathcal{F}$ and the selection mapping S, decomposed into Profile $\to$ Analyze $\to$ Rank $\to$ Recommend (§3.3).

The separation is methodological, not organizational. Every quantitative claim about Synthony's selection quality is, transitively, a claim about measurements taken on Table-Synthesizers' output: the selector is exactly as trustworthy as the ground truth it was calibrated against, and that ground truth is manufactured entirely by the infrastructure layer (§3.4).

3.2 Table-Synthesizers: Generation Infrastructure and Ground-Truth Generator

3.2.1 Unified synthesis interface

Table-Synthesizers wraps 19 published tabular generation algorithms behind one abstract interface (`BaseSynthesizer`): `fit(data)`, `sample(n)`, `get_state()`/`load_state()`, with automatic encoding and decoding of mixed-type pandas DataFrames (label encoding for categoricals, model-specific transforms applied symmetrically at fit and sample time). Heavyweight models with fragile dependency stacks (TabSyn, TabDiff, TabDDPM) run through subprocess isolation, so a crash or CUDA fault in one candidate cannot corrupt a benchmark sweep.

The uniform interface is what makes a *fair* cross-family benchmark possible: every generator receives identical input, identical preprocessing, and produces output scored by identical measurement code. Without this layer, per-model differences in data handling (e.g., which encoder a GAN uses for categoricals versus how a copula discretizes them) would confound any comparison of the generative algorithms themselves. This is the same standardization argument made by SDV [PWV16] and SynthCity [QCS23]; the distinction (paper §2.2) is that those libraries stop at standardization and benchmarking, whereas here the standardized output becomes training signal for an automated selector.

3.2.2 The wrapped model families

The 19 models span every major architectural family of tabular generation. We describe each family's generative mechanism briefly, since the *capability dimensions* of §3.3.2 are abstractions over precisely these mechanisms. (Generators cited elsewhere in this thesis but absent from this list — REaLTabFormer and ReTabSyn — are reference-only context and are not implemented in either system.)

Statistical and resampling. - *CART* ([Rei05]; as popularized by `synthpop`, [NRD16]): sequential conditional synthesis — column j is drawn from a regression/classification tree fit on columns $1..j-1$. No gradient training, near-zero latency, strong marginal fidelity on moderate-size data. - *SMOTE* [CBH02]: k-NN interpolation between existing records; technically an oversampler, included as a fast, interpolation-based baseline. - *GaussianCopula* ([PWV16]; Sklar's theorem, [Skl59]): fits marginal distributions per column and couples them with a Gaussian copula over the rank-transformed space.

Probabilistic graphical models. - *BayesianNetwork* [ZCP14]: structure learning over a DAG plus conditional probability tables; sampling is ancestral. Interpretable dependence structure, but continuous columns must be discretized.

Deep generative — GANs. - *CTGAN* [XSC19]: mode-specific normalization (per-column variational Gaussian mixtures) plus conditional-vector training-by-sampling to combat categorical imbalance — the canonical answer to the *Zipfian imbalance* failure mode, §3.3.1. - *PATECTGAN* (PATE framework: [PAE17]; PATE-GAN: [JYS19]): CTGAN's generator trained against an ensemble of teacher discriminators aggregated with differential privacy.

Deep generative — VAEs. - *TVAE* [XSC19]: variational autoencoder with the same mode-specific normalization as CTGAN; empirically the strongest latency/quality trade-off among the deep models in our benchmark. - *LTM-VAE* (internal lab model, unpublished): latent table modeling with a VAE over a learned latent space.

Deep generative — diffusion. - *TabDDPM* [KBR23]: denoising diffusion ([HJA20]) with Gaussian diffusion for numeric columns and multinomial diffusion for categoricals. - *TabDiff ([SXH25]; ICLR 2025)*: joint continuous-time diffusion with per-column learnable noise schedules.

Deep generative — hybrid latent diffusion. - *TabSyn* [ZZS24]: diffusion in the latent space of a purpose-built VAE — score-based generation decoupled from mixed-type

encoding. - *AutoDiff* [SLH23]: autoencoder plus diffusion in the learned latent space, similar decomposition.

Deep generative — flows. - *NFlow* (normalizing flows; [PNR21], via SynthCity): invertible transforms with exact likelihood.

Tree-based adversarial. - *ARF* ([WBK23]): adversarial random forests — iterated real-vs-synthetic forest discrimination refines a piecewise-uniform density estimate. No neural network, remarkably strong on small data (the paper's tie-breaker rule prefers it below 500 rows).

LLM/transformer-based. - *GReaT* [BSL23]: rows serialized to text ("Age is 42, Sex is male, ..."), a pretrained GPT-2-class model fine-tuned on them, generation by sampling and parsing. Brings world knowledge; pays for it in latency.

Foundation-model in-context. - *TabPFGen* ([MDS24]; built on TabPFN, [HME23]): energy-based SGLD sampling guided by TabPFN's pretrained prior — no per-dataset gradient training at all.

Differential-privacy mechanisms. - *AIM* ([MMS22]): workload-adaptive iterative mechanism selecting which noisy marginals to measure under a privacy budget, generation via probabilistic inference (Private-PGM). - *DPCART*: CART with differentially-private tree pruning.

Baseline. - *Identity*: returns the training data unchanged — perfect fidelity, zero privacy, the calibration anchor for both metric families.

3.2.3 Role 1: the candidate pool

Three nested model counts must be kept distinct throughout the thesis; they differ by design maturity, not accident:

The benchmark's N=10 pool (paper §4.1) spans four categories — differential privacy, deep generative, tree-based, statistical — with Identity excluded as trivially perfect on fi-

Table 3.1: Three nested model counts: what the library builds, what the registry knows, what the benchmark evaluates.

Scope	Count	Members
Library can build (`DEFAULT_MODELS`)	**19**	all of §3.2.2
Synthony's capability registry	**15**	19 minus GaussianCopula, LTM-VAE, TabDiff, TabPFGen
Published benchmark oracle	**10**	AIM, DPCART · AutoDiff, TabDDPM, TVAE, NFlow · ARF, CART · BayesianNetwork, SMOTE

delity and uninformative for selection. The 19→15→10 attrition is itself a finding about the cost of ground truth (§3.4.3).

3.2.4 Role 2: ground-truth generation

Benchmark corpus. Seven public datasets sourced via OpenML [VRB13], chosen to spread the four stress dimensions of §3.3.1 rather than to maximize size. Statistics below were re-measured from the repository's actual CSVs:

The underlying paper states the benchmark's dimensionality range as "6–17 features." Every copy of `faults.csv` in the benchmark pipeline — the original input file and all synthetic outputs across all three recorded benchmark trials (`trial1`, `trial4`, `spark`) — has 34 columns, and no reduced-feature variant of `faults` exists anywhere in the repository. The paper's range is therefore corrected here to 6–34; `faults` was always the 34-column dataset shown above (validity-audit item D, Experiments §5.10).

Table 3.2: Benchmark datasets with statistics verified from the repository data and salient stressor annotations.

Dataset	Rows	Cols	Types	Salient stressors
IndianLiverPatient	579	11	mixed (Gender categorical; bilirubin/enzyme assays continuous)	small-sample; heavy right skew in enzyme columns
insurance	1,338	7	mixed (sex/smoker/region categorical; charges continuous)	skewed target (charges); small
faults (steel plates)	1,941	34	continuous + binary fault indicators	high dimensionality; correlated geometry features
Obesity	2,111	17	mixed, categorical-heavy (7+ categorical)	categorical interactions; synthetic SMOTE-augmented provenance
abalone	4,177	9	1 categorical (Sex) + 8 continuous	inter-feature correlation (weights/dimensions nearly collinear)
wilt	4,839	6	5 continuous + binary class	severe class imbalance (Zipfian on the label)
Bean (dry bean)	13,611	17	16 continuous + 7-way class	largest; near-degenerate correlations among shape features

Simulation. Every candidate in the pool is trained on every original dataset and generates a synthetic counterpart of matched size — the full factorial sweep that the oracle definition requires and that inference-time selection must avoid. One cell (insurance, SMOTE) is absent; insurance rankings therefore contain 9 models (paper §4.1).

Dual measurement. Two independent readings are taken of each synthetic output; keeping them distinct is load-bearing for the calibration argument in §3.3.4.

(a) Profile-comparison measurement (implementation: `scripts/generate_model_capabilities.py`; formulas: `docs/scoring_methodology.md`). For each (dataset, model) pair the synthetic output's statistical profile — per-column skewness, cardinality, and the inter-column correlation matrix — is compared against the original's, dimension by dimension. Fixed formulas map preservation ratios onto discrete capability scores $c \in \{0, \ldots, 4\}$ ($\geq$0.90 preservation $\rightarrow$ 4 "excellent," down to $<$0.25 $\rightarrow$ 0 "fails"). One measurement subtlety: cardinality preservation is density-normalized (unique-count ratio corrected for synthetic/original row-count ratio) to remove the bias that arises when a generator emits fewer rows than the original (methodology doc §3.2, v7.0.0 change). This measurement answers *what does this model's output look like* — bottom-up, per-model, per-dimension — and it seeds the capability registry.

(b) Oracle-ranking measurement (implementation: `scripts/build_ground_truth.py`). For each (dataset, intent) pair, all candidates are ranked by the metric that intent scalarizes to:

- **Privacy** — *Proportion Closer to Real*: the fraction of synthetic records lying closer to a real training record than to other synthetic records — a distance-to-closest-record–family re-identification proxy (cf. [SOT22]); lower is better.
- **Fidelity** — *Column Shape Score*: per-column distributional similarity (Kolmogorov–Smirnov complement for continuous columns, total-variation complement for categoricals, as in SDMetrics), averaged; higher is better.

- **Utility** — *train-on-synthetic, test-on-real* (TSTR; [EHR17]): classifiers/regressors fit on synthetic data, evaluated on held-out real data (test ROC-AUC for classification, adjusted R^2 for regression).

- **Latency** — wall-clock train-plus-generate seconds (lower is better). This is the third intent in the released evaluation whose results Chapter 5 reports; the TSTR utility metric above is examined alongside it as a sensitivity oracle (§5.11), and selection quality is robust to the choice.

This measurement answers *which model actually wins* — top-down, per-pair — and it serves as both calibration target (§3.3.4) and evaluation yardstick (Chapter 5).

The distinction between (a) and (b) is the methodological core: (a) grounds the registry in measured per-dimension behavior, giving the scoring function an interpretable inductive bias; (b) provides end-to-end supervision. Collapsing them — learning capability scores directly from oracle rankings with no profile-comparison grounding — is precisely the joint-optimization ablation, and it overfits (§3.3.4).

3.3 Synthony: Profile, Analyze, Rank, Recommend

Synthony approximates the oracle

$$g^* \;=\; \arg\max_{g \in \mathcal{G}} \; \mathcal{F}(\mathbf{m}_g \mid \mathcal{I}) \tag{1}$$

(where $\mathbf{m}_g$ is generator g's metric vector on dataset D and $\mathcal{F}$ the intent-derived scalarization) with a surrogate that trains no generator. Equation (1) requires the full factorial sweep of §3.2.4; the surrogate requires only §3.2.4's *pre-computed* outputs.

3.3.1 Profile — stress as a synthesis-specific meta-feature

Given dataset D, Synthony computes $\phi(D) \in \mathbb{R}^4$ along four difficulty dimensions, each chosen because it names a *documented failure mechanism* of a generator family rather than a generic dataset statistic:

1. **Long-tail skew** — maximum absolute Fisher–Pearson skewness $g_1 = m_3/m_2^{3/2}$ (third standardized moment) across numeric columns; flagged when any column exceeds $|g_1| > 2.0$. Mechanism: GAN/VAE objectives optimize central tendency; severe marginal skew induces mode collapse in the tails [MGN18].

2. **Cardinality complexity** — maximum unique-value count across columns; flagged above 500. Mechanism: embedding-based categorical encoders spread capacity across hundreds of levels, collapsing modes even when the distribution is not skewed (the "needle-in-haystack" failure mode).

3. **Zipfian concentration** — maximum top-20% frequency ratio across categorical columns (fraction of rows occupied by the most frequent 20% of categories); flagged above 0.80. Mechanism: power-law categorical distributions [New05] cause models with continuous latent bottlenecks to silently erase rare-but-critical tail categories — invisible to aggregate fidelity metrics, catastrophic for minority-sensitive downstream tasks.

4. **Small-sample regime** — total row count; flagged below 500. Mechanism: expressive deep models memorize rather than generalize at low n, leaking private records [SOT22] — the reason ARF/CART dominate the small end of our benchmark.

Profiling cost is $O(\text{rows} \times \text{cols})$ and independent of $|\mathcal{G}|$ — the entire point of the surrogate.

3.3.2 Analyze — from stress to requirements

The continuous profile is quantized into a discrete requirement vector $\mathbf{r}(\phi(D)) \in \{0, \ldots, 4\}^6$. **The implemented derivation** (`recommender/engine.py`, `_calculate_required_capabilities`; thresholds loaded from the registry metadata, verified against the code 2026-07-15) is a *gated three-value scheme per dimension*: the requirement is 0 unless that dimension's stress flag fired, then a moderate level (3) up to a severe boundary, then the maximum (4) at or above it — skew: flag at $|g_1| > 2.0$, severe at ≥ 4.0; cardinality: flag at > 500 unique, severe at $\geq$ 5,000; Zipfian: flag at ratio > 0.8, severe at ≥ 0.9; small-data: flat 4 below 1,000 rows. *(The underlying paper's description of this mapping differs in minor respects from the implementation; this thesis documents the implementation — see Experiments §5.10-I, with reconciliation deferred to Future Work §6.5.)* Two dimensions extend the four stress statistics: a **correlation-handling** requirement from a separate detector whose implemented semantics are *suspected nonlinear dependence* — it fires only when the correlation matrix is dense (fraction of pairs with $|\rho| > 0.1$ exceeds 0.5) **and** mean pairwise linear $R^2 < 0.3$, i.e., dense structure that linear fits fail to explain. Strongly but *linearly* correlated data (abalone's collinear weight columns: density 1.0, mean R^2 0.67) deliberately does not raise this requirement, since linear structure is preservable by any competent model; and a **privacy-DP** dimension that deliberately does *not* derive from the data: it enters only through intent-conditioned scale factors, so a privacy intent can upweight DP capability without imposing a DP requirement on every dataset. The requirement vector states what *this dataset demands* on the same $\{0, \ldots, 4\}$ scale the registry uses to state what *each model provides* — the two sides of the match in §3.3.3.

3.3.3 Rank — capability matching

Each generator carries a registry vector $\mathbf{C}_g \in \{0, \ldots, 4\}^J$ (J=6), seeded by measurement (§3.2.4a). Under intent i with scale factors $\boldsymbol{\alpha}_i \in [0, 10]^J$:

$$\text{Score}(g, i) \;=\; \sum_{j=1}^{J} \alpha_{i,j} \cdot c_{g,j} \cdot r_j(\phi(D)) \tag{2}$$

with a soft-thresholding match: full credit (1.0) when $c_{g,j} \geq r_j$, graded partial credit (0.7, 0.4) for near-misses, zero otherwise — graceful degradation instead of hard constraint satisfaction. The additive form is a deliberate epistemic choice: every recommendation decomposes into J inspectable per-dimension contributions that a practitioner can audit and override, at the acknowledged cost of missing stressor interactions (a dataset that is *both* high-cardinality *and* severely skewed may be harder than the sum of its parts; capturing this requires higher-order terms or a learned model — deferred to Future Work).

3.3.4 Recommend — and how the free parameters are calibrated

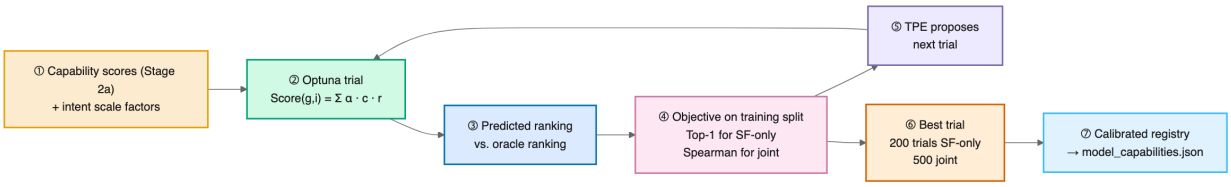

Figure 3.1: Bayesian calibration loop (Optuna/TPE): candidate parameters are scored against the oracle rankings on the training split under the variant's training objective — Top-1 for SF-only (200 trials), Spearman for joint (500 trials) — and the best trial is written to the registry.

The top-ranked generator is recommended with ranked alternatives (Figure 3.1 diagrams the calibration pipeline). The free parameters of Eq. (2) are calibrated offline against the oracle rankings (§3.2.4b) by Bayesian optimization with the Tree-structured Parzen Estimator [BBB11]: TPE models $p(\text{params} \mid \text{good trials})$ and $p(\text{params} \mid \text{bad trials})$ as separate densities and proposes parameters maximizing their ratio — appropriate here because the objective is a non-differentiable rank statistic over a mixed integer/continuous space. The

training objective differs by variant, and the difference is deliberate. The *joint* variant maximizes mean **Spearman rank correlation** between predicted and oracle rankings on the training split (500 trials, 50 startup), because with 78 free parameters Top-1 is too sparse a signal — each of the 12 training pairs moves it by 1/12 — whereas Spearman rewards correct ordering across the full ranking. The *SF-only* variant, whose 18-parameter space is small enough to tolerate the sparser signal, uses **Top-1 accuracy** (200 trials). The two variants therefore differ in both parameter count and objective; that confound is inherited from the underlying paper and is carried forward explicitly rather than hidden (Experiments §5.5, §5.10-G).

Two variants bound how much is learned:

- **SF-only** — 18 parameters (6 capabilities × 3 intents); capability scores stay frozen at their measurement-grounded values.
- **Joint** — 78 parameters (60 capability scores, integer $\{0..4\}$, plus 18 scale factors, float $[0, 10]$).

The published outcome — SF-only generalizes better (held-out Spearman 0.573 vs. 0.557; joint Top-1 collapses to 0) — is the empirical defense of the central methodological choice: *keep the registry grounded in direct measurement and learn only the intent weighting.* Freeing all 78 parameters lets the optimizer discard the inductive bias that the profile-comparison measurement encoded, and 12 training pairs cannot re-derive it.

An optional **hybrid path** passes the top 3–5 rule-based candidates to an LLM for re-ranking with natural-language reasoning, falling back to the pure rule-based ranking on LLM unavailability or failure. The LLM is never the anchor — only a re-ranker over an already-grounded shortlist — a design decision reinforced by the zero-shot LLM baseline's performance in Chapter 5.

3.4 The Dependency Between the Two Systems

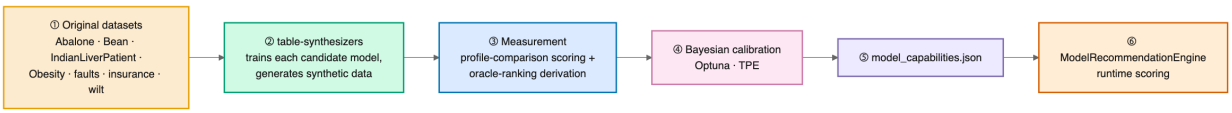

Figure 3.2: Registry-construction data flow: original datasets, Table-Synthesizers simulation, dual measurement, Bayesian calibration, the calibrated registry, and runtime use.

3.4.1 Data flow

The end-to-end flow is diagrammed in Figure 3.2. Original datasets → Table-Synthesizers simulation → dual measurement (§3.2.4a + §3.2.4b) → Bayesian calibration (§3.3.4) → `config/model_capabilities.json` → runtime recommendation (§3.3.3). Everything left of the registry file is offline and re-runnable; nothing right of it trains a generator.

3.4.2 The zero-shot property, stated precisely

Because Synthony never trains a generator at inference time, its knowledge of generator behavior is exactly as current, complete, and correct as the last Table-Synthesizers benchmark run. The benefit is the deployment property the oracle-informed kNN baseline lacks: kNN must interpolate from precomputed oracle rankings of neighbor datasets — i.e., it needs $\mathcal{Y}$ at test time — while Synthony needs only the registry, so a newly wrapped generator becomes recommendable after one offline measurement pass, with no oracle rerun. The liability is bounded staleness: the registry inherits the benchmark's blind spots.

3.4.3 Coverage boundary (honest limitation)

The 19/15/10 attrition of §3.2.3 quantifies that liability. Four library models (GaussianCopula, LTM-VAE, TabDiff, TabPFGen) were never simulated into the registry — they cannot

be recommended regardless of merit. Five registry models (CART, DPCART, TabSyn, AutoDiff, SMOTE) currently lack an execution runner in Synthony's own agentic pipeline, so they receive no *fresh* calibration signal beyond the original static benchmark. And the oracle itself covers 10 models $\times$ 7 datasets $\times$ 3 intents with one missing cell — a corpus small enough that its composition visibly shapes results (CART's 8/21 oracle wins reflect the benchmark's moderate-size, low-cardinality skew; paper §4.5). Replacing this one-shot static corpus with continuously accumulated live outcomes is deferred to Future Work, consistent with the thesis's scoping decision that all post-publication implementation remains there.

CHAPTER 4

System Architecture

4.1 Two Repositories, One Pipeline

The framework is implemented as two independently usable systems whose only coupling is a data contract:

- **Table-Synthesizers** (Python library): the generation and measurement substrate — 19 wrapped generators, one interface, one evaluation code path (Methodology §3.2).
- **SYNTHONY** (Python library + REST API + MCP server + CLI): the selection layer — profiling, requirement analysis, capability scoring, recommendation (Methodology §3.3).

The coupling contract is deliberately thin: Table-Synthesizers' benchmark output (per-model synthetic datasets and their evaluation metrics) flows into SYNTHONY's registry-construction scripts as *files*, not as API calls. Neither system imports the other; either can be replaced behind its contract. This is why the selection layer can claim zero-shot deployment (Methodology §3.4.2): at inference time the only artifact it needs from the substrate is the calibrated registry file.

4.2 Table-Synthesizers: Internal Data Flow

Reading the stages of Figure 4.1, tied to the actual code:

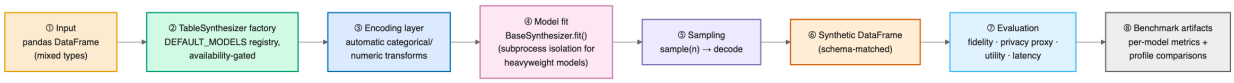

Figure 4.1: Table-Synthesizers generation/measurement data flow (left to right): input DataFrame through the availability-gated factory, symmetric encoding, model fitting with subprocess isolation, sampling, evaluation, and benchmark artifacts.

Stages 1–2. A caller passes a raw mixed-type DataFrame and a model name; the factory (`src/stg/tableSynthesizer.py`) resolves it against `DEFAULT_MODELS`, a registry that is *availability-gated* — models whose optional dependency stacks fail to import are excluded at load time rather than crashing at fit time. This gating is why the implemented-model count is environment-dependent and why the thesis reports it as verified-by-import, not by documentation (Methodology §3.2.3).

Stages 3–5. Every generator sits behind the same four-method interface (`fit` / `sample` / `get_state` / `load_state`), with encoding and decoding applied symmetrically so each model sees data in its native format while callers only ever touch DataFrames. Heavyweight models with fragile dependencies (TabSyn, TabDiff, TabDDPM) execute in isolated subprocesses: a CUDA fault or crash in one candidate cannot corrupt a benchmark sweep — an engineering property that directly protects the ground truth's completeness (a crashed cell becomes a missing measurement, not a corrupted sweep).

Stages 6–8. The synthetic output is schema-matched to the input, then scored by the shared evaluation path: the per-intent metrics of Methodology §3.2.4 plus the profile-comparison statistics (synthetic-vs-original skewness, cardinality, correlation). Stage (8)'s artifacts are the entire interface to the selection layer.

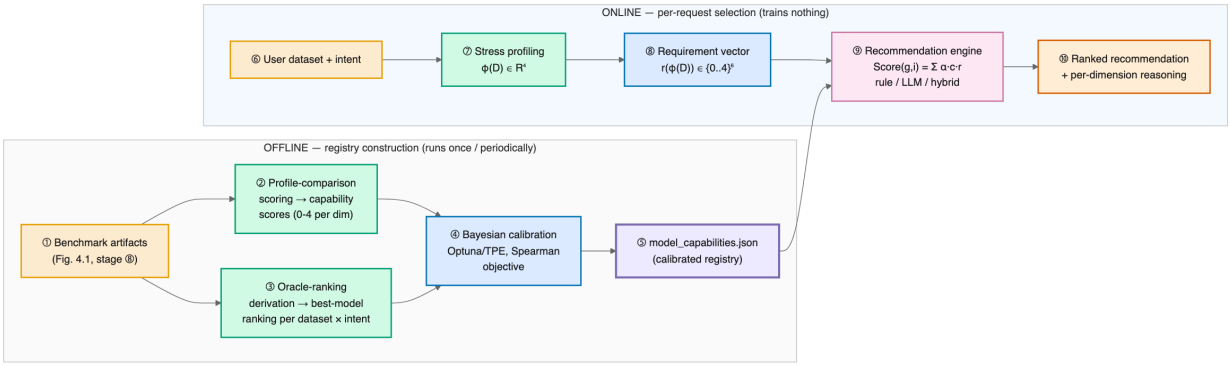

Figure 4.2: End-to-end integration flow. Offline half: benchmark artifacts through dual measurement and Bayesian calibration to the registry file. Online half: user dataset and intent through stress profiling and requirement derivation to the recommendation engine. Exactly one edge (the registry file) crosses the offline/online divide.

4.3 SYNTHONY Integration Flow: From Benchmark Artifacts to a Recommendation

Figure 4.2 makes the thesis's central architectural claim visually checkable: **exactly one edge crosses the offline/online divide** — the registry file ((5) → (9)). Everything expensive (training all generators, scoring all outputs, calibrating all parameters) is left of the divide and amortized across all future requests; everything right of it is cheap (profiling is O(rows × cols)) and trains nothing. The three request surfaces (REST API, MCP server, CLI) all enter at stage (6) and share stages (7)-(10) without divergence; the surfaces themselves, and the scope of the published MCP server, are stated in §4.4.

Stage notes:

- (2)/(3) are the *dual measurement* of Methodology §3.2.4 — kept as two separate boxes deliberately, since collapsing them is precisely the joint-optimization variant that overfits (Experiments §5.7).

- (4) consumes both: (2) supplies the starting capability scores, (3) the target rankings; the SF-only variant freezes (2)'s output and tunes only intent weights.
- (9)'s internal branching (rule-based / LLM / hybrid dispatch, the hybrid fallback chain) is abstracted to one box per the diagram conventions; its detailed gate logic is an implementation concern outside the scope of this architectural overview.

4.4 Interface Surfaces

Three surfaces expose stages (6)–(10) of Figure 4.2: a REST API (FastAPI, with a SQLite layer that caches profiles and tracks sessions), a command-line interface, and a Model Context Protocol (MCP) server speaking JSON-RPC 2.0 over stdio. All three are documented in the underlying paper's Appendix C. They matter here for one reason: they share a single code path below stage (6), so a capture taken through any one of them is evidence about the system rather than about that interface — which is what makes the live MCP transcripts of Appendix A admissible.

The MCP surface is the one worth stating precisely, because its scope is easy to overstate. What the paper publishes is a *selection* surface:

- **13 tools** — data loading (`list_datasets`, `load_dataset`), profiling (`analyze_stress_profile`), ranking (`rank_models_rule` / `_llm` / `_hybrid`), model inspection (`get_model_info`, `list_models`, `check_model_constraints`), explanation (`explain_recommendation_reasoning`, `get_tie_breaker_logic`), and benchmarking (`benchmark_compare`, `generate_benchmark_dataset`).
- **5 resources** — the model registry, per-model detail, cached dataset profiles, stress-detection thresholds, and the active LLM system prompt, each addressed by URI.
- **4 prompts** — guided workflows, of which `analyze-and-recommend` is the full pipeline and `update-knowledge-base` is the nearest thing to a feedback cycle.

Every one of those is read-and-rank. None trains a generator, and none owns a run: the surface exposes the selector of Methodology §3.3 and nothing to the left of the registry file. The server as it now stands carries four further tools that appear nowhere in the published description — start a synthesis run, poll its state, list runs, cancel one. Those four are precisely the run lifecycle, and the first of them will, on request, train the recommended model, benchmark it, and iterate on a quality shortfall: the closed loop of Future Work §6.3, not the system Chapter 5 evaluates. The boundary between what is published and what is not therefore falls exactly where the offline/online divide of §4.3 falls, which is the reason §4.6 can exclude the execution loop without excluding the protocol.

One caveat carries forward. An agent-invocable tool surface is a *precondition* for agentic operation, not evidence of it. Chapter 5 evaluates the ranking these tools serve; no experiment in this thesis measures an agent's behaviour across the protocol, and the paper is itself explicit that its `update-knowledge-base` loop is human-supervised. The word "agent" is discussed on those terms in §6.3.

4.5 The Data Contract Between the Two Systems

The full coupling surface, enumerable in one table — anything not listed here does not cross the boundary:

Two properties of this contract carry methodological weight. First, it is *replayable*: the offline half can be re-run from retained artifacts without re-training generators, which is what makes registry recalibration (Future Work §6.1) an incremental operation. Second, it is *auditable*: because the registry is a human-readable file of per-dimension integer scores rather than an opaque model, every recommendation decomposes into inspectable contributions — the interpretability property claimed in Methodology §3.3.3 is enforced by the architecture, not just asserted.

Table 4.1: The complete data contract between Table-Synthesizers and SYNTHONY.

Artifact	Producer	Consumer	Form
Per-model evaluation metrics	Table-Synthesizers benchmark sweep	Oracle-ranking derivation ((3))	Spreadsheet/CSV rows per (dataset, model)
Profile-comparison statistics	Table-Synthesizers evaluation path	Capability scoring ((2))	JSON per (dataset, model)
Calibrated registry	Calibration ((4))	Recommendation engine ((9))	`model_capabilities.json` (static file)

4.6 What This Chapter Deliberately Excludes

For scope fidelity with the rest of the thesis: the agentic *execution* loop (run persistence, run-state machine, subprocess training runners, cancellation, multi-turn reasoning) and the downstream multi-agent integration are post-publication work and are treated exclusively in Future Work (Chapter 6). The architecture above is the system as published and evaluated in Chapter 5 (Experiments).

The MCP tool surface is deliberately *not* on that list, because it is published (§4.4); what is excluded is the run-lifecycle tooling layered on it afterwards.

CHAPTER 5

Experiments

5.1 Research Questions

The evaluation is organized around four questions, ordered from the underlying hypothesis to the deployed system:

- **RQ1 (Predictability).** Do dataset stress meta-features predict which synthesizer family succeeds — i.e., is selection learnable from dataset properties at all?
- **RQ2 (Selection quality).** Does capability-based, intent- conditioned scoring select better than deployable zero-knowledge alternatives (heuristics, zero-shot LLMs, chance), and how close does it come to an oracle-informed meta-learner?
- **RQ3 (Calibration contribution).** What do the individual components — stress profiling, capability matching, intent- conditioned scale factors — each contribute, and how much calibration does 12 training pairs actually support?
- **RQ4 (Failure surface).** Where does selection fail, and is the failure structure informative (which intents, which dataset regimes)?

5.2 Experimental Setup

Datasets. Seven public tabular datasets sourced via OpenML [VRB13]: Abalone, Bean, IndianLiverPatient, Obesity, faults, insurance, wilt — 579 to 13,611 rows, mixed continu-

ous/categorical schemas, chosen to spread the four stress dimensions rather than to maximize scale. Verified per-dataset statistics and stressor annotations appear in Methodology §3.2.4 (including the feature-count correction for `faults`, resolved at §5.10, item D).

Intents. Three user intents — privacy, fidelity, and a third quality intent — yielding 7 $\times$ 3 = 21 dataset-intent pairs. The third intent is *latency* in the released evaluation that produced the numbers below; a train-on-synthetic–test-on-real *utility* oracle is examined alongside it as a sensitivity analysis (§5.11), and the selection result is robust to the choice (§5.10, item E).

Candidate pool. N = 10 synthesizers spanning four categories: differential privacy (AIM, DPCART), deep generative (AutoDiff, TabDDPM, TVAE, NFlow), tree-based (ARF, CART), and statistical (BayesianNetwork, SMOTE). Identity is excluded as trivially perfect on fidelity and zero on privacy. One cell (insurance, SMOTE) is absent; insurance rankings contain 9 models.

Ground truth (intent-conditioned oracles). Every generator is trained on every dataset; per-intent metrics — Proportion Closer to Real (privacy, lower better), Column Shape Score (fidelity, higher better), and wall-clock latency (lower better) for the third intent — induce a full oracle ranking per pair (Methodology §3.2.4b). A train-on-synthetic–test-on-real utility oracle (test ROC-AUC for classification, adjusted R^2 for regression; [EHR17]) is examined alongside latency as a sensitivity analysis in §5.11.

Split. 70/30 at the *dataset* level (seed 42): 4 training datasets (12 pairs) and 3 held-out datasets (9 pairs). Dataset-level splitting ensures no dataset appears in both splits under different intents, precluding the leakage a pair-level split would invite for the kNN baseline. This single split is retained only because it is the one the published numbers were computed on; §5.12 replaces it with leave-one-dataset-out over all 21 pairs and with an exhaustive enumeration of all $\binom{7}{3} = 35$ dataset-level splits, and reports the result there as the split-independent one.

5.3 Baselines

Four selection strategies, spanning the deployability spectrum:

1. **Random** — uniform generator sampling; expectation over 1,000 trials.

2. **Vanilla LLM (GPT-4o-mini)** — zero-shot selection from dataset schema and summary statistics, temperature 0, no stress profile or capability scores. **Integrity note (must survive into the thesis):** the published run was compromised — API unavailability caused an alphabetical-order fallback, disclosed in the paper's footnote. This thesis therefore reports the row for completeness but **withholds all interpretive claims about LLM selection capability** unless the baseline is re-run properly (§5.10, item A). Statements in the paper's conclusion attributing behavior to GPT-4o-mini are not repeated here.

3. **Static Heuristic** — a practitioner-mimicking rule using only row count and intent (tree-based for fidelity, a fixed speed order for latency, DP models for privacy), no dataset profiling.

4. **Meta-feature kNN (k = 3)** — 9 generic meta-features, Euclidean neighbor search, aggregation of neighbors' *oracle rankings*. Because it consumes ground-truth evaluation at training time, it is reported as an **oracle-informed reference point**, not a deployable zero-knowledge method.

5.4 Metrics

Over full 10-model rankings per pair (Figure 5.1 diagrams the scoring pipeline): **Top-3 accuracy** and **Spearman rank correlation** as primary metrics — Top-3 reflects the realistic deployment scenario of a short list; Spearman rewards correct ordering across the entire ranking. **Top-1 accuracy** and **NDCG** as secondary. With 9 test pairs, each Top-1

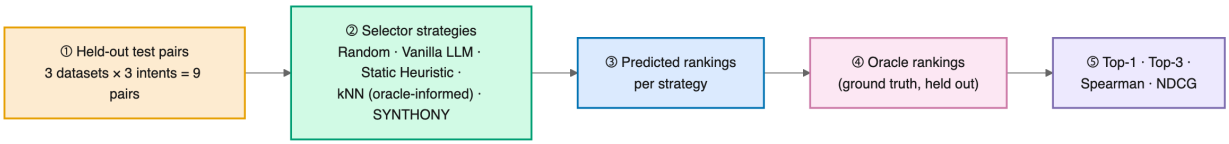

Figure 5.1: Evaluation protocol: held-out dataset-intent pairs are ranked by each selector strategy and scored against held-out oracle rankings.

prediction moves accuracy by ≈11.1 percentage points; differences smaller than ∼0.15 are interpreted with caution throughout.

5.5 SYNTHONY Variants

Two calibration variants (Methodology §3.3.4) bound how much is learned:

- **SF-only:** 18 scale-factor parameters (6 capabilities × 3 intents), hand-crafted capability scores frozen; Optuna, 200 trials, Top-1 training objective.
- **Joint:** 78 parameters (60 capability scores + 18 scale factors); 500 trials, Spearman training objective (Top-1 is too sparse a signal for 78 parameters).

The two variants differ in *both* parameter count and objective — a confound the paper acknowledges; controlled variants live in its Appendix A (§5.10, item B, for the parameter-count bookkeeping).

5.6 Main Results (RQ2)

Test-set performance, 9 held-out pairs (paper Table 1; bold = best zero-knowledge method; Figure 5.2 plots the same results):

† Compromised baseline; see §5.3 and §5.10-A.

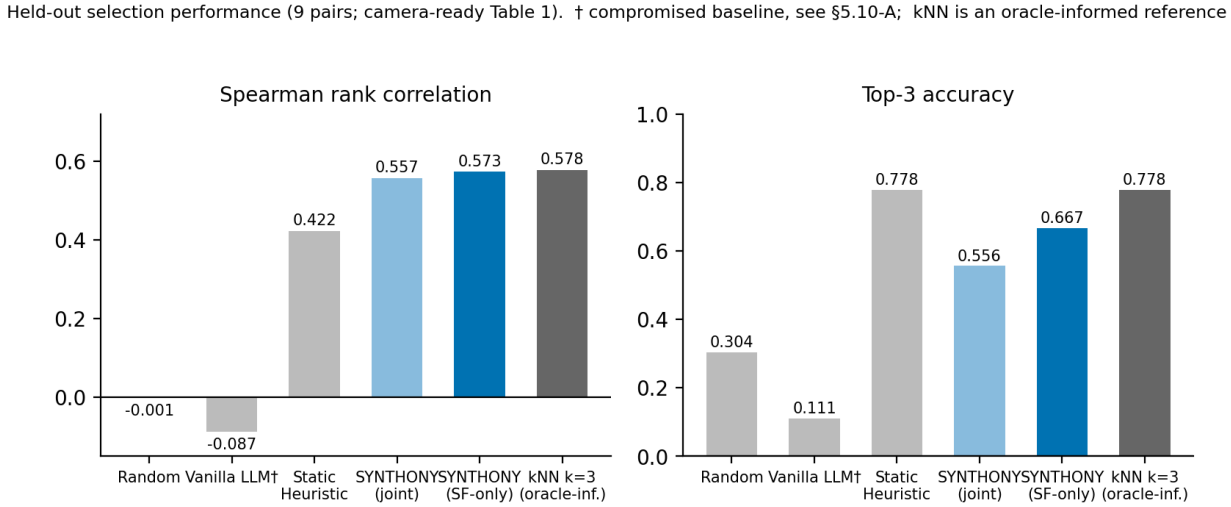

Figure 5.2: Held-out selection performance (9 pairs): Spearman rank correlation and Top-3 accuracy per strategy. The Vanilla LLM baseline is compromised (Section 5.10, item A); kNN is an oracle-informed reference.

Table 5.1: Held-out test-set performance of selection strategies (9 dataset-intent pairs).

Strategy	Top-1	Top-3	Spearman	NDCG
kNN (k=3), *oracle-informed reference*	.556	.778	.578	.918
Random (E[1000])	.111	.304	−.001	.815
Vanilla LLM (GPT-4o-mini)†	.000	.111	−.087	.784
Static Heuristic	.333	.778	.422	.889
SYNTHONY (joint opt)	.000	.556	.557	.893
SYNTHONY (SF-only)	**.444**	**.667**	**.573**	**.914**

Three readings, in decreasing strength of evidence:

1. **SF-only is the best deployable method** on Spearman (0.573) and Top-1 (0.444), and it nearly matches the oracle-informed kNN reference (0.578 Spearman) despite a fundamental asymmetry: kNN interpolates from ground-truth rankings of neighbor datasets at inference time; SYNTHONY needs no oracle access at all. The practical consequence: a newly registered synthesizer becomes recommendable after one offline measurement pass, with no benchmark rerun — a property kNN structurally cannot offer.

2. **Joint optimization does not generalize better** (0.557 Spearman; Top-1 collapses to 0): freeing all 78 parameters on 12 training pairs discards the measurement-grounded inductive bias without enough signal to re-derive it. This is the empirical defense of the SF-only design (Methodology §3.3.4), modulo the objective confound noted in §5.5.

3. **The Static Heuristic's profile is diagnostic**: high Top-3 (0.778, from always placing strong models in a short list) with poor Spearman (0.422) — shortlist plausibility without full-ranking quality. This is the pattern the Spearman metric exists to expose.

5.7 Ablations (RQ3)

All ablation variants use SF-only optimization (18 parameters) to isolate component effects (paper Table 2; Figure 5.3 plots the Spearman column):

- **Stress profiling is the load-bearing component**: removing it causes the largest Spearman drop ($0.573 \rightarrow 0.430$). Without dataset-derived requirements, the system cannot differentiate which models suit which datasets.

- **Focus scaling currently *hurts*** on this sample: removing it *improves* Spearman ($0.573 \rightarrow 0.602$). Twelve training pairs are not enough to learn 18 scale factors reliably

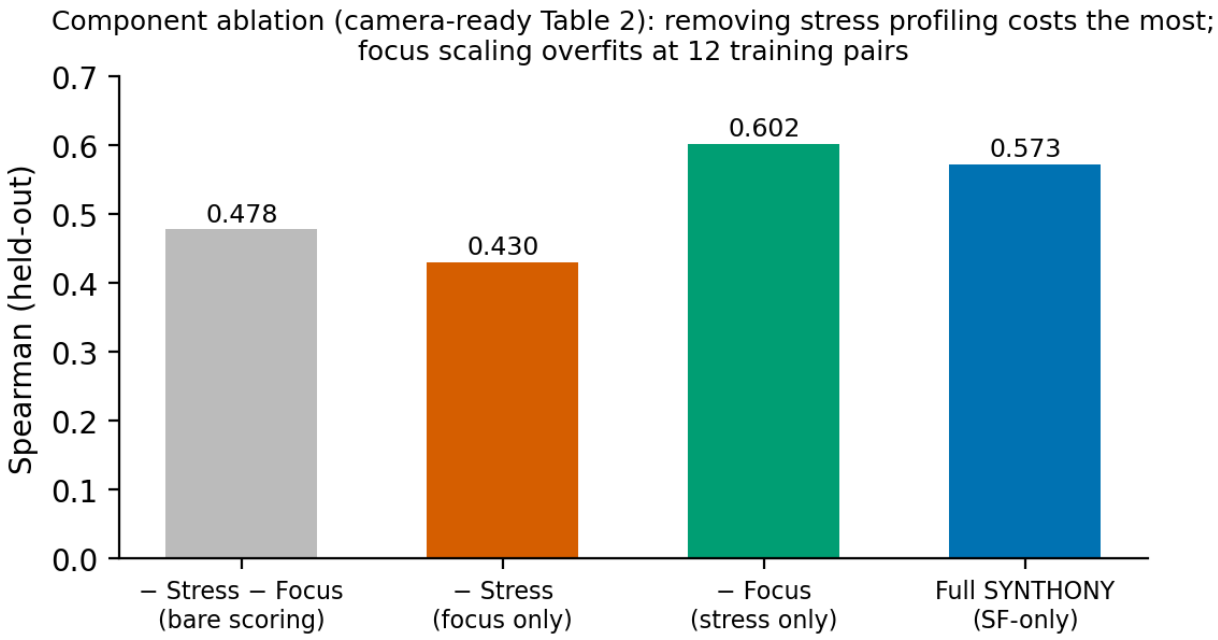

Figure 5.3: Component ablation (held-out Spearman): removing stress profiling costs the most; removing focus scaling improves generalization at the current calibration sample size.

Table 5.2: Component ablation on the SF-only pipeline (held-out, 9 pairs).

Variant	Top-1	Top-3	Spearman	NDCG
Vanilla LLM (no SYNTHONY)†	.000	.111	−.087	.784
− Stress − Focus (bare scoring)	.444	.667	.478	.904
− Focus scaling (stress only)	.444	.667	**.602**	.910
− Stress profiling (focus only)	.333	.667	.430	.901
Full SYNTHONY (SF-only)	.444	.667	.573	.914

— the paper conjectures (testably) that focus scaling turns beneficial at $\geq$20 datasets, particularly for the privacy intent. The thesis should present this as an honest negative result about calibration sample size, not as a defect of intent conditioning as a concept.

- **Even bare capability matching carries signal** (0.478 vs. random's -0.001), confirming the registry encodes real structure before any calibration.

5.8 Do Stress Profiles Predict Synthesizer Families? (RQ1)

Predicting the oracle-best synthesizer directly from dataset meta-features (paper Table 3; test set, 9 pairs):

Table 5.3: Predicting the oracle-best synthesizer directly from dataset meta-features (held-out, 9 pairs).

Predictor	Top-1	Top-3	Spearman	NDCG
Majority vote	.333	.778	.300	.899
Logistic regression	.556	.889	.293	.893
Decision tree	.222	.667	.299	.900
kNN (k=3)	.556	.778	**.578**	.918

The discriminative signal lives in *full-ranking* quality: kNN's Spearman (0.578) nearly doubles the frequency-based and parametric baselines ($\sim$0.30), while majority vote's respectable Top-3 (0.778) merely restates CART's frequent wins. This validates the stress-profiling hypothesis (RQ1: dataset difficulty signatures predict synthesizer efficacy in a learnable way) — and, notably, SYNTHONY's capability-mediated 0.573 nearly matches kNN's direct interpolation despite generalizing through an abstraction layer rather than memorized rankings.

5.9 Per-Intent Analysis and the Failure Surface (RQ4)

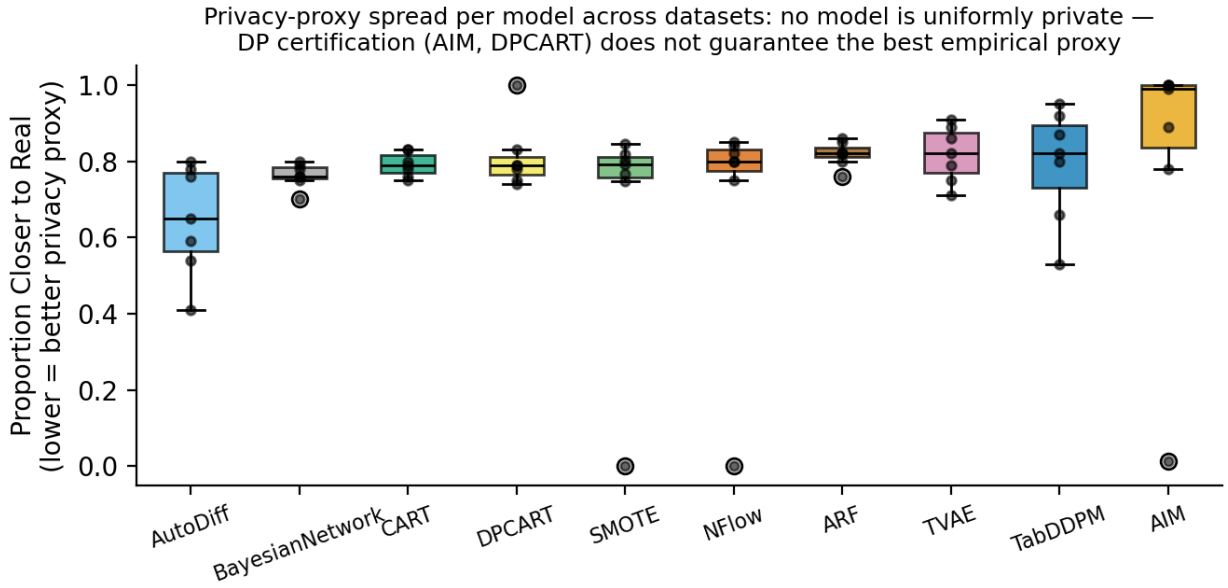

Figure 5.4: Privacy-proxy spread (Proportion Closer to Real; lower is better) per model across datasets: no model is uniformly private, and DP certification does not guarantee the best empirical proxy.

CART dominance and what it means. CART is oracle-best on 8/21 pairs overall, but recomputing the winners from the recovered evaluation matrix (§5.11) shows this decomposes into 7 of 7 *latency* wins — CART trains in near-zero time — plus a single fidelity win. Its apparent dominance is therefore largely an artifact of the latency intent rewarding speed rather than of across-the-board quality. The quality intents are more contested: tree-based methods (ARF, CART) still take 4 of 7 fidelity wins — unsurprising for a corpus of moderate-sized (500–13,000 rows), low-cardinality tables, the regime in which adversarial random forests were shown to match or beat state-of-the-art probabilistic circuits and deep generative models on tabular benchmarks, some two orders of magnitude faster [WBK23] — but no single model runs the table. The stronger reading is about the corpus rather than the literature: this benchmark is composed of exactly the data on which tree-based synthesis is

cheap and effective, which is a limitation of the ground truth (§6.2), not an endorsement of trees. A degenerate "always predict CART" selector achieves 33.3% Top-1 on the test set, which bounds what overall Top-1 can show. The discriminative evidence therefore lives in the *privacy* and *fidelity* intents and in full-ranking metrics.

Privacy is where selection matters — and where it is hardest. Six distinct model families win privacy across the seven datasets (AutoDiff 2; NFlow, AIM, TVAE, BayesianNetwork, and SMOTE one each, recomputed from the recovered matrix): the strongest possible signal that no fixed rule suffices. It is also the hardest intent for every method — the Static Heuristic's DP-models-for-privacy assumption fails outright (0% privacy Top-1) because DP certification does not reliably predict the empirical proxy: non-DP diffusion and flow models sometimes achieve lower distance-to-closest-record through smoother density estimation (Figure 5.4: no model is uniformly private, and the DP models are not the empirical winners). SYNTHONY's correct privacy predictions, though limited, demonstrate that stress profiling captures signal simple heuristics miss. On the 9 test pairs, SF-only's errors concentrate on privacy; the Static Heuristic's 3/9 correct Top-1 picks all fall under the non-privacy intents.

Latency observation. Where latency is examined (paper §4.7 context), CART completes in sub-second time — reported as "sub-second," not "0-second," per §5.10-F.

5.10 Threats to Validity and Known Reporting Issues

Items are listed with their disposition in this thesis; none should silently disappear at typesetting.

Table 5.4: Consolidated threats-to-validity audit with per-item dispositions.

#	Issue (from the critique)	Disposition in the thesis
A	Vanilla LLM baseline is an alphabetical-order fallback (API unavailability), yet the paper draws interpretive conclusions from it	Row reported for completeness, flagged †; **all interpretive claims about LLM capability withheld**. Re-running the baseline is required before restoring them (also Future Work §6.2)
B	Parameter-count inconsistency across paper sections (54 vs. 78: 6-generator vs. 10-generator accounting; the released `optimize_scaling.py` optimizes 6 overlap models)	**Resolved by direct verification against the published PDF**: the published paper contains no 54-parameter accounting anywhere — its Appendix A.2 specifies the Joint space as 60 capability scores (10 models × 6 dimensions) plus 18 scale factors, totalling 78. The 54-parameter/6-generator-overlap accounting survives only in unpublished internal drafts and in the released `optimize_scaling.py` (whose `OVERLAP_MODELS` constant targets that six-model subset) — a draft-and-code artifact, not a published inconsistency. The residual gap — that the committed script does not implement the published 78-parameter configuration — is tracked under item J

Table 5.4 (continued)

#	Issue (from the critique)	Disposition in the thesis
C	Main-text Table 1 vs. appendix Table 4 report different numbers for the same variant	**Resolved by direct verification against the published PDF**: in the published paper, Appendix A.5's Table 4 test row (.000/.556/.557/.893) is numerically identical to the main-text Table 1 Joint row. The differing numbers recorded in the pre-submission critique trace to a superseded earlier draft (which used a 7-pair test split); no mismatch exists in the final publication
D	Dataset feature-range claim ("6-17 features") conflicts with `faults.csv` (34 columns) in the repository	**Resolved** (Methodology §3.2.4): every copy of `faults.csv` in the pipeline — input and all synthetic outputs across all three recorded trials — has 34 columns; no reduced variant exists. The paper's range is corrected to 6–34

Table 5.4 (continued)

#	Issue (from the critique)	Disposition in the thesis
E	Released ground-truth script builds privacy/fidelity/**latency** oracles; the thesis narrative frames the third intent as **utility**	**Resolved** (§5.11) on two independent grounds. (i) Reproduction from the recovered evaluation matrix shows the published numbers were computed with **latency** (kNN reference 0.581 vs. published 0.578), while a TSTR-utility oracle yields 0.525; the selection signal is robust across both (0.53–0.58). (ii) The paper is internally inconsistent in the same direction: its Appendix C.2 tie-breaking rules provide for "speed-optimized intents," a category with no referent under a privacy/fidelity/utility triple. Third-intent label corrected to latency; utility retained as a sensitivity oracle
F	"0-second latency" for CART is a timer-resolution artifact	Reported as "sub-second" throughout

Table 5.4 (continued)

#	Issue (from the critique)	Disposition in the thesis
G	SF-only vs. Joint differ in both parameter space and objective (confound)	Acknowledged in §5.5/§5.6, and **sharpened by verification against the published PDF**: the paper states "controlled variants appear in Appendix A," but the published Appendix A contains only the Joint procedure and its results — no SF-only-with-Spearman-objective controlled run exists anywhere in the publication, so there is nothing to port into this thesis. Running the controlled variant requires the calibration pipeline whose artifacts are missing (item J); until then the confound stands unresolved, and this thesis draws no conclusion from the SF-only-versus-Joint gap beyond "Joint did not generalize better"
H	Small test set (9 pairs; ±11.1pp per Top-1 prediction) and majority-class solvability (33.3%)	Stated in §5.4 and §5.9; primary conclusions rest on Spearman and per-intent structure, not overall Top-1. The *single-split* component is **resolved** in §5.12: leave-one-dataset-out over all 21 pairs (pooled kNN 0.543) and exhaustive enumeration of all 35 dataset-level splits (range 0.441–0.651, seed 42 at the 77th percentile) show the result is not split-dependent. The small-n caveat on Top-1 itself stands

Table 5.4 (continued)

#	Issue (from the critique)	Disposition in the thesis
I	Minor discrepancies exist between the paper's method description and the released implementation (requirement-derivation thresholds; correlation criterion), verified in code 2026-07-15	Thesis documents the implementation (Methodology §3.3.2); published results were produced by the implementation and stand unaffected. Reconciliation and possible erratum deferred to Future Work §6.5; full detail internal (`ERRATUM_internal.md`, never in thesis text)
J	SYNTHONY's own SF-only calibrated result (0.573, Table 5.1) cannot be regenerated from the committed repository: the 18 optimized scale-factor parameters that produce it are absent (the registry's scale-factor configuration holds only uncalibrated, uniform placeholder weights). Only the kNN reference, Random, and Static Heuristic rows independently reproduce (§5.11)	**Open**: recovering the calibrated scale-factor artifact, or re-running calibration against the recovered evaluation matrix of §5.11, is the sole outstanding step for a fully self-contained reproduction of Table 5.1; tracked as Future Work §6.1's first item. Until then, 0.573 is reported as published, not independently re-derived by this thesis
K	Three entries in the published paper's reference list do not match the publisher record: the SIGMOD Record survey ([DGP25]) lists an author set and a DOI that resolve to a different paper; the *Neurocomputing* survey ([KRF25]) lists three co-authors who are not on the paper and a DOI that does not resolve; and the *Data Science Journal* benchmark ([DWP25]) is attributed to three authors none of whom is on it. All three were carried into early drafts of this thesis from the paper's reference list	**Resolved** for this thesis: every field was re-derived from the Crossref record of record and the bibliography corrected (author lists, volume/issue, pages, year, DOIs). Two claims that had rested on these sources are also re-pointed: §5.9 and §6.2 now cite [WBK23] for tree-based competitiveness, and Methodology §3.1.1 cites [KRF25] only for its actual finding, budget-dependent ranking instability. The published reference list is uncorrected; an erratum is the natural remedy and is grouped with item I under Future Work §6.5

Statistical power. with 9 test pairs, no pairwise method comparison in Table 1 reaches

conventional significance; the chapter's claims are therefore framed as *consistent evidence across four metrics and ablation structure*, not as significance-tested superiority. Expanding the benchmark (Future Work §6.2) is the remedy, not heavier statistics on n = 9; what *can* be done at this scale — exhausting the split space rather than resampling within one split — is done in §5.12.

5.11 Reproduction and Sensitivity to the Third-Intent Definition

The evaluation matrix underlying Table 5.1 was recovered and re-run post-hoc, enabling a direct reproduction of the oracle-derived rows and a sensitivity check on the choice of third intent — the calibration-versus-evaluation ambiguity flagged as item E of §5.10 above. The recovered matrix covers all eleven wrapped models across the seven datasets, with *Proportion Closer to Real* for privacy, *Column Shape Score* for fidelity, and *both* a wall-clock *latency* measurement and a train-on-synthetic–test-on-real (TSTR) *utility* measurement (test ROC-AUC for classification, adjusted R^2 for regression) for the third intent. Rebuilding the oracle-informed kNN reference from this matrix (eleven models, dataset-level 70/30 split, seed 42) gives:

Table 5.5: Reproduction of the oracle-derived rows under both candidate third-intent oracles. The selection signal is strongly positive under either definition; the published value matches the latency oracle.

Method	Latency oracle	TSTR-utility oracle	Published
kNN (oracle-informed)	0.581	0.525	0.578
Random ($\mathbb{E}[1000]$)	$\approx$0.00	$\approx$0.00	-0.001

Two conclusions follow. First, the published Table 5.1 numbers were produced with **latency** as the third intent: the kNN reference reproduces at 0.581 (versus the published 0.578) under the latency oracle, while the TSTR-utility oracle yields 0.525. This resolves

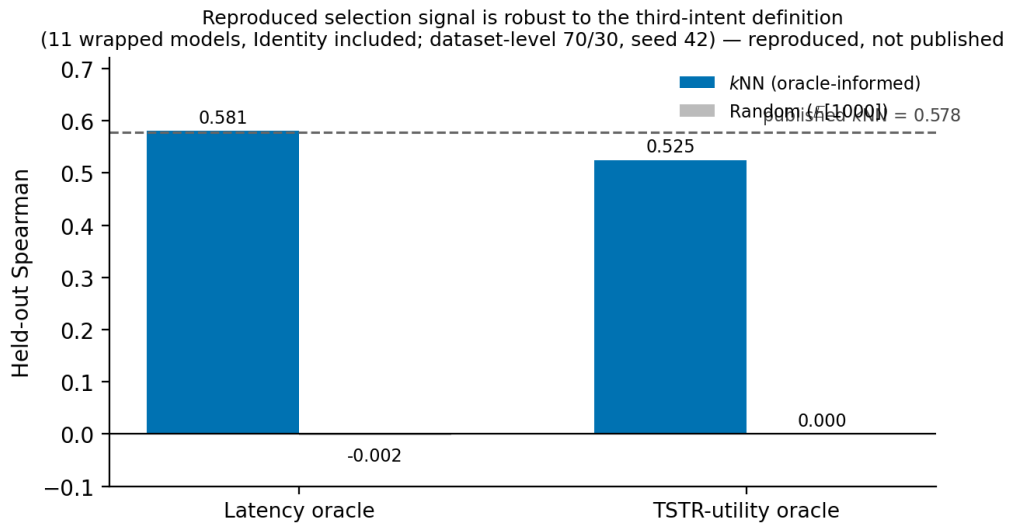

Figure 5.5: Reproduction of the oracle-informed kNN reference (and the random floor) under both candidate third-intent oracles, from the recovered evaluation matrix. The reproduced value (0.581 under latency) sits three thousandths from the published 0.578 (dashed line); the signal is an order of magnitude above the random floor and holds (0.525) when the third intent is downstream utility instead of latency. These are *reproduced* numbers under the 11-wrapped-model, Identity-included configuration that regenerates the published table — distinct from the 10-evaluated, Identity-excluded set used for the winner analysis (Figure 1.1), and not to be conflated with the published values in Figure 5.2.

item E in favour of latency, and corrects the intent label used elsewhere in this chapter.

The numerical match is not the only evidence. The paper's own tie-breaking rules, stated in its Appendix C.2, presuppose a time-valuing intent: they direct that "speed-optimized intents prefer CART," alongside separate provision for "quality-optimized" ones. A speed-optimized intent has no referent in a privacy/fidelity/*utility* triple — TSTR utility is indifferent to how long a generator took to fit — but it is exactly what a privacy/fidelity/*latency* triple's third member is. The paper never uses the word "latency"; its narrative names the third intent *utility* (abstract; §4.1; the scale-factor definition of its Appendix A.2), while its tie-breaker prose and its released ground-truth script both describe a wall-clock quantity. The reproduction above and this internal inconsistency point the same way, from independent evidence: what was measured is time.

A terminological note, since both words appear in the sources. *Latency* and *speed* are inverse measures of one axis, not synonyms: this thesis reports latency — wall-clock train-plus-generate seconds, lower better (§3.2.4) — which is the quantity the released script computes, and uses "speed" only informally. A reader meeting "speed-optimized" in the paper should read it as the same intent. Second, and more consequentially, the central finding is **robust to the third-intent definition**: the oracle-informed reference retains a strong positive rank correlation (0.53–0.58) whether the third intent is latency or downstream utility, an order of magnitude above the random floor (≈0) (Figure 5.5). The reproduction includes all eleven wrapped models (Identity among them); excluding Identity raises the kNN reference to 0.600. SYNTHONY's own SF-only row is not re-derived here, because the released artifacts contain uniform (uncalibrated) scale factors rather than the optimized set (validity-audit item J, §5.10); recovering that calibration is the sole outstanding item for a complete end-to-end reproduction (Future Work §6.1).

5.12 Split Sensitivity: Leave-One-Dataset-Out Cross-Validation

Everything reported above rests on a single dataset-level split — four training datasets, three held out, drawn with seed 42 — and with seven datasets only $\binom{7}{3} = 35$ such splits exist. A conclusion that depends on which three datasets happened to be drawn is not a conclusion. This section replaces the single split with two exhaustive protocols, both computed from the recovered evaluation matrix of §5.11 and both original to this thesis.

Protocol. Only the split axis varies: oracle construction, the six-model prediction universe of the released baselines, the calibration procedure, and the metric implementations are the committed ones under which the published kNN row reproduces (§5.11). Two protocols are run. *Leave-one-dataset-out (LODO)*: seven folds, each holding out one entire dataset (three dataset-intent pairs) and fitting on the remaining six (eighteen pairs), so that every one of the 21 pairs is tested exactly once by a selector that never saw that dataset — the pooled figure uses the whole benchmark rather than a nine-pair subset. *Exhaustive enumeration*: all 35 four/three dataset-level splits, locating the published seed-42 split within the resulting distribution.

Top-1 and Top-3 are structurally depressed in Table 5.6: predictions range over the six overlap models the released baselines rank, while the oracle ranks eleven, so the oracle's best is frequently a model no method can name. Excluding Identity (the ten-model oracle) raises the kNN reference to Top-1 .238 and NDCG .610 while leaving every Spearman value unchanged — Spearman is computed over the intersection of the two rankings, whose relative order removing an unpredicted model does not disturb. Spearman is therefore the statistic to read here, for the reason already given in §3.3.4 and §5.4.

Three findings follow.

The selection signal survives every possible split. Across all 35 dataset-level splits the kNN reference ranges from 0.441 to 0.651 (mean 0.543, SD 0.052); not one split falls below 0.44, against a random-ranking floor of 0.004. The published seed-42 split (0.581) sits

Table 5.6: Leave-one-dataset-out results, pooled over all 21 dataset-intent pairs (each tested exactly once, by a fold that never saw its dataset). The final column is the seed-42 single-split value from the §5.11 reproduction, shown for Spearman only — Top-K and NDCG under the six-model prediction convention are not comparable to the published table. Eleven-model oracle, latency third intent.

	LODO, pooled (n = 21)				Seed-42 (n = 9)
Strategy	Top-1	Top-3	Spearman	NDCG	Spearman
*k*NN (*k*=3), *oracle-informed reference*	.143	.143	**.543**	.558	.581
SYNTHONY (SF-only, re-calibrated per fold)	.048	.095	**.306**	.511	—
SYNTHONY (as released, uncalibrated)	.048	.143	.037	.485	—
Static Heuristic	.048	.190	.450	.529	.524
Random ($\mathbb{E}$[1000])	.041	.120	.004	.482	−.002

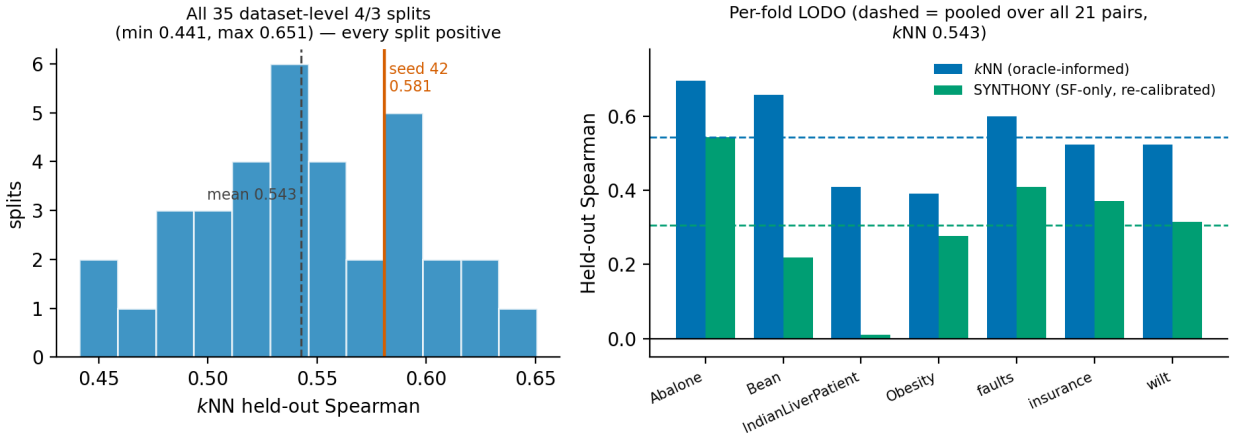

Figure 5.6: Split sensitivity of the oracle-informed *k*NN reference. *Left*: the distribution of held-out Spearman across all 35 dataset-level 4/3 splits; the published seed-42 split (0.581) sits at the 77th percentile of a distribution whose worst case is 0.441. *Right*: per-fold LODO, with pooled values dashed. Thesis-original robustness analysis computed from the recovered evaluation matrix, not published numbers.

at the 77th percentile — mildly favourable, within one standard deviation of the mean, and not an outlier (Figure 5.6, left). The single-split result was not lucky in any sense that would change a conclusion, and the worst split available would have supported the same qualitative claim.

LODO makes the protocol leakage-free, and shows nothing was riding on it. The reproducibility investigation behind §5.11 raised the possibility that the oracle-informed reference — the one baseline that consumes oracle rankings of neighbouring datasets, and hence the most leakage-sensitive — was inflated. Under LODO the question is settled by construction: each pair is predicted from neighbours drawn only from its fold's six training datasets. It is further verified that this neighbour pool is exactly the one the released selector already used, since that selector excludes the query dataset and, with seven datasets, what remains *is* the LODO training set; per-pair scores are numerically identical under the two protocols. What LODO changes is thus not the value but its standing — a number that was leakage-free only as an accident of the neighbour rule becomes leakage-free by design. The pooled LODO value of 0.543 over all 21 pairs, not any single split, is the figure this thesis would defend.

Re-calibrating the released engine per fold recovers much of its ranking quality, and overfits exactly as predicted. SF-only calibration (six scale factors per intent, 400-sample random search on the fold's eighteen training pairs) lifts the committed engine from 0.037 — indistinguishable from random — to 0.306 pooled, confirming the item-J diagnosis of §5.10 that the released registry's uniform scale factors leave intent-conditioning inert. The fold-mean train-to-test gap, $+0.409 \rightarrow +0.306$, reproduces under a leakage-free protocol precisely the honest negative of the ablation in §5.7: eighteen free parameters against eighteen training pairs overfit. This re-calibrated figure is emphatically *not* a reproduction of the published SF-only result (0.573): the calibration procedure differs from the paper's, and the calibrated parameters that produce 0.573 remain absent from the repository (item J). It is a lower bound on what the released artifact achieves under a leakage-free protocol —

neither a re-derivation of the published value nor a challenge to it.

Sensitivity to the third intent. Repeating both protocols against the TSTR-utility oracle of §5.11 gives the same picture: pooled LODO 0.475 for the kNN reference and 0.371 for the re-calibrated engine, a 35-split range of 0.283–0.676, and the seed-42 split at 0.525 (69th percentile). The conclusions of this section do not depend on which third intent is used.

Both protocols are reproducible from the committed artifacts: `scripts/lodo_cv.py` in the SYNTHONY repository, archived with its outputs under `assets/lodo/` here. Per-pair predictions, per-fold scale factors, the full 35-split enumeration, and the pinned configuration (model universe, intents, search budget, seeds) are saved alongside the results.

CHAPTER 6

Future Work

Scope note. All directions in this chapter were unimplemented at the time of the paper's publication (March 2026). Where subsequent prototyping in the project repositories informs a design sketch, it is cited as engineering-in-progress; none of it has been evaluated to the standard of Chapter 5, and no results from it are claimed in this thesis.

6.1 Learning the Capability Registry from Accumulated Outcomes

The paper names its primary future direction explicitly: *learning capability scores from data rather than hand-crafting them.* The methodology chapter showed why the naive version of this fails — the joint-optimization variant, which frees all 78 parameters against 12 training pairs, overfits (held-out Spearman 0.557 vs. SF-only's 0.573). The lesson is not that learning is wrong but that the supervision corpus is too small; the direction is therefore to grow the corpus continuously rather than to learn harder from a static one.

The concrete design generalizes the registry-construction pipeline of §3.4.1 from a one-shot offline benchmark to a living feedback loop. Every full synthesis run — train, generate, benchmark — already produces exactly the labeled example the calibration consumes: a dataset stress profile, a model identity, a benchmark score vector, and (equally valuable) failure labels when a candidate crashes. Four increments follow, in ascending ambition:

1. **Outcome accumulation.** An idempotent extraction of every run's (profile, model,

score vector, success flag) tuple into a durable outcomes store — the successor to the static evaluation spreadsheet of §3.2.4.

2. **Continuous recalibration (Level 0).** Re-pointing the existing Bayesian calibration (§3.3.4) at the growing store, making the registry self-correcting as evidence accumulates rather than frozen at the last manual benchmark.

3. **Meta-learned scoring (Level 1).** Per-model score regressors over stress meta-features — gradient boosting or the meta-feature kNN already present as a Chapter 5 baseline, promoted from baseline to engine option — with the measurement-grounded rule-based scores retained as the cold-start prior when a profile has no near neighbors. This directly attacks the remaining gap to the oracle-informed kNN reference (0.573 vs. 0.578 Spearman) while preserving the zero-shot deployment property kNN lacks.

4. **Exploration (Level 2).** A contextual bandit (Thompson sampling; context = stress profile, arm = model, reward = the scalarized benchmark score) to occasionally try non-top-ranked candidates — without exploration, a selector that always exploits its current best guess never learns whether the runner-up would have won (Thompson sampling and its practical variants: [RVK18]).

One design decision gates all four: the **reward definition**. The benchmark emits a score *vector* (fidelity, privacy, utility); either it is scalarized consistently with how run-acceptance thresholds already judge runs, or the vector is predicted whole and weighted per intent at query time. An accidental, implicit scalarization baked into the outcomes store would silently distort every downstream increment, so this decision must precede the first line of extraction code.

At the data volumes realistically accumulated (dozens to hundreds of runs), the appropriate model class remains calibration and meta-learning — there are no neural weights to train anywhere in this plan, and there should not be.

6.2 Closing the Coverage Boundary and Hardening the Benchmark

Section 3.4.3 quantified the coverage attrition: 19 models the library can build, 15 the registry knows, 10 the oracle covers. Closing it is unglamorous but strictly prior to §6.1 mattering for the missing models — no learning scheme can recommend a model the registry has never heard of, and no outcome store accumulates evidence for a model no runner can execute. Five workstreams:

1. **Registry completion.** Simulate and measure the four library models absent from the registry (GaussianCopula, LTM-VAE, TabDiff, TabPFGen) through the §3.4.1 pipeline. This is registry work, not research, but it adds two architectural families (copulas, foundation-model in-context generation) the selector currently cannot reason about at all.

2. **Runner completion.** Five registry models lack an execution runner in the orchestration pipeline (CART, DPCART, TabSyn, AutoDiff, SMOTE); until built, their capability scores stay frozen at the original static benchmark and drift-prone.

3. **Pool growth is continuous, not one-time.** Newly published generators guarantee the boundary keeps moving — e.g., RL-based synthesis (ReTabSyn; [LKL26]), cross-tabular foundation models, and time-series-native tabular generators, none of which is implemented in either Table-Synthesizers or Synthony today; they are future registration targets only. The time-series case implies an eventual *modality* extension of the stress profile itself (temporal stressors have no dimension in ϕ today).

4. **Benchmark expansion.** The paper's own analysis identifies the corpus composition as the main threat to interpretation: CART's 8/21 oracle wins reflect a benchmark of moderate-size, low-cardinality datasets — the regime in which tree-based synthesis

matches or beats deep generative models at a fraction of their compute [WBK23]. Expanding toward ≥20 datasets with genuinely high cardinality (>1,000 unique values) and larger scale (>100,000 rows) should shift oracle winners toward the deep generative families and reduce majority-class solvability (currently 33.3% Top-1 by always predicting CART). The paper also conjectures that intent-conditioned scale factors — which the ablation found *harmful* at 12 training pairs — become beneficial at this scale, a directly testable prediction.

5. **Baseline repair.** The published zero-shot LLM baseline was compromised by API unavailability (disclosed in the paper's footnote); re-running it properly is required before any claim about LLM selection capability, positive or negative, is defensible.

6.3 Agentic Orchestration

The paper positions SYNTHONY as an "agent," and the title's use of that word outruns what was evaluated: the published system performs stress profiling and capability scoring in a single pass — there is no perception-action loop. The paper does ship the *interface* an agent would act through (the MCP server of Appendix C, §4.3), and it is candid about the gap: its own `update-knowledge-base` prompt, the closest thing to a feedback cycle, is described there as "currently human-supervised (the agent proposes changes; a developer reviews and applies them)." The gap between shipping that surface and closing the loop behind it (§4.4) is what this section specifies. Making the term honest is itself a future-work program, sketched here as a six-stage design (for which preliminary engineering exists in the project repository, post-dating the paper):

1. **Provider-agnostic LLM access** — one client abstraction over multiple LLM providers with cost accounting, so reasoning steps are portable and auditable.

2. **Durable run persistence** — every selection episode recorded as a run with typed

steps, making agent behavior replayable and debuggable rather than ephemeral.

3. **Multi-turn reasoning** — replacing the single-shot hybrid re-rank (§3.3.4) with a tool-using loop: the reasoning model inspects the stress profile, queries column statistics, consults rule-based scores and prior attempts, and *commits* a recommendation through a typed tool call, with graceful degradation to the rule-based path on failure.

4. **Closed-loop execution** — the selector trains its own recommendation (subprocess-isolated, cancellable), benchmarks the result, and iterates on quality shortfall: recommend → train → evaluate → re-reason. This converts selection from a one-shot prediction into a bounded search, and — connecting to §6.1 — each iteration deposits an outcome into the learning store.

5. **Protocol surface** — extending the *existing* MCP server (paper Appendix C; §4.3) from a selection surface to a selection-plus-synthesis one. The thirteen published tools are read-and-rank only; closing the loop means adding the run lifecycle — start a training run, poll its state, cancel it — so that an external agent can invoke the whole recommend-train-evaluate cycle rather than only its first step. The protocol choice is already made; what is missing is what sits behind it.

6. **Operational hardening** — concurrency control, cancellation semantics, and a CLI for headless operation.

The evaluation question this raises is genuinely open: Chapter 5's metrics (Top-K, Spearman against a fixed oracle) assess a *ranking*, not a *policy*. An agentic selector that iterates should be evaluated on regret against the oracle *under a training budget* — closer to the framing of algorithm selection under a compute budget [Ric76] than to static ranking accuracy — and no such evaluation exists yet. Agentic operation also imports agentic *failure modes* — silent errors inside multi-step agent traces, and adversarial exploitation of tool-call provenance — both directly applicable to a selector that executes its own tool calls and

training runs, and both arguing that the §6.3 design should log provenance-complete traces from day one. Finally, if accumulated synthetic outputs ever feed back into training data for the generators themselves, recursive-training collapse must be guarded against explicitly ([SSZ24]).

6.4 Multi-Agent Integration: Synthony as a Recommendation Layer for Downstream Agents

The most speculative direction embeds Synthony as the recommendation-system layer inside a *different* agent — for instance, a table-centric question-answering agent that today fixes its tool set, table schemas, workflow type, and LLM at session start with no feedback loop. The design (worked out in a companion integration plan, with a verified-then- deferred adapter prototype) proceeds in five phases: a logging-only adapter with a null pass-through default, so absent configuration the host agent's behavior is byte-identical; consumption of ranked tools/tables; per-message workflow and model hot-swapping; an autonomous self-evaluation loop closing the feedback channel; and finally specialized sub-agents orchestrated by ranking — the phase at which the host system becomes genuinely multi-*agent* rather than multi-*table*.

The methodologically interesting obstacle is a **domain mismatch** discovered during prototyping: Synthony's recommendation surface answers "which of k synthesis models fits this dataset," while an agent host needs "which tool/table/workflow should this agent use for this message" — a different decision domain requiring its own endpoint, its own feedback schema, and its own (initially rule-based) ranking policy. Overloading the synthesis-selection endpoint would conflate two problems that share machinery but not semantics. The general lesson generalizes beyond this pairing: *selection-as-a- service transfers across decision domains only if the recommendation contract is domain-typed* — a hypothesis this integration would test.

6.5 Methodological Extensions

Three smaller extensions follow directly from limitations acknowledged in Chapter 3:

- **Stressor interactions.** The additive scoring of Eq. (2) cannot express that high cardinality *and* severe skew jointly exceed the sum of their difficulties. Higher-order interaction terms — or the Level-1 meta-learner of §6.1, which captures interactions implicitly — would test whether the interpretability cost is worth paying.

- **Requirement derivation as a learned mapping.** The threshold-binned $f : \phi(D) \to \mathbf{r}$ of §3.3.2 is hand-set; with a larger benchmark (§6.2), the bin edges themselves become calibratable parameters.

- **Specification-implementation reconciliation.** Post-publication verification against the released code identified minor discrepancies between the underlying paper's method description and the shipped implementation — chiefly in the requirement-derivation thresholds and the correlation criterion. The published results were produced by the implementation and are unaffected; reconciling the written specification with the code, and issuing an erratum where warranted, is deferred to future work. The same erratum would carry the three incorrect references in the paper's own reference list (validity-audit item K, Experiments §5.10), whose fields this thesis has already corrected against the publisher records.

- **Per-intent evaluation depth.** The privacy intent showed both the strongest discriminative signal (five distinct oracle winners across seven datasets as published; six on the recovered evaluation matrix, §5.9) and the hardest prediction problem — DP certification does not reliably predict empirical re-identification risk, as non-DP diffusion models sometimes achieve lower distance-to-closest-record through smoother density estimation. Privacy-focused selection is thus the most compelling single use case for stress-aware selection and the one demanding the most training data; a privacy-only

deep dive is a natural follow-on study. Replacing the distance-based proxy with attack-based privacy evaluation (membership inference and related measures — out of scope for this thesis) would ground the privacy oracle in adversarial reality rather than geometric proxy, likely reshuffling the privacy rankings and, with them, the calibrated privacy scale factors.

CHAPTER 7

Conclusion

7.1 What Was Asked and What Was Built

This thesis began from a diagnosis: the bottleneck in tabular data synthesis has moved from *access* to generative models to the *selection* of them. Five architecturally incommensurable families now coexist, each with regimes of genuine superiority and documented failure mechanisms, and the practitioner deciding among them has had catalogs and leaderboards but no selector. The thesis posed that gap as an instance of the classical Algorithm Selection Problem and answered it with a two-system framework:

- **Table-Synthesizers**, the generation and measurement substrate: 19 published generators behind one interface, one evaluation code path, and subprocess-isolated execution — the infrastructure that makes cross-family comparison fair and ground truth manufacturable.
- **SYNTHONY**, the selection layer: stress profiling as a synthesis-specific meta-feature representation, a measurement-grounded capability registry, intent-conditioned additive scoring, and Bayesian calibration against intent-conditioned oracles — a selector that recommends without training a single generator at inference time.

The two systems couple through a deliberately thin, auditable data contract (one registry file crosses the offline/online divide), which is what makes the selector zero-shot, its

recommendations per-dimension inspectable, and its registry extensible to new generators without benchmark reruns.

7.2 What the Evidence Showed

Three findings survive the small-sample caveats stated throughout (9 held-out pairs; conclusions rest on rank correlation and structure, not significance tests):

1. **Dataset difficulty signatures predict synthesizer efficacy.** Stress meta-features carry learnable selection signal: a meta-feature kNN reaches 0.578 held-out Spearman against the oracle, nearly double frequency-based and parametric baselines — the core hypothesis, confirmed (RQ1).

2. **Capability-based selection nearly matches oracle-informed meta-learning — without the oracle.** SYNTHONY's SF-only variant attains 0.573 Spearman and the best deployable Top-1 (0.444), within 0.005 of the kNN reference that consumes ground-truth rankings at inference time. The gap it closes is architectural, not incremental: no oracle access at test time, interpretable per-dimension reasoning, and new-generator extensibility (RQ2).

3. **Grounded measurement beats free optimization at this data scale.** Freeing all 78 parameters (joint variant) generalizes worse than tuning 18 intent weights over frozen, measurement-grounded capability scores — and the ablation shows stress profiling is the load-bearing component (its removal costs 0.14 Spearman) while intent scaling, at 12 training pairs, currently costs rather than pays. The honest reading: calibration is bounded by ground-truth volume, which is precisely why the registry-construction pipeline was built to be re-runnable (RQ3).

On the question the paper's conclusion answered about zero-shot LLM selectors, this

thesis deliberately answers nothing: the published baseline was compromised (Experiments §5.10-A), and no claim in either direction is made until it is re-run.

7.3 Limitations, Restated Once

Four boundaries qualify every claim above. The benchmark is small and composition-biased (7 moderate-sized datasets; tree-friendly regime; 33.3% majority-class solvability on the test split) — a limitation of the data, not of the split chosen from it, since §5.12 exhausts all 35 dataset-level splits and every one supports the same conclusion. The privacy oracle is a distance-based proxy, not an adversarial measurement. The coverage boundary is real: 19 models buildable, 15 registered, 10 benchmarked — a selector cannot recommend what its registry has never measured. And the additive scoring function cannot express stressor interactions. None of these is incidental; each maps to a specific, already-scoped remedy in Chapter 6.

7.4 Closing: the Distance Still to Travel

The title's preposition is the thesis's most precise claim. An Auto-Synthesizer — the synthesis counterpart of AutoML — would profile, select, execute, evaluate, and learn from its own outcomes in a closed loop. This thesis delivers the foundation deliberately: the components are built, measured against each other by one code path, and calibrated on manufactured ground truth; the selection layer is validated to near-oracle-informed quality under honest caveats; and the remaining distance — the executing, self-improving loop, its learning store, and its expanding registry — is specified in Chapter 6 at task-and-gate granularity rather than as aspiration. What remains is not a redesign. It is the loop.

APPENDIX A

Worked Case Study: Live Profile and Recommendation Captures

Captured live on 2026-07-15 over the real MCP stdio interface (tools `analyze_stress_profile`, `rank_models_rule`, `rank_models_hybrid`), against `dataset/input_data/IndianLiverPatient.csv` — the benchmark's small-sample + severe-skew case, chosen as the case-study dataset for exactly that reason. Everything below is transcribed from the actual tool responses; nothing is illustrative or hand-adjusted.

Re-capture commands (any MCP client against the registered `synthony` server):

```
analyze_stress_profile {"dataset_name": "IndianLiverPatient"}
rank_models_rule       {"dataset_profile": <profile from above>, "top_n": 5}
rank_models_hybrid     {"dataset_profile": <profile>, "method": "hybrid"}
```

A.1 Dataset Summary

A.2 Dataset-Level Stress Profile (Profile step, §3.3.1)

A.3 Per-Column Analysis (11 columns)

The five enzyme/bilirubin assay columns are precisely the severe-skew cluster — clinically expected (liver-function markers are right-skewed in patient populations) and exactly the failure surface the Long Tail stressor names.

Table A.1: Case-study dataset summary (IndianLiverPatient).

Field	Value
Dataset	IndianLiverPatient (`dataset/input_data/IndianLiverPatient.csv`)
Shape	579 rows × 11 columns
Types	9 numeric, 2 categorical (Gender, Dataset)
Missing values	0.0% in every column
Capture timestamp	2026-07-15T22:29:17

Table A.2: Dataset-level stress profile for IndianLiverPatient.

Stress dimension	Measured statistic	Threshold	Flag
Long-tail skew	max $\lvert g_1 \rvert$ = **10.512** (Aspartate_Aminotransferase)	> 2.0	**TRUE** — 5 severe columns
Cardinality complexity	max unique = 262 (Alkaline_Phosphotase)	> 500	FALSE
Zipfian concentration	not detected (max categorical top-20% ratio 0.758 < 0.8)	> 0.8	FALSE
Small-sample regime	579 rows	$< 1{,}000$	**TRUE**
Higher-order correlation	density 0.611, mean R^2 0.151	density > 0.5	**TRUE**

Table A.3: Per-column analysis for IndianLiverPatient (11 columns).

Column	Type	Unique	Skewness g_1	Difficulty (overall / skew / card / zipf)
Age	numeric	72	−0.034	1 / 0 / 1 / 0
Gender	categorical	2	— (zipf ratio 0.758)	1 / 0 / 0 / 1
Total_Bilirubin	numeric	113	**4.891**	**4** / 4 / 1 / 0
Direct_Bilirubin	numeric	80	**3.199**	3 / 3 / 1 / 0
Alkaline_Phosphotase	numeric	262	**3.754**	3 / 3 / 1 / 0
Alamine_Aminotransferase	numeric	152	**6.528**	**4** / 4 / 1 / 0
Aspartate_Aminotransferase	numeric	177	**10.512**	**4** / 4 / 1 / 0
Total_Protiens	numeric	58	−0.292	1 / 0 / 1 / 0
Albumin	numeric	40	−0.049	0 / 0 / 0 / 0
Albumin_and_Globulin_Ratio	numeric	69	0.992	2 / 2 / 1 / 0
Dataset (label)	categorical	2	— (zipf ratio 0.715)	1 / 0 / 0 / 1

A.4 Derived Requirement Vector (Analyze step, §3.3.2)

A.5 Hard Constraint Filtering (row-count gates, applied before scoring)

Half the deep-generative wing of the registry is eliminated before scoring even begins — the small-data trap operating as a hard gate.

Table A.4: Derived requirement vector for IndianLiverPatient.

Capability dimension	Required level (0–4)	Driven by
skew_handling	**4**	max $g_1 = 10.512 \geq 3.0$
cardinality_handling	0	max unique 262 < first bin edge
zipfian_handling	0	not detected
small_data	**4**	579 rows < 1,000
correlation_handling	**3**	density 0.611 > 0.5
privacy_dp	—	enters via intent scale factors only (no intent set in this capture)

Table A.5: Hard-constraint filtering: models excluded by row-count gates.

Excluded model	Reason (verbatim)
TabSyn	Dataset too small (579 < 1000)
PATECTGAN	Dataset too small (579 < 1000)
AutoDiff	Dataset too small (579 < 1000)
TabDDPM	Dataset too small (579 < 1000)
AIM	Dataset too small (579 < 1000)

Table A.6: Rule-based recommendation output for IndianLiverPatient.

Rank	Model	Confidence	small_data	correlation	skew	Headline reasoning (verbatim)	Headline warning (verbatim)
1 (recommended)	**ARF**	0.9545	4/4 ✓	4/3 ✓	2 < 4 **(!)**	"Best for small data (score 4)"	"skew handling: 2 < 4"
alt	Identity	1.0000	4/4 ✓	4/3 ✓	3	"Perfect reproduction of original data"	"No privacy (copies original data verbatim)"
alt	CART	1.0000	4/4 ✓	4/3 ✓	3	"Highest quality of any model (0.989 avg)"	"Low privacy (0.057 — close to identity)"
alt	SMOTE	1.0000	4/4 ✓	4/3 ✓	3	"Best correlation preservation of benchmarked models"	"Designed for oversampling, not full synthesis"
alt	BayesianNetwork	1.0000	4/4 ✓	3/3 ✓	3	"Very high overall quality (0.974 avg)"	"Moderate correlation preservation (0.500)"
alt	GReaT	0.9167	2 < 4 **(!)**	3/3 ✓	4/4 ✓	"Handles extreme skewness (>4.0) better than any model"	"small data: 2 < 4"

A.6 Recommendation Output (Rank step, §3.3.3) — `rank_models_rule`, method `rule_based_v2`

A.7 Hybrid-Path Degradation, Captured Live

`rank_models_hybrid` with `method: "hybrid"` was invoked in an environment with no LLM credential configured. Returned:

Table A.7: Hybrid-path degradation, captured live.

Field	Value
`method`	`"hybrid (llm unavailable - used rule_based)"`
`llm_reasoning`	`null`
Ranking	**identical** to A.6, model for model, score for score

This is the graceful-degradation contract of §3.3.4 witnessed at the real tool boundary: hybrid mode never fails a request over LLM unavailability, and its fallback is bit-identical to the deterministic rule path.

A.8 Observations from the Capture

1. **Soft thresholds doing their job.** The dataset demands skew_handling 4; the recommended ARF provides only 2, and the system says so in its own output ("**(!)** skew handling: $2 < 4$") while still recommending it — graceful partial credit, not hard constraint satisfaction. The runner-up tension is visible in the same table: GReaT is the only candidate meeting skew 4/4, but fails the small-data requirement ($2 < 4$) that ARF aces. The scoring is arbitrating exactly the stressor conflict this dataset was chosen to exhibit.

2. **Identity appears in ranked output.** Unlike the paper benchmark (which excludes Identity as trivially perfect), the MCP ranking tools return it as a 1.0-confidence alternative with its "not a real synthesizer" warning attached. The thesis case study should either present it with that caveat or filter it — flagged so the choice is explicit, not accidental.

3. **Displayed confidence is not the ranking key.** The primary recommendation (ARF, 0.9545) carries a *lower* confidence score than four alternatives at 1.0000 — primary selection uses the full capability-match scoring, while `confidence_score` is a separate quantity. Worth one clarifying sentence wherever this table appears, or a reader will think the ranking is broken.

4. **Intent-blindness of this capture.** These MCP tools take no intent/focus parameter, so this is the *unweighted* default path (α uniform). Cross-checking the oracle heatmap (`assets/fig_winner_heatmap.png`) for this dataset: the fidelity winner (SMOTE) sits in the top alternatives ✓, but the privacy winner (AIM) is **excluded by the 1,000-row gate** — a concrete instance where a hard constraint and an intent-conditioned oracle disagree, and a good discussion point for the thesis: row-count gates encode a small-data-quality prior that the privacy intent might rationally override.

5. **Capability provenance is visible in the payload.** Each model's entry carries `capabilities_source` ("spark" empirical vs. "literature" for GReaT) and its empirical preservation scores — the measurement-grounding of Methodology §3.2.4a is inspectable in every live response, which is the interpretability claim made concrete.

A.9 Second Capture: Rule vs. Agentic Comparison on abalone (downloaded variant)

Captured live 2026-07-15/16 on `abalone_downloaded` (4,177 rows × 9 columns; a header-variant copy of UCI Abalone registered into the data directory). Complementary stress signature to A.1's IndianLiverPatient: **not** small-data, but severe skew (Height, $g_1 = 3.129$) *plus* high cardinality (4 weight columns, max 2,429 unique values) — and, unlike ILP, **no row-count exclusions**: all 15 registry models pass the gates, so both paths choose over the full field.

Profile summary (from `analyze_stress_profile`):

Table A.8: Dataset-level stress profile for the abalone (downloaded) variant.

Stress dimension	Measured	Flag
Long-tail skew	max $g_1 = 3.129$ (Height only)	**TRUE**
Cardinality	4 columns > 500 unique (Whole/ Shucked/Viscera/Shell_weight; max 2,429)	**TRUE**
Zipfian	not detected	FALSE
Small-sample	4,177 rows	FALSE
Higher-order correlation	density 1.00, mean R^2 0.672	FALSE†

† See A.11 observation 4 — this flag looks inconsistent with A.2's.

The comparison. The same dataset was run through both selection paths: the deterministic rule engine (`rank_models_rule`) and the full agentic reasoning loop (run `285a8041`: 4 tool-use turns against a self-hosted vLLM `Qwen3.5-35B-A3B-FP8`, 14,179 tokens, ∼73 s of reasoning — the first fully self-hosted agentic run of this system).

Table A.9: Rule-based versus agentic selection on the same dataset.

	Rule path (`rule_based_v2`)	Agentic path (4-turn LLM loop)
Primary	**GReaT** (confidence 1.00)	**GReaT** (confidence 0.92)
#2	Identity (1.00)	CART (0.88)
#3	CART (1.00)	SMOTE (0.85)
#4	SMOTE (1.00)	— (top-3 submitted)
Requirement vector used	skew 3, cardinality 3, others 0	agent-computed weighted scores: GReaT 8.9, CART 8.0, SMOTE 8.0
Stated tie-break	— (no near-tie)	"GPU Quality Focus prioritizes GReaT as cpu_only=false"
Extra outputs	capability match + verbatim warnings	per-candidate **hyperparameter hints** (e.g. GReaT: batch 500, lr 1e-3, 100 epochs) + dataset-grounded rationales
Cost	0 tokens, <100 ms	14,179 tokens, ∼73 s, $0.00 (self-hosted; "no pricing known" fallback)
Excluded models	none (4,177 rows passes all gates)	none
`degraded`	n/a	**false** — no fallback taken

A.10 What the Comparison Shows

1. **Convergence, not divergence.** Both paths independently select GReaT and substantially agree on the runner-up set (CART, SMOTE). On a dataset whose stress profile points clearly at one capability pattern (skew + cardinality → the LLM-based generator's 4/4 + 4/4), the agentic loop *confirms* the rule anchor rather than overriding it. This is the designed relationship — the LLM is a re-ranker over grounded scores, never an unanchored chooser — observed end-to-end on live infrastructure.

2. **Where the paths genuinely differ is texture, not verdict.** The agentic output adds per-candidate hyperparameter hints, explicit weighted-score arithmetic in its reasoning ("GReaT=8.9, CART=8.0, SMOTE=8.0"), and a named tie-break policy — at a cost of ∼73 s and 14k tokens versus <100 ms and zero. The rule path's verdict is free; the agentic path's *explanation* is richer.

3. **The agent self-filtered the Identity baseline.** The rule path returns Identity as alternative #1 (confidence 1.00, with its "not a real synthesizer" warning); the agent's submitted top-3 contains only real synthesizers. Nothing in the prompt forced this — a small, observed behavioral difference in the agent's favor.

A.11 Additional Observations

1. **Confidence scales are not comparable across paths.** Rule confidences saturate at 1.00 for four models simultaneously; agentic confidences (0.92/0.88/0.85) are the LLM's own graded assessment. Any figure comparing them must say so.

2. **Oracle cross-check is only partially possible.** From the oracle heatmap (`assets/fig_winner_heatmap.png`), abalone's winners are AutoDiff (privacy) and CART (fidelity, latency). CART — both paths' top real-synthesizer alternative — is consistent with two of three intents. **GReaT itself was never in the 6-model evaluated**

subset, so the oracle cannot judge either path's primary pick: a live instance of the recommendable-but-unbenchmarked coverage boundary (Methodology §3.4.3) surfacing in practice.

3. **Requirement-bin observation — verified against the implementation (2026-07-15).** The engine's derived requirement for this dataset (skew 3 at $g_1 = 3.129$) follows the implemented thresholds, which differ in minor respects from the underlying paper's description of the mapping. This thesis documents the implementation throughout (Methodology §3.3.2); the divergence is recorded as validity item §5.10-I and its reconciliation deferred to Future Work §6.5. *(Full detail: internal erratum file, never in thesis text.)*

4. **Correlation-flag observation — verified against the implementation (2026-07-15): behavior correct.** The flag detects *dense correlation that linear fits fail to explain* — suspected nonlinear dependence — so ILP (dense, linearly weak → TRUE) and abalone (dense, linearly strong → FALSE) are both correct: abalone's structure is strong but linear, which any correlation-preserving model handles, so no elevated requirement. Methodology §3.3.2 documents these implemented semantics; the underlying paper's brief description of this criterion differs and is covered by the same §5.10-I / Future Work §6.5 remark. *(Full detail: internal erratum file.)*

REFERENCES

[BBB11] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. "Algorithms for Hyper-Parameter Optimization." In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, 2011.

[BSL23] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. "Language Models are Realistic Tabular Data Generators." In *International Conference on Learning Representations (ICLR)*, 2023.

[BTS24] André Bauer, Simon Trapp, Michael Stenger, Robert Leppich, Samuel Kounev, Mark Leznik, Kyle Chard, and Ian Foster. "Comprehensive Exploration of Synthetic Data Generation: A Survey." *arXiv preprint arXiv:2401.02524*, 2024.

[CBH02] 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.

[CIL25] Jenny Chim, Julia Ive, and Maria Liakata. "Evaluating Synthetic Data Generation from User Generated Text." *Computational Linguistics*, **51**(1):191–233, 2025.

[CWP22] Yinan Cheng, Chi-Hua Wang, Vamsi K. Potluru, Tucker Balch, and Guang Cheng. "Downstream Task-Oriented Generative Model Selections on Synthetic Data Training for Fraud Detection Models." In *Synthetic Data for AI in Finance Workshop at ICAIF*, 2022.

[DGP25] Maria F. Davila R., Sven Groen, Fabian Panse, and Wolfram Wingerath. "Navigating Tabular Data Synthesis Research: Understanding User Needs and Tool Capabilities." *ACM SIGMOD Record*, **53**(4):18–35, 2025.

[DL24] Yuntao Du and Ninghui Li. "Systematic Assessment of Tabular Data Synthesis." *arXiv preprint arXiv:2402.06806*, 2024.

[DWP25] 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**(1):37, 2025.

[EHR17] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs." *arXiv preprint arXiv:1706.02633*, 2017.

[EMS20] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. "AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data." *arXiv preprint arXiv:2003.06505*, 2020.

[FKE15] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. "Efficient and Robust Automated Machine Learning." In *Advances in Neural Information Processing Systems (NIPS)*, volume 28, 2015.

[GZW25] Jingyi Gu, Xuan Zhang, and Guiling Wang. "Beyond the Norm: A Survey of Synthetic Data Generation for Rare Events." *arXiv preprint arXiv:2506.06380*, 2025.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models." In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851, 2020.

[HME23] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. "TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second." In *International Conference on Learning Representations (ICLR)*, 2023.

[JSH22] James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N. Cohen, and Adrian Weller. "Synthetic Data – what, why and how?" *arXiv preprint arXiv:2205.03257*, 2022.

[JYS19] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. "PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees." In *International Conference on Learning Representations (ICLR)*, 2019.

[KBR23] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. "TabDDPM: Modelling Tabular Data with Diffusion Models." In *International Conference on Machine Learning (ICML)*, pp. 17564–17579, 2023.

[KRF25] 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*, **658**:131655, 2025.

[Lit93] Roderick J. A. Little. "Statistical Analysis of Masked Data." *Journal of Official Statistics*, **9**(2):407–426, 1993.

[LKL26] Xiaofeng Lin, Seungbae Kim, Zhuoya Li, Zachary DeSoto, Charles Fleming, and Guang Cheng. "ReTabSyn: Realistic Tabular Data Synthesis via Reinforcement Learning." In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

[MDS24] Junwei Ma, Apoorv Dankar, George Stein, Guangwei Yu, and Anthony Caterini. "TabPFGen: Tabular Data Generation with TabPFN." *arXiv preprint arXiv:2406.05216*, 2024.

[MGN18] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. "Which Training Methods for GANs do Actually Converge?" In *International Conference on Machine Learning (ICML)*, pp. 3481–3490, 2018.

[MMS22] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. "AIM: An Adaptive and Iterative Mechanism for Differentially Private Synthetic Data." *Proceedings of the VLDB Endowment*, **15**(11):2599–2612, 2022.

[NDT25] Mihai Nadăş, Laura Dioşan, and Andreea Tomescu. "Synthetic Data Generation Using Large Language Models: Advances in Text and Code." *IEEE Access*, **13**:134615–134633, 2025.

[New05] Mark E. J. Newman. "Power Laws, Pareto Distributions and Zipf's Law." *Contemporary Physics*, **46**(5):323–351, 2005.

[NRD16] Beata Nowok, Gillian M. Raab, and Chris Dibben. "synthpop: Bespoke Creation of Synthetic Data in R." *Journal of Statistical Software*, **74**(11):1–26, 2016.

[PAE17] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. "Semi-supervised knowledge transfer for deep learning from private training data." In *International Conference on Learning Representations*, 2017. arXiv:1610.05755.

[PNR21] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing Flows for Probabilistic Modeling and Inference." *Journal of Machine Learning Research*, **22**(57):1–64, 2021.

[PWV16] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. "The Synthetic Data Vault." In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, 2016.

[QCS23] Zhaozhi Qian, Bogdan-Constantin Cebere, and Mihaela van der Schaar. "Synthcity: Facilitating Innovative Use Cases of Synthetic Data in Different Data Modalities." *arXiv preprint arXiv:2301.07573*, 2023.

[RAR24] Hooman H. Rashidi, Samer Albahra, Brian P. Rubin, and Bo Hu. "A Novel and Fully Automated Platform for Synthetic Tabular Data Generation and Validation." *Scientific Reports*, **14**:23312, 2024.

[Rei03] Jerome P. Reiter. "Inference for Partially Synthetic, Public Use Microdata Sets." *Survey Methodology*, **29**(2):181–188, 2003.

[Rei05] Jerome P. Reiter. "Using CART to Generate Partially Synthetic Public Use Microdata." *Journal of Official Statistics*, **21**(3):441–462, 2005.

[RGS22] Adriano Rivolli, Luís P. F. Garcia, Carlos Soares, Joaquin Vanschoren, and André C. P. L. F. de Carvalho. "Meta-features for Meta-learning." *Knowledge-Based Systems*, **240**:108101, 2022.

[Ric76] John R. Rice. "The Algorithm Selection Problem." *Advances in Computers*, **15**:65–118, 1976.

[RRR03] Trivellore E. Raghunathan, Jerome P. Reiter, and Donald B. Rubin. "Multiple Imputation for Statistical Disclosure Limitation." *Journal of Official Statistics*, **19**(1):1–16, 2003.

[Rub93] Donald B. Rubin. "Discussion: Statistical Disclosure Limitation." *Journal of Official Statistics*, **9**(2):461–468, 1993.

[RVK18] Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. "A Tutorial on Thompson Sampling." *Foundations and Trends in Machine Learning*, **11**(1):1–96, 2018.

[SD23] Aivin V. Solatorio and Olivier Dupriez. "REaLTabFormer: Generating Realistic Relational and Tabular Data using Transformers." *arXiv preprint arXiv:2302.02041*, 2023.

[Skl59] Abe Sklar. "Fonctions de répartition à n dimensions et leurs marges." *Publications de l'Institut de Statistique de l'Université de Paris*, **8**:229–231, 1959.

[SLH23] Namjoon Suh, Xiaofeng Lin, Din-Yin Hsieh, Mehrdad Honarkhah, and Guang Cheng. "AutoDiff: Combining Auto-encoder and Diffusion Model for Tabular Data Synthesizing." In *NeurIPS Workshop on Synthetic Data Generation with Generative AI (SyntheticData4ML)*, 2023.

[SLN26] Hochan Son, Xiaofeng Lin, Jason Ni, and Guang Cheng. "SYNTHONY: A Stress-Aware, Intent-Conditioned Agent for Deep Tabular Generative Models Selection." In *2nd Workshop on Deep Generative Models in Machine Learning: Theories, Principles, and Efficacy (DeLTa), ICLR*, 2026.

[SOT22] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. "Synthetic Data – Anonymisation Groundhog Day." In *USENIX Security Symposium*, pp. 1451–1468, 2022.

[SSZ24] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. "AI Models Collapse When Trained on Recursively Generated Data." *Nature*, **631**:755–759, 2024.

[SWD25] 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*, 2025.

[SXH25] 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 (ICLR)*, 2025.

[Van19] Joaquin Vanschoren. "Meta-Learning." In *Automated Machine Learning: Methods, Systems, Challenges*, pp. 35–61. Springer, 2019. Survey version: arXiv:1810.03548, 2018.

[VRB13] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. "OpenML: Networked Science in Machine Learning." *ACM SIGKDD Explorations Newsletter*, **15**(2):49–60, 2013.

[WBK23] David S. Watson, Kristin Blesch, Jan Kapar, and Marvin N. Wright. "Adversarial Random Forests for Density Estimation and Generative Modeling." In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023.

[WM97] David H. Wolpert and William G. Macready. "No Free Lunch Theorems for Optimization." *IEEE Transactions on Evolutionary Computation*, **1**(1):67–82, 1997.

[XSC19] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. "Modeling Tabular data using Conditional GAN." In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pp. 7335–7343, 2019.

[YZN24] Chao Yan, Ziqi Zhang, Steve Nyemba, and Zhuohang Li. "Generating Synthetic Electronic Health Record Data Using Generative Adversarial Networks: Tutorial." *JMIR AI*, **3**:e52615, 2024.

[ZCP14] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. "PrivBayes: Private Data Release via Bayesian Networks." In *ACM SIGMOD International Conference on Management of Data*, pp. 1423–1434, 2014.

[Zhu24] Jun Zhu. "Synthetic data generation by diffusion models." *National Science Review*, **11**(8):nwae276, 2024.

[ZZS24] Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. "Mixed-Type Tabular Data Synthesis with Score-based Diffusion in Latent Space." In *International Conference on Learning Representations (ICLR)*, 2024.