

Meta-Learning for Dataset Selection: An Adaptive Utility Scoring Framework with Interpretable Meta-Feature Analysis

Faisal AL-Saqqar^{1,*}, Amjad H. Alkilani², Mohammad I. Nusir^{3,*}, Wael AlQassas⁴, and Mohammad El-Bashir⁵

^{1,4,5} Computer Science Department, Al al-Bayt University, Mafrq, Jordan.

² University of Arizona Global Campus, Chandler, Arizona, USA.

³ CBM Integrated Software Inc, San Diego, CA, USA.

* Corresponding authors: mnuseir@cbmis.com and faisalss@aabu.edu.jo. Contributing authors: w_qassas@aabu.edu.jo; mohdelb@aabu.edu.jo; Amjad.Alkilani@uagc.edu.

Received: 19 July 2026 **Accepted:** 8 August 2026 **Published:** 6 October 2026

DOI: 10.15849/ijasca.160 **License:** Open access under CC BY 4.0. Authors retain copyright.

Publisher: Al-Zaytoonah University of Jordan (ZUJ), Amman, Jordan.

Cite this article: F. AL-Saqqar, A. H. Alkilani, M. I. Nusir, W. AlQassas, and M. El-Bashir, "Meta-learning for dataset selection: an adaptive utility scoring framework with interpretable meta-feature analysis," *Int. J. Advance Soft Compu. Appl.*, vol. 18, no. 3, pp. 87–112, 2026, doi: <https://doi.org/10.15849/ijasca.160>.

ABSTRACT

Dataset selection is commonly based on dataset size, prior experience, or computationally expensive trial-and-error evaluation. We treat that choice as a meta-learning problem instead. The Adaptive Dataset Utility Score (ADUS) ranks candidate datasets without training the downstream classifier portfolio on each candidate, using a scoring function developed from a separate collection of datasets whose classifier performance has already been measured. It reads 31 meta-features covering a dataset's statistics, structure, information content, and complexity, then combines them with weights that are either set by hand or fit to past results using non-negative least squares under a nested leave-one-dataset-out protocol. Across 20 datasets tested against five classifiers, both ADUS variants ranked the true best-performing dataset first in the aggregated leave-one-dataset-out ranking, a result no baseline or learned meta-learner matched, and the default-coefficient variant tracked realised utility at a Spearman correlation of 0.665, against 0.614 for a random forest meta-learner and 0.155 for ranking on sample count. Information-theoretic and complexity-based meta-features contributed most strongly to predictive performance, with the complexity group carrying the single largest share. Unexpectedly, the bell-shaped complexity term in the default formula improved raw ranking correlation when removed, even though it also reduced the fraction of top-performing datasets recovered in the top-3 -- a genuine trade-off documented in the component ablation rather than a reason to drop the term outright.

Keywords: meta-learning; dataset selection; meta-features; classifier utility prediction; interpretable machine learning.

1 Introduction

The performance gap between two machine learning models is often dominated less by the model itself than by the characteristics of the data on which the model is trained.

Decades of empirical practice and a growing body of theoretical work agree on this point, and recent data-centric machine learning studies have made the claim more precise by showing that data quality and composition frequently explain more of the variance in downstream accuracy than architecture or hyperparameter choices [14], yet the workflow that practitioners follow to choose among candidate datasets has changed very little. A typical pipeline still consists of training every model on every dataset, evaluating each pair with cross-validation, and only then selecting which data to keep. This procedure scales poorly. When a project must screen tens or hundreds of public corpora before committing to a final benchmark, the cost of exhaustive evaluation quickly becomes prohibitive.

The present paper studies an alternative: estimating the utility of a dataset for a fixed classifier portfolio without retraining that portfolio on the candidate itself. The approach belongs to the broader family of meta-learning, in which knowledge accumulated over prior learning episodes

is used to guide new ones. In the classical formulation of [13], an algorithm selection mapping assigns a problem to the best-performing algorithm in a portfolio. Subsequent work generalised this idea to predict performance from problem descriptors, the so-called meta-features, and applied it to algorithm recommendation, hyperparameter initialisation, and warm-start ensembles. The variant studied here inverts the standard direction. Rather than fixing the dataset and ranking algorithms, we fix a portfolio of algorithms and rank datasets. This setting is most directly demonstrated by our experiments in the small-tabular-classification regime; the procurement, benchmark-construction, and transfer-learning scenarios below are motivating applications rather than settings we evaluate directly.

Two design goals motivate the framework introduced in this work. The first is interpretability. Practitioners who rely on a recommendation to justify a procurement decision require more than a scalar score. They expect a textual rationale, a breakdown of contributing factors, and a quantification of confidence. The second goal is robustness to small meta-datasets. Even comprehensive benchmark suites rarely contain more than a few dozen problems, which makes high-capacity meta-learners prone to overfitting. The framework therefore combines a compact analytical scoring function with a meta-learning predictor that is trained and validated under a strict leave-one-dataset-out protocol.

The principal artefact of the framework is an Adaptive Dataset Utility Score (ADUS). ADUS is a linear combination of five normalised quantities derived from meta-features: an information-density term, a class-separability term, a complexity term, a redundancy penalty, and a noise penalty (explicit formulas for all five are given in Section 3.3). The default coefficients are interpretable and act as a prior over what makes a dataset useful, while a non-negative least-squares (NNLS) routine learns data-driven coefficients when historical performance records are available, fit within a nested leave-one-dataset-out loop so that no held-out dataset's utility ever leaks into its own predicted score (Section 3.3.1). The NNLS step has a desirable side effect: weights of unhelpful components collapse to zero, exposing which factors actually carry signal. On our collection, the learned solution retains information, separability, and complexity, and zeros the redundancy and noise weights.

We evaluate the framework on a curated collection of 20 datasets that spans four real benchmarks from the UCI repository and scikit-learn (iris, wine, breast cancer, digits) and 16 synthetic datasets designed to vary independently along class imbalance, label noise, feature redundancy, intrinsic dimensionality, and class geometry (the complete inventory, including exact generation parameters and random seeds, is given in Appendix A). Five canonical base learners (logistic regression, random forest, SVM, multilayer perceptron, and XGBoost) supply the ground-truth utility labels through five repeated stratified 75/25 train-test splits, using mean balanced accuracy across the five classifiers as the utility target so that class-imbalanced candidates are not scored optimistically. ADUS with default analytical coefficients achieves a Spearman correlation of 0.665 and identifies the single best dataset in the aggregated leave-one-dataset-out ranking; a Random Forest meta-learner trained on all 31 meta-features reaches 0.614. A noise injection study across all 20 datasets confirms that the score degrades under Gaussian feature noise, label flipping, missingness, and random feature perturbation, with label flipping producing by far the largest and most variable degradation.

The contributions of this paper are summarised as follows.

1. A taxonomy of 31 meta-features grouped into statistical, structural, information-theoretic, and complexity descriptors, designed to be computable in time linear or at worst quadratic in

the number of samples and features, and therefore practical at screening time (complete formulas and computational cost are given in Section 3.2 and Appendix B).

2. The Adaptive Dataset Utility Score, a transparent five-term scoring function with both an interpretable default and an NNLS-learned variant whose coefficients are bounded below by zero and admit an immediate textual explanation.

3. A comprehensive empirical study on 20 heterogeneous datasets that compares ADUS against random ranking, a sample-count proxy, an information-density baseline, and four meta-learners (Random Forest, XGBoost, LightGBM, and a multilayer perceptron) under a leave-one-dataset-out protocol with five base classifiers.

4. An ablation study that decomposes the contribution of each meta-feature group and each ADUS component, showing that the complexity group is the single strongest contributor to ranking quality in this collection, and that the bell-shaped complexity term inside ADUS trades ranking correlation against top-3 retrieval quality -- an empirical limitation of the current term rather than a reason to remove it without replacement.

5. A Python implementation with a CustomTkinter graphical interface that integrates dataset ingestion, meta-feature extraction, classifier training, ADUS scoring, noise-robustness analysis, and SHAP-based explainability into a single tool, released with this paper (Data and Code Availability statement, page after Conclusion).

The remainder of the paper is organised as follows. Section 2 surveys related work on meta-learning, meta-feature engineering, and algorithm recommendation. Section 3 formalises the dataset-selection problem and develops the ADUS framework, including the meta-feature taxonomy and the NNLS coefficient-learning procedure. Section 4 describes the experimental design, datasets, base learners, and metrics. Section 5 reports the main empirical results and Section 6 contains the ablation study. Section 7 discusses limitations, implications, and threats to validity. Section 8 concludes and outlines future work.

2 Related Work

Meta-learning has matured over three decades into a recognisable subfield of machine learning. We organise the relevant literature into three threads: algorithm recommendation through meta-features, data quality assessment, and automated dataset profiling within AutoML.

2.1 Algorithm Recommendation and Meta-Features

The framework of algorithm selection introduced by [13] and revisited by [15] establishes the conceptual basis on which most modern meta-learning systems rest. The defining choice is the descriptor used to characterise a problem. Statistical meta-features such as the number of instances, number of features, class entropy, and ratios between them were collected in the influential STATLOG project [10] and remain part of every modern meta-feature library. Information-theoretic descriptors based on mutual information between features and the target were popularised by [1], who also introduced landmarks -- cheap reference classifiers whose performance is itself a feature. Complexity measures developed by [4], and later expanded into the Data Complexity Library by [8], summarise class overlap, feature efficiency, and topology. The framework proposed in this paper draws on all three families. Crucially, however, we restrict the descriptor set to features that can be computed in time at most quadratic in the sample count, which excludes pairwise distance estimators that scale poorly with the sample count.

Recent advances replace hand-crafted meta-features with learned dataset embeddings. DeepSets-style architectures [19] and permutation-equivariant set encoders have been used in Dataset2Vec [5] and similar works to obtain fixed-length vectors directly from raw rows. While these methods can capture interactions that hand-crafted statistics miss, they typically require pretraining on a large meta-dataset, which is incompatible with our setting where only 20 problems are available. The compact analytical formulation studied here therefore remains a reasonable choice for small-sample meta-learning.

2.2 Data Quality Assessment and Dataset Valuation

A parallel literature focuses on assessing data quality independently of any downstream learner. In [11] authors introduced confident learning for label-error detection, and [12] studied data dropping for noisy supervision. In [14] authors catalogued the prevalence of label noise, missingness, and selection bias in production datasets. A related but distinct thread, data valuation, assigns each training example or dataset a contribution score toward a specific model's performance, typically through Shapley-value approximations or influence functions; this line of work targets sample- or source-level attribution within a single training run rather than screening among whole candidate datasets before training starts, which is the problem ADUS addresses. Although our work shares the same motivation as the data-quality literature, it differs in scope: we predict the expected utility of a dataset rather than diagnose individual labels. The redundancy and noise terms in ADUS quantify global properties of the data, and they remain interpretable to a domain expert who must justify a decision to acquire or reject a corpus.

2.3 AutoML and Performance Prediction

Modern AutoML systems integrate meta-learning at multiple levels. Auto-sklearn [3] uses meta-features to warm-start Bayesian optimisation. In [17] researchers trained surrogate models for cross-dataset hyperparameter transfer, and OpenML [16] provides a benchmark of over 20,000 tasks tagged with classical meta-features. Closer to the present work, [2] and [18] cast performance prediction as a regression problem over meta-features. Our contribution is complementary. Instead of predicting a scalar accuracy for a single algorithm, we predict the aggregate utility of a dataset against a portfolio, and we provide a transparent scoring function that does not depend on a black-box surrogate. The default ADUS variant requires no historical data at all, which is an important property in domains where a meta-dataset is unavailable.

2.4 Positioning Relative to Prior Work

Table A summarises how ADUS relates to the four closest lines of work discussed above along the dimensions a reader is most likely to compare.

Table A. Positioning of ADUS relative to the closest prior approaches.

Approach	Target	Representation	Interpretability	Validation protocol
Auto-sklearn meta-warm-start [3]	Per-algorithm accuracy	Classical meta-features	Low (surrogate-driven search)	Held-out OpenML tasks
Dataset2Vec [5]	Learned embedding similarity	Set-encoder embedding	Low (opaque embedding)	Pretraining meta-dataset

OpenML meta-learning [16]	Per-algorithm accuracy	Classical meta- features	Medium	Cross-task holdout
Confident learning / data valuation [11,12]	Per-example label quality	Model-derived scores	Medium (example-level)	Within one training run
ADUS (this work)	Portfolio-level dataset utility	31-descriptor taxonomy + 5- term score	High (explicit formula + SHAP)	Nested leave-one- dataset-out

3 Methodology

3.1 Problem Formulation

Let $D=D_1, D_2, \dots, D_N$ denote a finite collection of candidate datasets, where each $D_i = (X_i, y_i)$ consists of a feature matrix X_i in $R^{n_i \times d_i}$ and a per-sample label vector y_i in $\{1, \dots, K_i\}$, $i.e., y_i, j \in 1, \dots, K_i$ for every observation $j = 1 \dots n_i$. Sample count n_i , feature count d_i , and class count K_i are indexed by dataset because different candidates need not share the same dimensions or label alphabet. Let $A = \{A_1, \dots, A_M\}$ be a portfolio of supervised classifiers. The realised utility of dataset D_i under a performance metric μ is defined as the mean of μ over the algorithms in A , evaluated on a held-out test split; we call this quantity the mean portfolio performance rather than a formal expectation, since no probability distribution is placed over A . The aim is to learn a function f that maps the meta-feature representation of a dataset to its predicted utility.

3.2 Meta-Feature Taxonomy

The framework extracts 31 meta-features grouped into four families. Throughout this section, n denotes the number of instances, d the number of features after one-hot encoding, and K the number of classes of the dataset under consideration (subscripts are dropped within this section for readability). All 31 formulas below are reproduced with variable names matching the released implementation in Appendix B.

3.2.1 Statistical Features (7 descriptors)

This group captures global properties of the joint feature distribution, computed from the training partition of each dataset (Section 4.3). The seven descriptors are, explicitly: (1) the mean, across features, of each feature's standard deviation; (2) the standard deviation, across features, of each feature's mean; (3) the mean absolute skewness across features; (4) the mean excess kurtosis across features; (5) the coefficient of variation, averaged across features, of each feature's standard deviation to its mean; (6) the class balance ratio, computed as the ratio of the smallest to the largest class frequency, with values close to one indicating balanced classes; and (7) the mean absolute pairwise Pearson correlation between feature columns. Features with zero variance contribute zero rather than an undefined value to the skewness and kurtosis terms.

3.2.2 Structural Features (10 descriptors)

Structural features describe the size and shape of the dataset without examining its content: the raw counts n , d , and K , the sample-to-feature ratio n/d , the number of binary features, the number of high-cardinality features, the proportion of missing entries, and the eigenvalue

spectrum of the centred feature covariance matrix summarised by the proportion of variance retained by the first principal component, the smallest k such that k principal components explain at least 90 percent of the variance, and the spectral effective rank, computed as $\exp(H)$ where H is the Shannon entropy of the normalised eigenvalue spectrum. A feature is classified as high-cardinality when its number of distinct values exceeds 10 percent of the sample count n ; this threshold is a fixed hyperparameter of the extractor rather than a data-dependent choice.

3.2.3 Information-Theoretic Features (6 descriptors).

This group quantifies the dependency between features and the target label: the mean and standard deviation of the mutual information between each feature and the label, the Shannon entropy of the label distribution, the average conditional entropy of features given the label, and the average and maximum pairwise mutual information between feature columns, which approximates feature redundancy. Feature-label mutual information uses scikit-learn's k -nearest-neighbour-based continuous estimator [`mutual_info_classif`], which avoids an explicit histogram bandwidth choice for that term; conditional entropy and pairwise feature-feature mutual information, which need an explicit joint distribution rather than a rank-based statistic, use Freedman-Diaconis histogram binning with the bin count clipped to $[2, 50]$ to bound cost and variance on small classes. Pairwise feature-feature mutual information is evaluated on at most 60 randomly sampled feature pairs per dataset (all pairs are used when d is small enough that $C(d,2)$ does not exceed 60) to keep extraction sub-quadratic in practice; on the widest dataset in the collection (80 features), full 31-feature extraction completed in under 0.3 seconds.

3.2.4 Complexity Features (8 descriptors)

Complexity descriptors approximate the difficulty of separating classes in the original feature space. The eight descriptors are: (1) the mean Fisher discriminant ratio across features, the ratio of between-class to within-class variance; (2) the maximum per-feature Fisher discriminant ratio; (3) the mean class-conditional range overlap across features and class pairs; (4) a nearest-neighbour label disagreement rate, the fraction of points whose single nearest neighbour (on a random subsample of at most 500 instances) carries a different label; (5) an intrinsic-dimensionality estimate; (6) a linear-separability landmark, the held-out accuracy of a linear discriminant classifier fit on a stratified 70/30 split of the same subsample; (7) a boundary-point ratio, the fraction of points whose five nearest neighbours include at least one point of a different class; and (8) a centroid-distance ratio, the mean pairwise distance between class centroids divided by the mean within-class spread. The intrinsic-dimensionality estimate follows the maximum-likelihood estimator of [7]: for each point, the log-ratio between the distance to its k -th and its nearer neighbours is averaged over k in $[1, 10]$ on the same 500-point subsample, giving a per-point local dimension estimate that is then averaged and clipped to $[0.5, d]$.

3.3 Adaptive Dataset Utility Score

The Adaptive Dataset Utility Score (ADUS) combines a small number of normalised quantities derived from the meta-feature vector. We denote by I , S , C , R , and N the information-density, separability, complexity, redundancy, and noise components. Each raw quantity is drawn from a single meta-feature and min-max normalised to $[0, 1]$ against a reference cohort: I from mean feature-label mutual information, S from the linear-separability landmark accuracy, R from mean pairwise feature-feature mutual information, and N from the nearest-neighbour label disagreement rate. C is obtained by first passing the intrinsic-dimensionality estimate through a Gaussian bump centred on the reference cohort's median intrinsic dimension (bandwidth set by the cohort's median absolute deviation), and only then min-max normalising the result; this is what produces the bell shape referred to throughout the paper. For the default-coefficient

variant, the reference cohort is the full 20-dataset collection; for the NNLS-learned variant (Section 3.3.1), it is the 19-dataset training fold of each leave-one-out iteration, so a held-out dataset never contributes to its own normalisation statistics.

$$ADUS = \alpha I + \beta S + \gamma C - \delta R - \varepsilon N \quad (1)$$

with non-negative coefficients $\alpha, \beta, \gamma, \delta, \varepsilon$. The first three terms are rewards. They favour datasets with strong feature-label dependency, well-separated classes, and a balanced intrinsic complexity that is neither trivial nor adversarial. The remaining two terms are penalties. R discourages datasets with excessive feature redundancy, while N penalises high label or feature noise. The default configuration uses coefficients $(\alpha, \beta, \gamma, \delta, \varepsilon) = (1.0, 1.0, 0.5, 0.8, 0.7)$, chosen so that the two terms with the most direct, well-established link to classifier accuracy in the meta-learning literature (label dependency and separability) receive full weight, the bell-shaped complexity term receives half weight to reflect that it rewards only a middle range of difficulty, and the two penalty terms are weighted below unity so that a dataset with moderate redundancy or noise is not penalised out of contention on that basis alone. These defaults are a stated prior, not a fitted optimum, and Section 6.2 reports how sensitive the ranking is to removing each term. The bell-shaped C term peaks at intermediate intrinsic dimensionality and decays toward zero at both extremes.

3.3.1 Coefficient Learning via NNLS

When a meta-dataset of realised utilities is available, the coefficients are learned by minimising the squared error between predicted and observed utility subject to non-negativity. Formally, for a training fold of components stacked as a matrix M with columns $[I, S, C, -R, -N]$ (the two penalty columns are sign-flipped so that the NNLS non-negativity constraint on θ corresponds to non-negative α, ε in Eq. 1, and no intercept column is included, so $ADUS$ is constrained to pass through the origin when all components are zero) and a vector u of realised utilities, we solve

$$\min_{\theta} \|M\theta - u\|^2 \text{ subject to } \theta \geq 0 \quad (2)$$

using the active-set NNLS algorithm of [6]. To avoid the held-out dataset's utility leaking into its own prediction, coefficient fitting is nested inside the leave-one-dataset-out loop: for each held-out dataset, both the normalisation statistics used to build M and θ itself are computed from the remaining 19 datasets only, and the learned θ is then applied to the held-out dataset's own (training-fold-normalised) components. Repeating this for all 20 datasets yields 20 out-of-sample coefficient vectors rather than a single in-sample fit; Section 5.3 reports their mean and standard deviation as a direct measure of how stable the learned weights are on a collection this size. The procedure carries an important property: components that fail to correlate with realised utility receive a zero weight, which provides an automatic feature-selection mechanism over the components of $ADUS$. On our collection this consistently zeros the redundancy and noise weights across all 20 folds; Section 5.3 reports the full distribution.

3.4 Meta-Learning Model

To benchmark $ADUS$ against a fully data-driven alternative, we train a regression meta-learner that maps the full 31-dimensional meta-feature vector to predicted mean balanced accuracy. Four predictors are evaluated: random forest, gradient-boosted trees (XGBoost), LightGBM, and a small multilayer perceptron. All models use the same input representation after standard scaling fit on the training fold only. Random forest tree count is selected by a dedicated nested-LODO sweep over $\{50, 100, 150, 200, 300, 400, 500\}$ trees, reported as an exploratory analysis in Section 6.3 rather than folded into the main comparison; 300 trees minimised both mean

absolute error and maximised Spearman correlation in that sweep and is used as the operational default throughout the rest of the paper. XGBoost and LightGBM use 200 shallow estimators (max depth 3, learning rate 0.05) chosen as reasonable defaults for a 19-example training fold rather than tuned against the same data used for evaluation. Validation follows a strict leave-one-dataset-out (LODO) protocol: in each fold, one dataset is held out, the meta-learner is fit on the remaining 19 examples, and a single prediction is produced for the held-out problem. This protocol is the most pessimistic available in the small-sample regime, and it directly measures the meta-learner's ability to generalise to a previously unseen data distribution.

3.5 Noise Robustness Analysis

Real-world datasets rarely satisfy the assumptions made by complexity measures. To probe the resilience of ADUS, we inject four kinds of noise at six levels each (5%, 10%, 15%, 20%, 30%, 40%): additive Gaussian noise on standardised features, random label flipping, random missing-value insertion followed by median imputation, and feature perturbation by random permutation of a level-proportional number of whole columns. The sweep is run on all 20 datasets in the collection, not a hand-selected subset, and every (dataset, noise type, level) cell is repeated over 5 independent corruption seeds so that reported decay figures carry a standard deviation rather than reflecting one arbitrary realisation. The robustness score is defined explicitly as

$$\text{decay}(d, t, l) = \frac{|\text{ADUS}_{\text{clean}}(d) - \text{ADUS}_{\text{noisy}}(d, t, l)|}{\max(|\text{ADUS}_{\text{clean}}(d)|, 0.05)} \quad (3)$$

$$\text{robustness}(d) = 1 - \underset{t, l}{\text{mean}} \left[\underset{\text{seed}}{\text{mean decay}}(d, t, l, \text{seed}) \right] \quad (4)$$

where the 0.05 floor in the denominator prevents a near-zero clean score from producing an unbounded relative decay. For each (dataset, noise type, level, seed) combination we recompute meta-features on the corrupted data, recompute the ADUS score using the clean cohort's normalisation statistics, and record the relative change against that dataset's own clean score.

3.6 Explainability

Every ADUS recommendation is accompanied by a structured explanation. The explanation has three parts. First, a permutation-importance ranking attributes share of variance in the predicted utility to individual meta-features, computed within each leave-one-dataset-out training fold separately and averaged across folds so that the ranking reflects stability across resamples rather than a single unstable fit on 20 points (Section 5.6). Second, a SHAP analysis [9], implemented with a TreeExplainer for tree-based meta-learners and a KernelExplainer fallback for neural surrogates, produces per-dataset additive attributions; a worked example for the breast_cancer dataset is given in Section 5.6. Third, a templated textual narrative summarises the verdict, highlights the three most influential meta-features, and reports the ADUS components in human-readable form. The narrative is generated deterministically by a rule-based template that consumes the SHAP attributions and the component vector, ensuring that explanations are reproducible across runs.

3.7 Implementation

The complete system is implemented as a Python package organised into nine modules: configuration, dataset loading, meta-feature extraction, base-model training, meta-dataset construction, meta-learning, ADUS, explainability, noise robustness, and visualisation. A CustomTkinter desktop application exposes the pipeline through seven tabs that mirror the analytical workflow, with threaded execution and a live log panel for long-running operations.

Full 31-feature extraction over the entire 20-dataset collection completes in under 4 seconds on a single CPU core; the nested-LODO meta-learner comparison of Section 5.2 completes in under 10 seconds per model. Installation and usage instructions, along with the exact package versions used to produce every number in this paper, are provided with the released code (Data and Code Availability statement). Reproducibility is supported by a global random seed and by serialising every intermediate artefact (base-learner results, meta-features, predictions, robustness curves) to CSV.

4 Experimental Setup

4.1 Datasets

A collection of 20 datasets is used in all experiments. Four of these are widely used benchmarks distributed with scikit-learn or sourced from the UCI repository: the Iris dataset (150 samples, 4 features, 3 classes), Wine recognition (178 samples, 13 features, 3 classes), Wisconsin Breast Cancer Diagnostic (569 samples, 30 features, 2 classes), and the optical digit-recognition dataset Digits (1,797 samples, 64 features, 10 classes). The remaining 16 datasets are synthetically generated with scikit-learn under a single fixed generation seed (42), documented per dataset in Appendix A together with every generator argument, and are explicitly designed to vary one factor at a time. We return to single-seed generation as a limitation in Section 7.4. and are explicitly designed to vary one factor at a time. They include linearly separable two-class problems (`synth_easy_2class`), non-linear two-class problems (`synth_moons`, `synth_circles`, `synth_gaussian_quantiles`), multi-class problems with three to six classes (`synth_blobs_3class`, `synth_many_classes`, `synth_moderate_5class`), imbalanced configurations with a 92-to-8 minority ratio (`synth_imbalanced`) and a milder 65-to-35 split (`synth_mild_imbalance`), high-dimensional small-sample regimes (`synth_highdim_smalln`, 200 samples and 80 features), redundant feature spaces (`synth_redundant`), label noise (`synth_noisy_labels`, 20 percent flip), and feature noise (`synth_noisy_features`, 40 features of which 6 are informative), in addition to a 3,000-sample clean case (`synth_large_clean`) and a tiny 120-sample case (`synth_tiny`), plus a high-informative-ratio control case (`synth_high_signal_baseline`) added to round the synthetic collection out to 16 configurations. The full inventory with exact sample counts, feature counts, class counts, and generator arguments for every one of the 20 datasets is given in Appendix A. All datasets are stored as CSV files with a column named `target`. The deliberate variety is intended to expose ADUS to qualitatively different difficulty profiles within a controlled budget.

4.2 Base Learners and Training Protocol

Each dataset is processed by a uniform pipeline that performs median imputation of missing values, one-hot encoding of categorical features, and standardisation of numerical features, with the imputer and scaler fit on the training partition only. Utility is estimated from 5 independent stratified 75/25 train-test splits (seeds 42-46); we report the mean over these 5 repeats rather than a single split so that one unlucky split cannot dominate the reported number. Five canonical classifiers are then trained on the training split: logistic regression with L2 regularisation, a random forest with 100 trees, an RBF-kernel support vector machine with probability calibration via Platt scaling as implemented by scikit-learn's default `SVC(probability=True)`, which fits an internal 5-fold calibration during training, a multilayer perceptron with two hidden layers of 64 units each (ReLU activations, Adam optimiser, initial learning rate 1e-3, batch size 32, up to 500 epochs with early stopping after 15 epochs without improvement), and XGBoost with 300 trees and a learning rate of 0.1. Every model is evaluated on the held-out test split,

and we record balanced accuracy -- rather than raw accuracy, so that the 92:8 synth_imbalanced dataset is not scored optimistically by a majority-class classifier -- macro-averaged precision, recall, F1, and one-versus-rest AUC, as well as the generalisation gap defined as the difference between training and test balanced accuracy. The mean test balanced accuracy across the five classifiers and five repeated splits is used as the dataset utility target.

4.3 Meta-Feature Computation and Leakage Control

Meta-features are computed exclusively from the training partition of the first repeated split (seed 42), never from the test partition used to derive the utility target. This is a deliberate change from computing meta-features on the full dataset: several of the 31 descriptors are label-dependent (mutual information, conditional entropy, Fisher ratios, the linear-separability landmark), and computing them from data that includes the labels later used to define the utility target would let information about the test partition leak into the predictor's inputs. Structural descriptors that depend on sample count (n , n/d) are computed from this same training partition for consistency, so they reflect the training-partition size rather than the full dataset size. The 31 features are concatenated into a single row of a meta-dataset that contains 20 rows in total. For the meta-learning experiments the meta-dataset is paired with the utility target and used as the input to the meta-learner.

4.4 Evaluation Metrics

Ranking quality is evaluated with two families of metrics that answer different questions and are not required to move together. The Spearman rank correlation and Kendall tau measure overall agreement between predicted and realised rankings, computed once over the aggregated leave-one-dataset-out predictions for all 20 datasets. Top-k retrieval is reported as two distinct statistics: top1_hit is 1 if the dataset with the single highest realised utility is also ranked first by the predictor in that same aggregated ranking, and 0 otherwise -- by construction this is monotonic with any larger k , since an item ranked first is necessarily within any top- k window. top-k SET OVERLAP, reported for k in $\{3, 5\}$, is the fraction of the k genuinely best datasets that also appear in the predictor's own top- k , $|\text{predicted_top_k} \cap \text{actual_top_k}| / k$; this answers a different question (whether the whole top- k set is recovered, not whether one specific item is retrieved) and is not required to be monotonic in k . Mean absolute error, root mean squared error, and R-squared of the predicted utility are reported for the four meta-learners and, separately, for the two ADUS variants; because ADUS is a ranking score rather than a calibrated point predictor of accuracy, its MAE/RMSE/R-squared values are on a different scale from the meta-learners' and should be read only as an internal sanity check, not as a calibration claim. All metrics are computed under the leave-one-dataset-out protocol described in Section 3.4.

5 Results

5.1 Base-Learner Performance

Table 1 reports the test balanced accuracy of each base learner aggregated across all 20 datasets and five repeated splits. The support vector machine produces the highest mean balanced accuracy at 0.926, ahead of random forest at 0.908 and XGBoost at 0.903. Logistic regression and the multilayer perceptron trail at 0.842 and 0.871 respectively. A Friedman test across the five classifiers (paired by dataset, $n=20$) rejects equal performance (chi-squared = 36.40, $p < 0.001$); Holm-corrected pairwise Wilcoxon tests show the SVM and random forest are each significantly stronger than logistic regression and the MLP ($p_{\text{holm}} < 0.02$ in every case), while the random forest-versus-SVM and random forest-versus-XGBoost differences do not reach

significance after correction ($p_{\text{holm}} = 0.063$ and 0.494). The spread of 8.4 percentage points between the best and worst learner is large enough that the utility target reflects genuine cross-algorithm signal rather than a single dominant model. We also observe substantial variance across datasets: the easiest configuration, `synth_blobs_3class`, reaches a mean balanced accuracy of 0.999, while the hardest, the 92:8 imbalanced `synth_imbalanced`, averages only 0.715 once accuracy is corrected for class imbalance.

Table 1. Mean test balanced accuracy of each base learner across the 20-dataset collection, averaged over 5 repeated stratified 75/25 splits (seeds 42-46). Friedman $\chi^2=36.40$, $p<0.001$; see Section 5.1 for pairwise significance.

Base learner	Mean balanced accuracy	Mean F1	Mean AUC	Mean gap
Support Vector Machine	0.926	0.930	0.974	0.039
Random Forest	0.908	0.911	0.964	0.093
XGBoost	0.903	0.906	0.962	0.097
Multilayer Perceptron	0.871	0.874	0.942	0.052
Logistic Regression	0.842	0.846	0.901	0.037

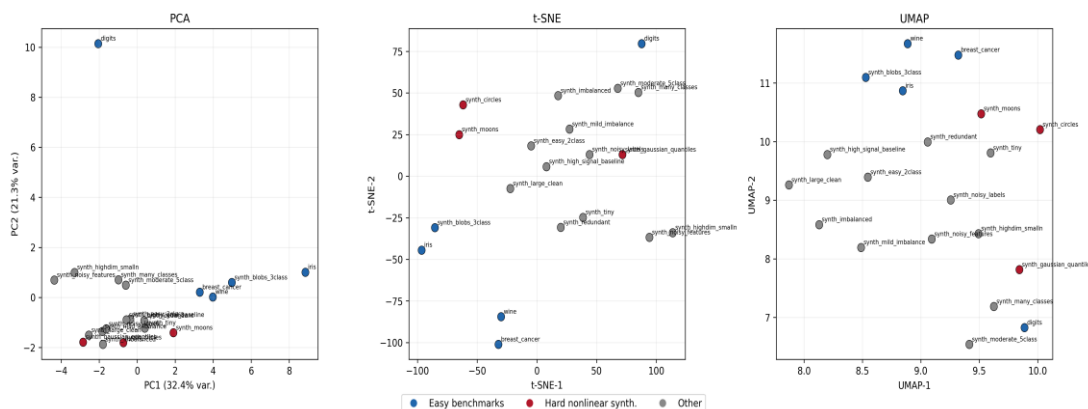


Figure 1. Two-dimensional projections (PCA, t-SNE, UMAP) of the 20-dataset collection in meta-feature space.

All three projections in Figure 1 place the easy benchmarks (`digits`, `breast_cancer`, `wine`, `iris`, `synth_blobs_3class`) close together, while the deliberately hard non-linear problems (`synth_gaussian_quantiles`, `synth_circles`, `synth_moons`) separate into a distinct region. Because PCA axes are linear combinations of all 31 standardised meta-features rather than any single named quantity, we report the leading PC1 loadings directly rather than attributing the separation to one property: the six largest-magnitude loadings on PC1 belong to `centroid_distance_ratio`, `mean_feature_correlation`, `mean_mutual_info`, `overlap_region_mean` (negative), `pca_var_ratio_pc1`, and `fisher_ratio_mean`, so the axis that separates easy from hard datasets is best read as a mixture of separability, redundancy, and information-density signal rather than any one meta-feature alone. This is a useful sanity check: the meta-feature representation captures the qualitative gradient of dataset difficulty without explicit supervision.

5.2 Meta-Learner Performance Under Leave-One-Dataset-Out

Table 2 reports the nested LODO performance of the four meta-learners on the training-partition-only, leakage-controlled meta-dataset. Random forest is the strongest of the four, achieving Spearman 0.614, Kendall tau 0.463, and the only positive R-squared in the comparison (0.333); LightGBM follows at Spearman 0.577 with a smaller positive R-squared of 0.135. XGBoost trails at Spearman 0.398 with a negative R-squared, indicating its predictions preserve some ordinal information without being well-calibrated in absolute terms. The multilayer perceptron performs worst by a wide margin: Spearman -0.068 and a strongly negative R-squared of -3401, which indicates substantial out-of-sample prediction error and poor calibration rather than any useful ordinal signal, consistent with a small network overfitting a 19-example training fold. Calibration scatter plots for the two strongest models, random forest and LightGBM, are shown in Figure 2.

Table 2. Leave-one-dataset-out performance of meta-learners on predicting mean balanced test accuracy. Bold indicates the best score in each column (Random Forest on every column).

Model	MAE	RMSE	R ²	Spearman	Kendall
Random Forest	0.052	0.067	0.333	0.614	0.463
LightGBM	0.063	0.076	0.135	0.577	0.410
XGBoost	0.070	0.089	-0.175	0.398	0.305
Multilayer Perceptron	2.488	4.790	-3401.11	-0.068	-0.063

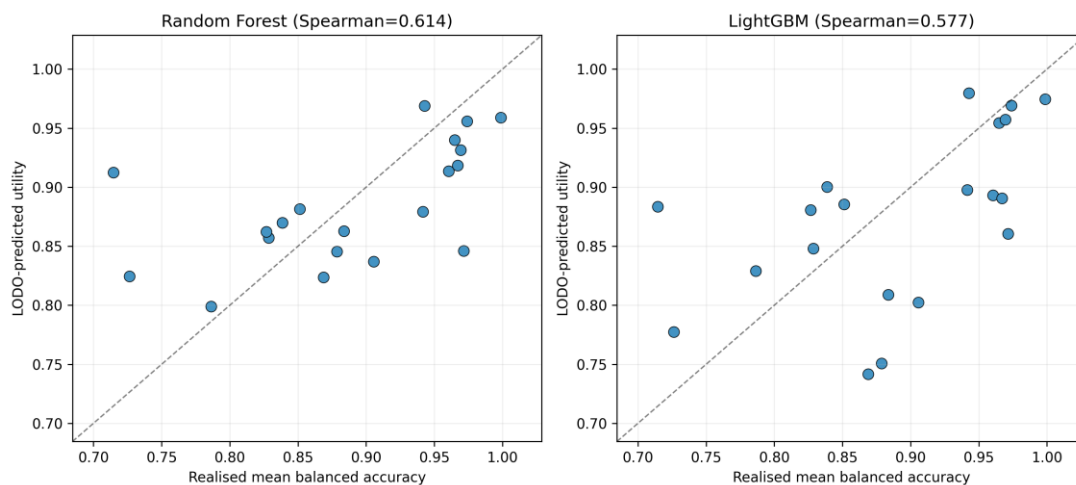


Figure 2. Predicted versus realised mean balanced test accuracy under the leave-one-dataset-out protocol, for the two best-performing meta-learners (Random Forest and LightGBM).

The modest R-squared of the best meta-learner is consistent with the small size of the meta-dataset and with the heterogeneity of the underlying problems. With 19 training points per fold and 31 candidate descriptors, the regression problem is fundamentally underdetermined, and any non-linear surrogate is at risk of overfitting. This observation motivates the learner analytical scoring approach pursued by ADUS.

5.3 ADUS Ranking Quality

Figure 3 displays the ranking induced by ADUS with NNLS-learned coefficients, aggregated across the nested leave-one-dataset-out folds. The top ranks are occupied, in order, by synth_blobs_3class, iris, and wine, all of which realise mean balanced accuracy above 0.94; the

single best dataset by realised utility, `synth_blobs_3class`, is also ranked first by ADUS in both variants (`top1_hit` = 1 for both, matched by no baseline or meta-learner in Table 3). The ranking is not uniformly sensible, however: `synth_imbalanced`, which realises the LOWEST utility of any dataset in the collection (0.715), is ranked 4th by the learned score, and `digits` (realised utility 0.970) is ranked only 12th. We read this as a genuine limit of the current component set rather than an artefact to explain away -- the learned score rewards separability and information density even when a dataset's difficulty stems primarily from class imbalance, a factor no component currently measures directly. The lowest ranks are assigned to `synth_gaussian_quantiles` and `synth_circles`, two of the three deliberately difficult non-linear problems. The NNLS procedure, refit independently inside each of the 20 leave-one-out folds, converges to a consistent solution: mean coefficients $(\alpha, \beta, \gamma, \delta, \epsilon) = (0.217 \pm 0.067, 0.847 \pm 0.041, 0.233 \pm 0.042, 0.000 \pm 0.000, 0.000 \pm 0.000)$, retaining information, separability, and complexity while zeroing redundancy and noise in every single fold. Separability carries the largest and most stable weight, a different picture from the default coefficients' equal weighting of information and separability; Section 6.2 revisits this in a direct component ablation.

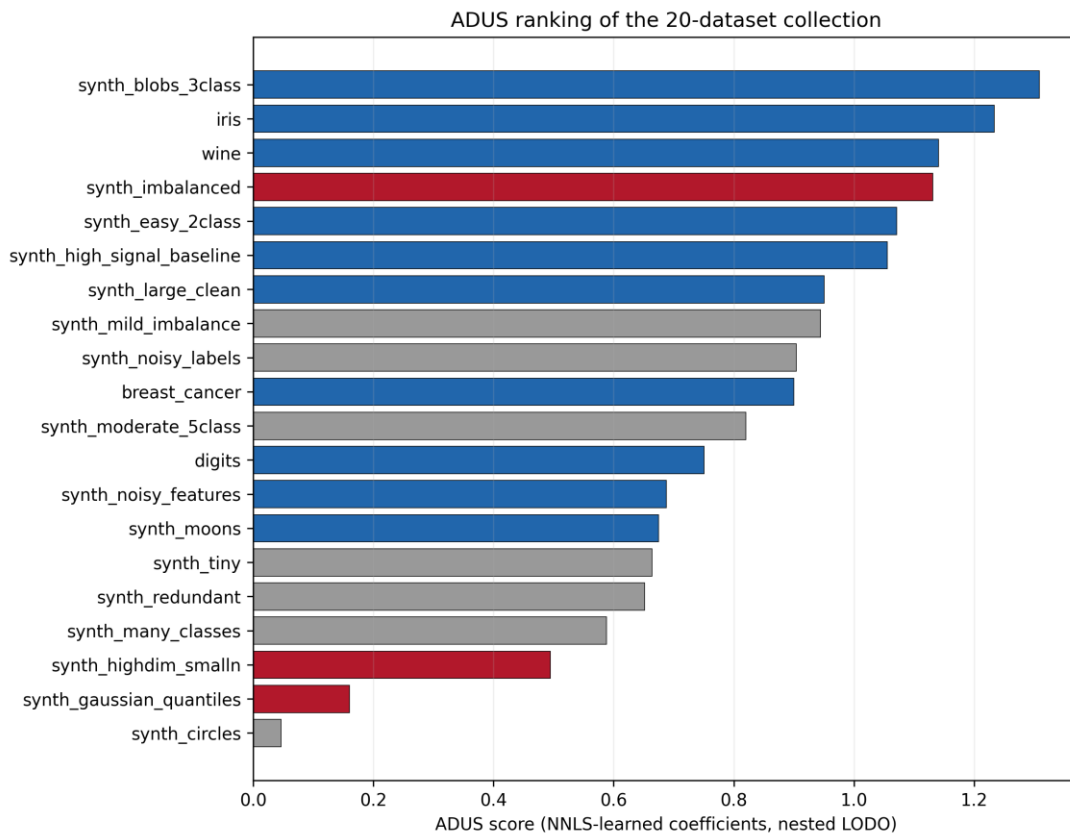


Figure 3. ADUS ranking of the 20-dataset collection using NNLS-learned coefficients (nested leave-one-dataset-out).

5.4 Comparison with Baselines

Table 3 contrasts ADUS with a random baseline, a sample-count proxy, a single-feature information-density baseline, and the four meta-learners, all evaluated with the `top1_hit` / `top-k SET OVERLAP` distinction defined in Section 4.4. A random baseline, evaluated as the average over 2,000 random permutations of the true ranking, yields Spearman -0.003 and a `top1_hit` rate of 0.054, matching the analytic expectation of $1/20 = 0.05$ for a uniformly random ranking of 20 items and serving as a sanity check on the evaluation code. Ranking by raw sample count improves only marginally to Spearman 0.155. A single information-theoretic

feature, mean mutual information, attains Spearman 0.534. ADUS with the default analytical coefficients reaches Spearman 0.665, the strongest correlation in the comparison, and is the only predictor together with ADUS-learned to achieve $\text{top1_hit} = 1$; it also achieves the highest top-3 set overlap (0.667, tied with the Random Forest and LightGBM meta-learners). ADUS with NNLS-learned coefficients trades away some of this ranking correlation (Spearman 0.558) and top-3 overlap (0.333) relative to the default coefficients on this collection, while still recovering the single best dataset. A paired bootstrap over the 20 datasets (5,000 resamples) puts the default-minus-learned Spearman difference at a 95% CI of $[-0.065, 0.317]$, which includes zero, so this difference should be read as suggestive rather than conclusively established on a collection this size (Section 7.4). The Random Forest meta-learner, despite consuming the entire meta-feature vector, attains Spearman 0.614 with a top1_hit rate of zero - it never placed the single truly best dataset first, even though its top-3 set overlap (0.667) matches ADUS-default. The comparison is visualised in Figure 4, split into a correlation panel and a retrieval panel so that metrics with different scales and interpretations are not read off the same axis.

Table 3. Ranking quality of dataset-utility predictors on the 20-dataset collection.

Predictor	Spearman	Kendall	Top-1 hit	Top-3 overlap	Top-5 overlap
Random baseline (MC avg.)	-0.003	-0.003	0.054	0.153	0.248
Sample-count proxy	0.155	0.100	0.000	0.333	0.400
Information density	0.534	0.379	0.000	0.333	0.200
ADUS (default)	0.665	0.568	1.000	0.667	0.600
ADUS (learned)	0.558	0.432	1.000	0.333	0.400
Meta-learner (Random Forest)	0.614	0.463	0.000	0.667	0.600
Meta-learner (XGBoost)	0.398	0.305	0.000	0.667	0.400
Meta-learner (LightGBM)	0.577	0.410	0.000	0.667	0.600
Meta-learner (MLP)	-0.068	-0.063	0.000	0.000	0.000

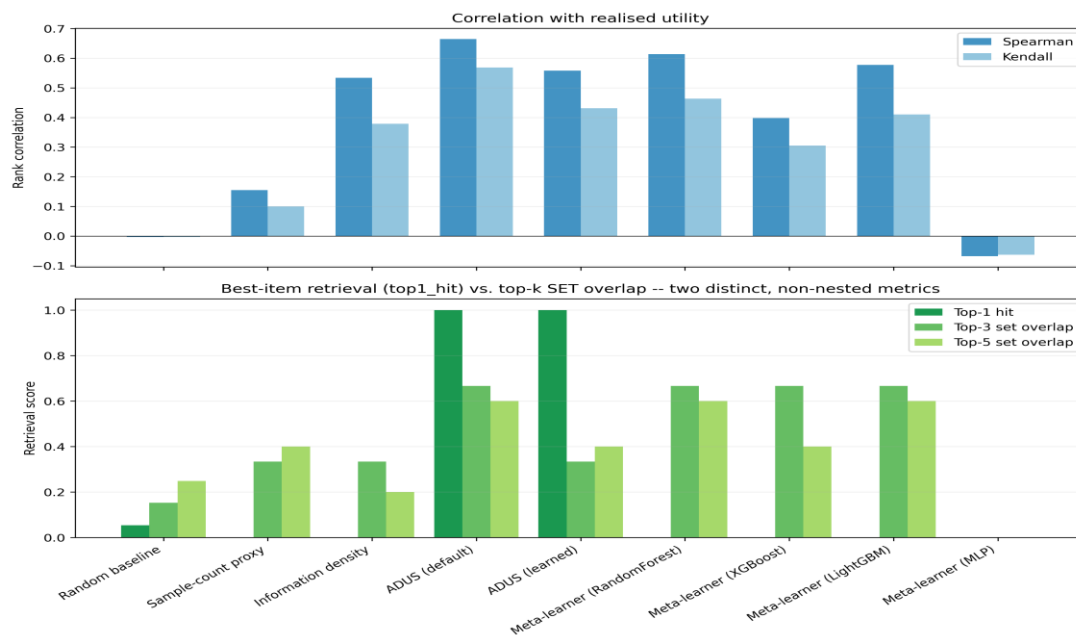


Figure 4. Comparison of dataset-utility predictors, correlation metrics and retrieval metrics shown in separate panels.

5.5 Noise Robustness

Figure 5 reports the noise-robustness curves for all 20 datasets (not a hand-selected subset) across the four noise families defined in Section 3.5, with 5 corruption seeds per cell. Three patterns emerge. First, Gaussian feature noise has the mildest average effect: mean relative decay of 0.244 (SD 0.353) across all levels and datasets, rising from 0.199 at the 5% level to 0.392 at the 40% level. Second, label flipping induces by far the largest and most variable decay, with a mean of 1.287 (SD 1.610) across levels, reaching a mean of 1.985 at the 40% flip level; several datasets' scores change sign under heavy label corruption, which the 0.05 floor in Eq. 3 keeps finite but does not prevent from exceeding 100% relative decay. Third, per-dataset robustness (Eq. 4) is highly uneven: most datasets score between 0.4 and 0.8, but *synth_circles* (-0.580), *synth_highdim_smalln* (-2.217), and *synth_tiny* (-0.042) show negative robustness, meaning their ADUS score is, on average, more volatile under injected noise than the clean score's own magnitude. These are exactly the smallest and/or most nonlinearly-structured datasets in the collection, and we read this as evidence that the current normalisation is least stable in the low-sample, low-density regime rather than as a property of ADUS generally. Aggregating across all sweeps yields an overall mean robustness score of 0.396 with a standard deviation of 0.700 across datasets -- a substantially less tidy picture than a single clean average would suggest, which is the point of reporting the full distribution rather than a headline number alone.

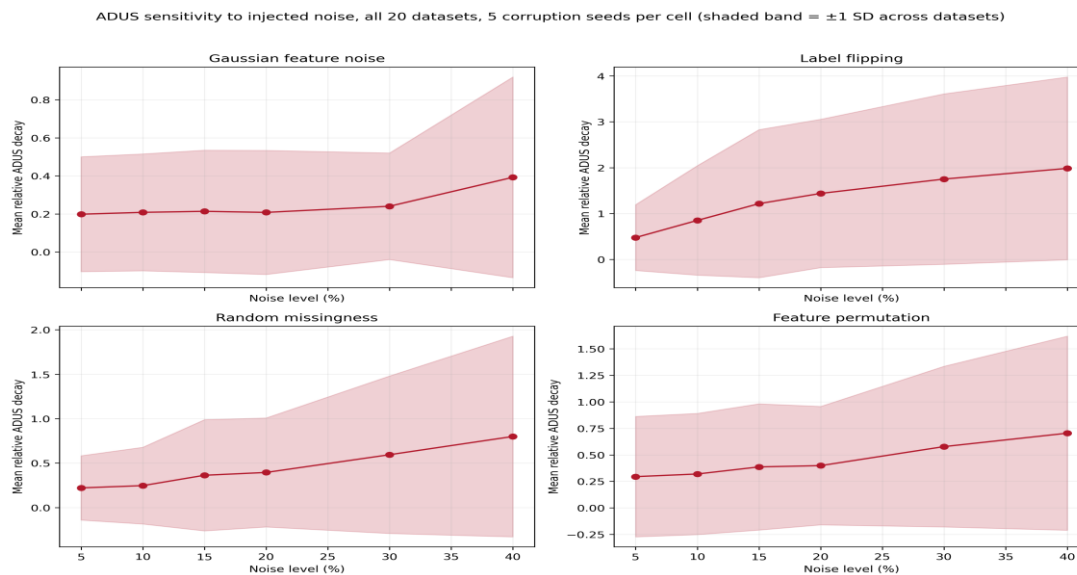


Figure 5. Sensitivity of ADUS to four families of injected noise across all 20 datasets and 5 corruption seeds per cell; shaded band is ± 1 SD across datasets.

5.6 Explainability Outputs

For each dataset, the explainability module produces three artefacts: a permutation importance plot of the meta-features used by the meta-learner (Figure 6), a SHAP attribution vector for the predicted utility, and a textual narrative. Permutation importance, computed within each leave-one-dataset-out training fold and averaged across all 20 folds, identifies *fisher_ratio_max* (a complexity descriptor), *std_feature_mean* (a statistical descriptor), and *mean_mutual_info* (an information-theoretic descriptor) as the three most influential meta-features, together accounting for approximately 54% of the total permutation importance mass. This is consistent with the group ablation of Section 6.1, where the complexity group alone outperforms every other single group, and it revises our reading of Section 5.4's baseline comparison: the strong showing of the single-feature information-density baseline (Spearman 0.534) reflects only part of the signal the full meta-learner exploits. As a worked example, for the *breast_cancer* dataset (realised utility 0.965, meta-learner prediction 0.960) the largest SHAP contributions come from *fisher_ratio_max* (+0.021), *std_feature_mean* (+0.017), and *linear_separability_lda* (+0.011), each pushing the predicted utility above the meta-dataset's mean -- a concrete instance of the same three-feature pattern seen in the aggregate importance ranking. SHAP attributions are computed once per meta-learner fit on the full 20-dataset meta-dataset for this illustrative example rather than averaged across LODO folds; production use of the explainability module retains the fold-averaged importance ranking for stability. Because both analyses are computed on a meta-model trained on only 20 points, the specific numerical attributions should be read as illustrative of which descriptors dominate rather than as precise, low-variance estimates.

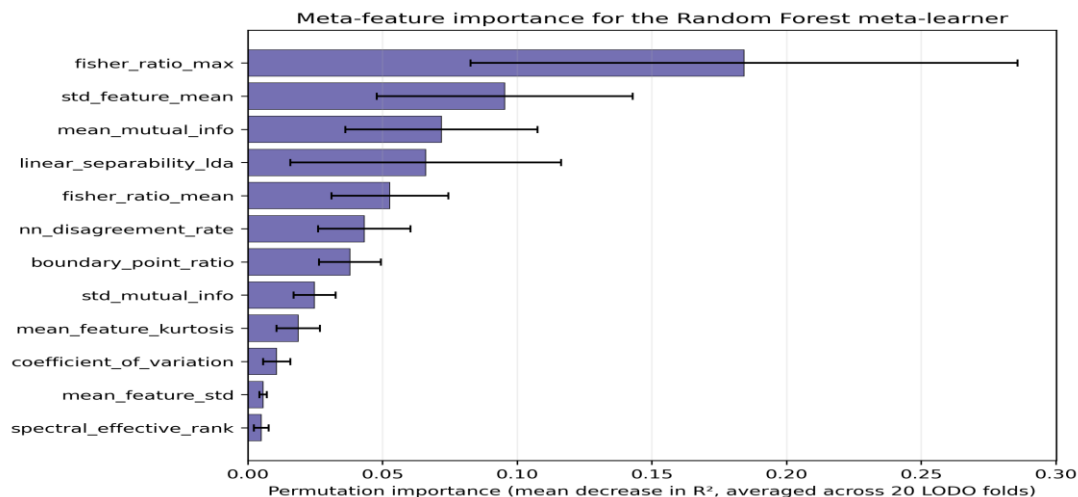


Figure 6. Permutation feature importance for the random forest meta-learner, averaged across leave-one-dataset-out folds. Feature names match the formal taxonomy of Section 3.2 and Appendix B.

6 Ablation Study

The ablation study probes three questions. First, which meta-feature group carries the predictive signal. Second, which components of the ADUS formula are essential, evaluated on both the default and the NNLS-learned coefficients so that the finding is not an artefact of coefficients that happened to be zeroed already. Third, how sensitive the meta-learner is to the number of trees. Each ablation is conducted under the same leave-one-dataset-out protocol used in Section 5.

6.1 Meta-Feature Groups

Table 4 reports the Spearman correlation of the random forest meta-learner when restricted to a single group of meta-features (only_*) and when one group at a time is removed (drop_*). The complexity group is the standout single group, achieving Spearman 0.696 on its own -- higher than the full 31-feature baseline (0.614) -- and its removal is the single most damaging ablation, dropping correlation to 0.400. Information-theoretic features are the second-strongest group in isolation (0.540) and the second-most damaging to remove (0.531). Statistical features perform moderately in isolation (0.451) but their removal barely affects the full model (0.605), suggesting redundancy with the other groups. Structural features perform worse than the random baseline in isolation (-0.110), and their removal has the smallest effect of all four groups (0.597), which is consistent with the permutation-importance finding of Section 5.6: none of the top-ranked features by importance came from the structural group. Two implications follow. First, the analytical ADUS formulation should prioritise complexity and information descriptors, which it does by design, though the present results argue for weighting complexity at least as heavily as information rather than at half its weight as the current default coefficients do (Section 3.3). Second, adding statistical and structural features to a meta-learner provides at best marginal benefit and at worst dilutes the signal carried by the strong groups. The meta-feature group ablation results are visualised in Figure 7.

Table 4. Meta-feature group ablation under leave-one-dataset-out (Random Forest, 300 trees).

Setting	# Features	MAE	Spearman	Kendall
All features (baseline)	31	0.052	0.614	0.463
only_statistical	7	0.068	0.451	0.274

only_structural	10	0.080	-0.110	-0.032
only_informatio n	6	0.060	0.540	0.368
only_complexity	8	0.045	0.696	0.558
drop_statistical	24	0.047	0.605	0.474
drop_structural	21	0.051	0.597	0.442
drop_informatio n	25	0.054	0.531	0.421
drop_complexity	23	0.065	0.400	0.274

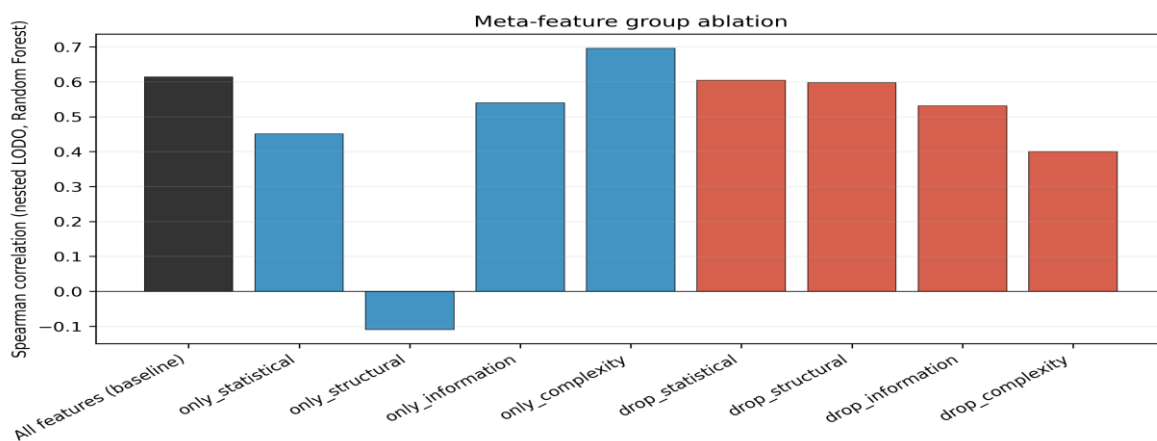


Figure 7. Spearman correlation of the random forest meta-learner under the meta-feature group ablation.

6.2 ADUS Components

Table 5 dissects the ADUS formula by removing one term at a time, separately for the default coefficients and for the NNLS-learned coefficients (the latter re-fit with the corresponding coefficient held at zero, still under the nested leave-one-dataset-out protocol, so the comparison is not predetermined by an already-zero weight). Two findings deserve attention. First, removing the information-density component degrades Spearman correlation in both variants - - from 0.665 to 0.468 for the default coefficients and from 0.558 to 0.483 for the learned coefficients -- confirming that *I* is a load-bearing contributor throughout. Second, the two variants disagree about which component is most essential: for the default coefficients, removing separability (*S*) has almost no effect on Spearman (0.665 to 0.595) but pushes top-3 overlap to a perfect 1.000, while for the learned coefficients removing separability is the single most damaging ablation (0.558 to 0.332), consistent with separability carrying the largest learned weight in Section 5.3. Third, and most interesting, removing the complexity component from the default coefficients IMPROVES Spearman correlation, from 0.665 to 0.795 -- the single largest change in either direction in this table -- while simultaneously reducing top-3 overlap from 0.667 to 0.333. This is a genuine trade-off rather than a reason to drop the term outright: the complexity term in the present implementation is bell-shaped over intrinsic dimensionality, peaking at moderate values and falling to zero at both extremes. The two-feature datasets *synth_moons* and *synth_circles* therefore receive a near-zero complexity contribution despite being non-trivially classifiable by a non-linear model, which systematically depresses their score and, on this collection, happens to hurt the overall rank correlation more than it helps the top-3 retrieval objective. Future work should replace the bell-shaped term with

a monotone proxy or a learned non-linear transform, and should re-run this same nested ablation to confirm whether the trade-off persists. The corresponding ADUS component-ablation results are illustrated in Figure 8.

Table 5. ADUS component ablation, default and NNLS-learned coefficients

Configuration	Spearman	Kendall	Top-3 overlap
Full ADUS (default)	0.665	0.568	0.667
Default -- Drop I (information)	0.468	0.411	0.667
Default -- Drop S (separability)	0.595	0.463	1.000
Default -- Drop C (complexity)	0.795	0.674	0.333
Default -- Drop R (redundancy)	0.656	0.516	0.667
Default -- Drop N (noise)	0.553	0.453	0.667
Full ADUS (NNLS-learned)	0.558	0.432	0.333
Learned -- Drop I	0.483	0.400	0.667
Learned -- Drop S	0.332	0.232	0.333
Learned -- Drop C	0.653	0.484	0.333
Learned -- Drop R	0.558	0.432	0.333
Learned -- Drop N	0.558	0.432	0.333

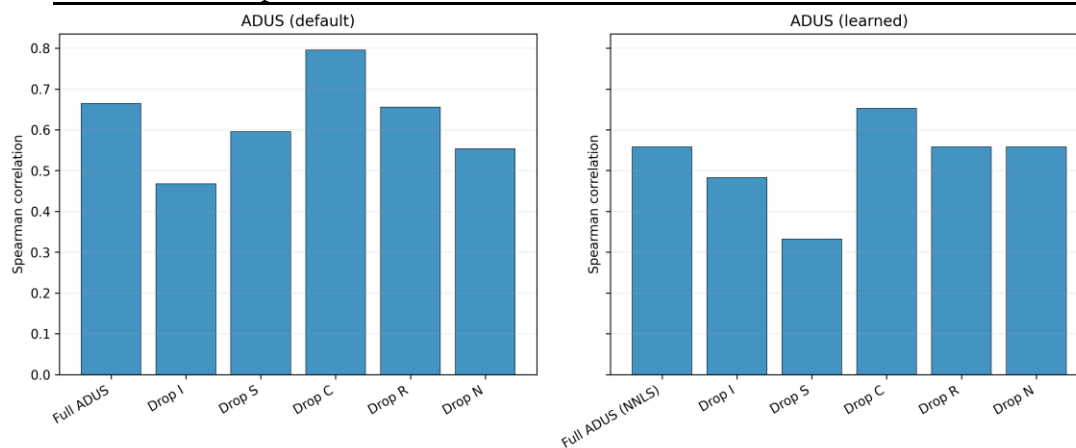


Figure 8. ADUS component ablation, default and NNLS-learned coefficients.

6.3 Meta-Learner Hyperparameters

Figure 9 reports the Spearman correlation, Kendall tau, and mean absolute error of the random forest meta-learner as the number of trees increases from 50 to 500, all computed under nested leave-one-dataset-out at each setting so no value in the sweep reuses the same held-out labels used to pick it. Spearman rises from 0.537 at 50 trees to a peak of 0.614 at 300 trees, and MAE falls in step, reaching its minimum of 0.052 at the same setting; both metrics degrade mildly beyond 300 trees (Spearman 0.579 at 500). We adopt 300 trees as the operational default used throughout Sections 5 and 6.1.

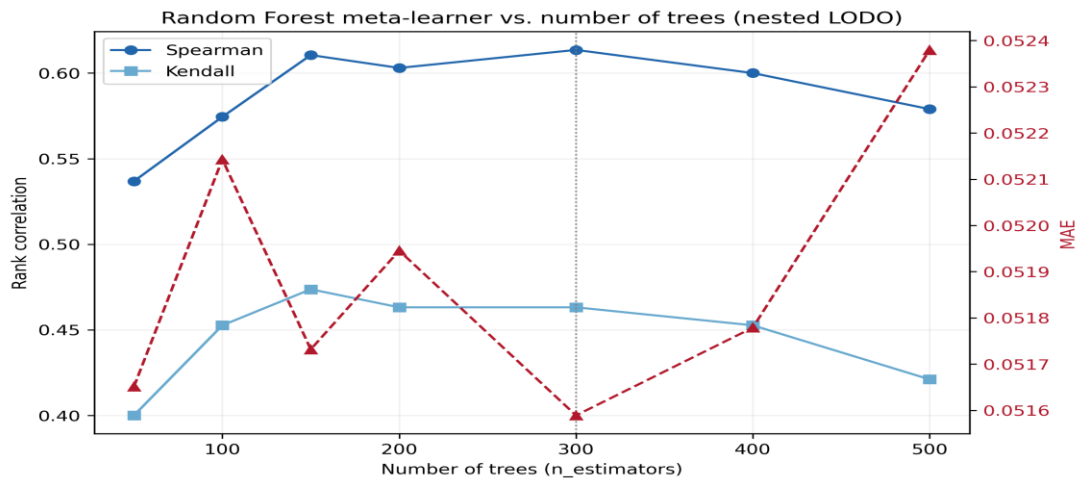


Figure 9. Spearman correlation, Kendall tau, and MAE of the random forest meta-learner as a function of the number of trees, evaluated under nested leave-one-dataset-out. The curve plateaus near 300 estimators.

7 Discussion

7.1 Why a Compact Scoring Function Is Competitive with a Meta-Learner

The empirical result that ADUS-default surpasses every one of the four learned meta-learners in Spearman correlation (0.665 against a best of 0.614 for Random Forest) on this specific 20-dataset collection deserves scrutiny. With only 19 training examples per fold and 31 candidate features, the regression problem is severely underdetermined. A random forest with 300 trees is capable of memorising the training set perfectly, but its leave-one-out generalisation depends on the existence of nearby points in feature space, which is rare for a held-out dataset that differs from the others on multiple dimensions. The analytical formulation of ADUS, by contrast, encodes a prior over which descriptors should matter and exploits no information beyond that prior. When the prior is approximately correct, it performs better than data-driven coefficients in low-sample regimes, and our results suggest that the prior is in fact reasonably well aligned with this collection, though the group ablation of Section 6.1 also suggests the prior currently under-weights complexity relative to information. Importantly, the default ADUS variant does not require any historical performance records to operate, which is a substantial practical advantage in cold-start situations.

7.2 The Zero Coefficients of NNLS

The NNLS procedure consistently assigns zero weight to the redundancy and noise components across all 20 leave-one-out folds, while retaining positive weight on information, separability, and complexity -- with separability carrying the largest and most stable share (Section 5.3). We interpret the zeroing of redundancy and noise as a finding rather than a failure: the mean pairwise feature-feature mutual information and the nearest-neighbour disagreement rate both have small variance across the 20 problems in this collection, so a squared-error fit gains little from assigning them non-zero weight. That separability, rather than information, ends up with the largest learned weight is a different picture from the equal 1.0/1.0 default weighting on these two terms, and it is consistent with the linear-separability landmark (a cheap classifier's own held-out accuracy) being a more direct proxy for downstream classifier performance than any purely information-theoretic quantity. Future work should test ADUS on a meta-dataset specifically engineered to vary along the redundancy and noise axes, where these terms are expected to recover positive weights.

7.3 The Bell-Shaped Complexity Term

The complexity component as currently implemented peaks at intermediate intrinsic dimensionality. The peak reflects the heuristic that very low-dimensional problems are either too easy to be informative or, in the non-linear case, surprisingly hard to characterise with linear meta-features. The `synth_circles` and `synth_moons` datasets both have two features and intrinsic dimensionality close to one, but their realised mean balanced accuracy is high (0.869 and 0.942 respectively) because non-linear classifiers solve them easily. Our complexity proxy fails to capture this, which is exactly what produces the trade-off documented in the component ablation of Section 6.2: removing the term raises overall Spearman correlation from 0.665 to 0.795 while lowering top-3 set overlap from 0.667 to 0.333. A more faithful proxy would combine the Levina-Bickel estimator with a non-linear separability surrogate such as a one-nearest-neighbour landmarker. We leave this extension, and a re-run of the nested ablation to confirm whether the trade-off persists under a revised term, to future work.

7.4 Threats to Validity

Four threats to the validity of our conclusions deserve mention. First, the meta-dataset is small. Although 20 datasets is sufficient to demonstrate the qualitative behaviour of ADUS, the absolute magnitudes of correlations are subject to wide confidence intervals. A paired bootstrap over the 20 datasets (5,000 resamples) gives a standard error of 0.184 for the ADUS-default Spearman correlation and 0.198 for the ADUS-learned correlation; the 95% confidence interval for their difference is [-0.065, 0.317], which includes zero. The apparent advantage of the default coefficients over the learned coefficients on this collection should therefore be read as suggestive rather than statistically established, and the same caveat applies to the 0.051-point gap between ADUS-default and the Random Forest meta-learner (95% CI [-0.253, 0.390], also spanning zero). Second, the synthetic datasets are generated with a single random seed (42, documented per-dataset in Appendix A). A more comprehensive study should vary the seed and report distributions rather than point estimates for the synthetic portion of the collection specifically. Third, the base learner portfolio is fixed and small. A larger portfolio that includes ensembles, kernel methods with non-RBF kernels, and modern boosting variants would produce different utility labels, and the ADUS coefficients would need to be retrained. Fourth, several per-dataset noise-robustness scores are negative and the overall robustness standard deviation (0.700) is larger than the mean (0.396, Section 5.5); this is reported as a substantive limitation of the current normalisation in low-sample or strongly nonlinear regimes, not smoothed into a single reassuring average.

7.5 Practical Implications

Three implications of our results are directly useful for practitioners. First, when only a handful of historical performance records are available, the default-coefficient version of ADUS is a reliable cold-start scoring function that requires no training and, on this collection, outperformed every learned alternative on rank correlation even though the difference from the strongest meta-learner is not statistically conclusive at this sample size. Second, the complexity and information-theoretic meta-feature groups captured most of the ranking signal in isolation (Section 6.1); practitioners facing a computational budget could restrict extraction to these two groups (14 of the 31 descriptors) with limited expected loss in ranking quality. Third, the choice of meta-feature groups matters more than the choice of meta-learning model: removing the complexity group is far more damaging than switching from Random Forest to LightGBM. We recommend that AutoML systems prioritise the complexity and information descriptors when constructing meta-feature representations, while treating class-imbalance-sensitive utility -- a

case where the current ADUS formulation under-performs, as shown by the synth_imbalanced anomaly in Section 5.3 -- as an open problem.

8 Conclusion and Future Work

This paper introduced ADUS, an Adaptive Dataset Utility Score that ranks candidate datasets for a downstream classifier portfolio before any expensive model training is performed. The score combines five interpretable terms derived from a 31-feature taxonomy organised into statistical, structural, information-theoretic, and complexity groups, and admits both an analytical default and an NNLS-learned variant fit under a nested leave-one-dataset-out protocol. Empirical evaluation on 20 heterogeneous datasets shows that ADUS-default attains Spearman correlation 0.665 with realised utility, ahead of the strongest data-driven meta-learner (Random Forest, 0.614) and a sample-count proxy (0.155), though this margin is not statistically conclusive at $n=20$ (Section 7.4). Both ADUS variants correctly place the single best-performing dataset first in the aggregated ranking, a result matched by no baseline or meta-learner; on top-3 set overlap, ADUS-default ties with the Random Forest and LightGBM meta-learners rather than surpassing them outright, and the information-density baseline, while weaker on rank correlation than ADUS, is not distinguishable from it on that single top-1 criterion alone. An ablation study isolates the complexity feature group as the single strongest contributor to ranking quality, ahead of the information-theoretic group, and surfaces a documented trade-off: the bell-shaped complexity term in ADUS raises rank correlation but lowers top-3 retrieval when removed from the default formula, on low-dimensional non-linear problems. The complete implementation, including a CustomTkinter desktop application and the 20 reproducible datasets (documented in full in Appendix A) is released alongside this paper.

Several directions are open for future work. The most immediate is to replace the bell-shaped complexity term with a monotone landmarker-based estimator, which is expected to recover positive ranking gains on the low-dimensional problems where the current proxy fails, and to re-run the nested component ablation to confirm the effect. A second direction is to scale the meta-dataset by integrating with OpenML, which would enable a fairer evaluation of high-capacity meta-learners, a more reliable NNLS solution, and multi-seed synthetic generation (Section 7.4) in place of the single-seed collection used here. A third direction is to extend the utility target beyond balanced accuracy -- for instance to macro-F1 or cost-sensitive metrics -- and to extend the descriptor set with a component that directly captures class-imbalance severity, which the synth_imbalanced anomaly of Section 5.3 suggests is currently under-represented. A fourth direction is to extend ADUS to regression and time-series tasks, where meta-feature taxonomies have been studied less systematically and where the notion of dataset utility differs qualitatively. Finally, integrating ADUS with active dataset acquisition, where a budget of additional labels is allocated to the candidate dataset predicted to gain the most utility, is a promising avenue for combining meta-learning with data-centric machine learning.

DECLARATIONS

Funding. The authors received no external funding for this research.

Conflicts of interest. The authors declare that they have no competing interests.

Ethics approval. Not applicable. This study did not involve human participants or animals.

Informed consent. Not applicable.

Data and code availability. The datasets and source code used in this study are publicly available at <https://github.com/nuseir/Meta-Learning-for-Dataset-Selection>

Author contributions. Abstract: Authors 1, 2 and 3, Introduction: 1, 2, 3, Related Work: 4 and 5, Methodology: 1, 2, 3, Experimental Setup 1, 2, 3, 4, 5, Results: 1, 2, 3, 4, 5, Ablation Study: 1, 2, 3, Discussion: 1, 2, 3, 4, 5, Conclusion and Future Work: 1, 2, 3.

Use of artificial intelligence. Authors declare that no AI tools were used beyond language editing.

Acknowledgements. None.

REFERENCES

- [1] Bensusan, H., Giraud-Carrier, C., & Kennedy, C. (2000). A higher-order approach to meta-learning. In *Proceedings of the ECML Workshop on Meta-Learning*, 109-117.
- [2] Drori, I., Krishnamurthy, Y., Rampin, R., Lourenco, R., One, J., Cho, K., Silva, C., & Freire, J. (2018). AlphaD3M: Machine learning pipeline synthesis. In *ICML AutoML Workshop*.
- [3] Feurer, M., Klein, A., Eggenberger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, 28, 2962-2970.
- [4] Ho, T. K., & Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3), 289-300.
- [5] Jomaa, H. S., Schmidt-Thieme, L., & Grabocka, J. (2021). Dataset2Vec: Learning dataset meta-features. *Data Mining and Knowledge Discovery*, 35(3), 964-985.
- [6] Lawson, C. L., & Hanson, R. J. (1974). *Solving Least Squares Problems*. Prentice-Hall, Englewood Cliffs, NJ.
- [7] Levina, E., & Bickel, P. J. (2005). Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems*, 17, 777-784.
- [8] Lorena, A. C., Garcia, L. P. F., Lehmann, J., Souto, M. C. P., & Ho, T. K. (2019). How complex is your classification problem? A survey on measuring classification complexity. *ACM Computing Surveys*, 52(5), 1-34.
- [9] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 30, 4765-4774.
- [10] Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.
- [11] Northcutt, C. G., Jiang, L., & Chuang, I. L. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373-1411.
- [12] Pleiss, G., Zhang, T., Elenberg, E., & Weinberger, K. Q. (2020). Identifying mislabeled data using the area under the margin ranking. In *Advances in Neural Information Processing Systems*, 33, 17044-17056.
- [13] Rice, J. R. (1976). The algorithm selection problem. *Advances in Computers*, 15, 65-118.
- [14] Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021). Everyone wants to do the model work, not the data work: Data cascades in high-stakes AI. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1-15.
- [15] Smith-Miles, K. A. (2008). Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys*, 41(1), 1-25.
- [16] Vanschoren, J., van Rijn, J. N., Bischl, B., & Torgo, L. (2014). OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2), 49-60.
- [17] Wistuba, M., Schilling, N., & Schmidt-Thieme, L. (2018). Scalable Gaussian process-based transfer surrogates for hyperparameter optimization. *Machine Learning*, 107(1), 43-78.

- [18] Yao, Q., Wang, M., Chen, Y., Dai, W., Li, Y. F., Tu, W. W., Yang, Q., & Yu, Y. (2018). Taking human out of learning applications: A survey on automated machine learning. arXiv preprint arXiv:1810.13306.
- [19] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., & Smola, A. J. (2017). Deep sets. In *Advances in Neural Information Processing Systems*, 30, 3391-3401.

APPENDIX A. COMPLETE DATASET INVENTORY

All 16 synthetic datasets are generated with scikit-learn under generation seed 42. n, d, K are sample count, feature count, and class count after generation (before the 75/25 train-test split used for utility estimation).

Table A1. Complete inventory of the 20-dataset collection.

Dataset	Source	N	D	K	Generator / key parameters
Iris	sklearn (real)	150	4	3	load_iris()
Wine	sklearn (real)	178	13	3	load_wine()
breast_cancer	sklearn (real)	569	30	2	load_breast_cancer()
Digits	sklearn (real)	1797	64	10	load_digits()
synth_easy_2classes	synthetic	400	10	2	make_classification, informative=6, class_sep=2.5
synth_moons	synthetic	400	2	2	make_moons, noise=0.15
synth_circles	synthetic	400	2	2	make_circles, noise=0.10, factor=0.5
synth_gaussian_quantiles	synthetic	400	8	2	make_gaussian_quantiles
synth_blobs_3class	synthetic	400	8	3	make_blobs, centers=3, cluster_std=2.2
synth_many_classes	synthetic	600	15	6	make_classification, informative=10, class_sep=1.2
synth_imbalanced	synthetic	500	10	2	make_classification, weights=[0.92,0.08], flip_y=0.01
synth_mild_imbalance	synthetic	500	10	2	make_classification, weights=[0.65,0.35]
synth_highdim_smalln	synthetic	200	80	2	make_classification, informative=15, redundant=10
synth_redundant	synthetic	400	20	2	make_classification, informative=5, redundant=13
synth_noisy_labels	synthetic	400	10	2	make_classification, flip_y=0.20
synth_noisy_features	synthetic	400	40	2	make_classification, informative=6, redundant=0
synth_large_clean	synthetic	3000	12	2	make_classification, informative=8, flip_y=0.0
synth_tiny	synthetic	120	6	2	make_classification, informative=4, class_sep=1.7

synth_moderate_5class	synthetic	500	12	5	make_classification, informative=8, redundant=2
synth_high_signal_baseline	synthetic	400	10	2	make_classification, informative=9, flip_y=0.02

APPENDIX B. COMPLETE META-FEATURE DEFINITIONS

All 31 meta-features are computed from the training partition of the first repeated split (Section 4.3). Variable names match the released implementation.

Table B1. Complete 31-descriptor meta-feature taxonomy (7 statistical, 10 structural, 6 information-theoretic, 8 complexity).

Feature	Group	Definition
mean_feature_std	Statistical	Mean, across features, of each feature's standard deviation
std_feature_mean	Statistical	Standard deviation, across features, of each feature's mean
mean_feature_skewness	Statistical	Mean absolute skewness across features (0 for zero-variance features)
mean_feature_kurtosis	Statistical	Mean excess kurtosis across features (0 for zero-variance features)
coefficient_of_variation	Statistical	Mean, across features, of $\text{std}/ \text{mean} $
class_balance_ratio	Statistical	min class frequency / max class frequency
mean_feature_correlation	Statistical	Mean absolute pairwise Pearson correlation between feature columns
n_samples	Structural	Number of instances n in the training partition
n_features	Structural	Number of features d
n_classes	Structural	Number of classes K
sample_feature_ratio	Structural	n / d
n_binary_features	Structural	Count of features with exactly 2 distinct values
n_high_cardinality_features	Structural	Count of features with $> 10\%$ of n distinct values
proportion_missing	Structural	Fraction of missing cells (0 unless missingness is injected)
pca_var_ratio_pc1	Structural	Variance ratio explained by the first principal component
k90_components	Structural	Smallest k such that k principal components explain $\geq 90\%$ variance
spectral_effective_rank	Structural	$\exp(\text{Shannon entropy of the normalised eigenvalue spectrum})$
mean_mutual_info	Information	Mean feature-label mutual information (k-NN estimator)
std_mutual_info	Information	Std. dev. of feature-label mutual information across features
label_entropy	Information	Shannon entropy (base 2) of the label distribution

mean_conditional_entropy	Information	Mean, across features, of $H(\text{feature} \text{label})$ via Freedman-Diaconis binning
mean_pairwise_feature_mi	Information	Mean pairwise mutual information between feature columns (≤ 60 sampled pairs)
max_pairwise_feature_mi	Information	Max pairwise mutual information between feature columns
fisher_ratio_mean	Complexity	Mean, across features, of between-class / within-class variance
fisher_ratio_max	Complexity	Max per-feature Fisher discriminant ratio
overlap_region_mean	Complexity	Mean class-conditional feature-range overlap across features and class pairs
nn_disagreement_rate	Complexity	Fraction of points whose nearest neighbour has a different label
intrinsic_dim_mle	Complexity	Levina-Bickel MLE intrinsic dimensionality estimate ($k=1..10$, clipped to $[0.5, d]$)
linear_separability_lda	Complexity	Held-out accuracy of an LDA classifier on a 70/30 subsample split
boundary_point_ratio	Complexity	Fraction of points with ≥ 1 differently-labelled point among 5 nearest neighbours
centroid_distance_ratio	Complexity	Mean inter-class centroid distance / mean within-class spread