2025

Benchmarking 15 Machine Learning Models for Binary Classification: Accuracy, Complexity, and Speed

A Comprehensive Evaluation Across 159 Tabular Datasets

Ed Kaempf
December 2025

Benchmarking 15 Machine Learning Models for Binary Classification: Accuracy, Complexity, and Speed

Executive Summary

This study benchmarks 15 machine learning models on 159 tabular datasets for binary classification to answer four questions: (1) Which models perform best overall? (2) What makes datasets difficult to classify? (3) Which models handle specific complexity types most effectively? (4) How do accuracy and speed trade off across models?

The models represent eight algorithm families: (1) tree-based (Decision Tree), (2) tree ensembles (Random Forest, Extra Trees, Gradient Boosting, LightGBM, XGBoost), (3) linear models (Logistic Regression, SGD Classifier, Linear Support Vector Classifier), (4) kernel (Support Vector Classifier), (5) instance-based (K-Nearest Neighbors), (6) probabilistic (Gaussian Naïve Bayes), (7) discriminant (Linear Discriminant Analysis, Quadratic Discriminant Analysis), and (8) neural network (Multi-Layer Perceptron).

Tabular datasets underpin the majority of enterprise machine learning applications, driving critical decisions in nearly every domain, such as healthcare diagnostics, financial risk modeling, fraud prevention, and customer analytics. Yet practitioners consistently face a fundamental challenge: determining which algorithm will perform best on their specific dataset. Traditional approaches rely on iterative testing of multiple models, a resource-intensive process that delays deployment and offers limited insight into why certain algorithms succeed or fail.

This study takes a systematic approach by linking model performance to measurable dataset complexity. Using the Lorena et al. framework, [1] we characterized each of 159 datasets across 22 complexity dimensions spanning feature discriminability, class separability, geometric structure, and neighborhood cohesion. By correlating the complexity measures with the performance of 15 diverse models, we identify not only which models deliver better results overall, but precisely which algorithms are best suited to handle specific types of dataset difficulty.

Key Findings

Model Performances

Statistical testing using the Friedman and Nemenyi procedures identified a clear performance hierarchy across the 15 evaluated models. Seven models (XGBoost, Random Forest, LightGBM, MLP, Extra Trees, SVC, and Gradient Boosting) form a statistically equivalent top tier, showing no statistically significant differences among themselves, while clearly outperforming the remaining models. This top tier achieved mean accuracy scores ranging from 0.850 to 0.861 across 159 datasets, compared to 0.811 to 0.831 for mid-tier models and 0.727 for the lowest-performing model (Naive Bayes). On small datasets (≤500 records), MLP (0.828) and SVC

(0.822) emerged as leaders, the only instance where non-ensemble models topped the hierarchy.

Notably, 68.6% of all pairwise model comparisons (72 of 105 pairs) showed statistically significant performance differences in model capabilities, rather than marginal distinctions. This finding indicates that any of the seven top-tier models can be selected with confidence, with choice determined by secondary considerations such as training time, interpretability, or available resources, rather than concerns about performance loss.

To assess overall performance, an integrated score balanced predictive accuracy (60%) with computational throughput (40%). Under this deployment metric, Linear models (Linear SVC, Logistic Regression, and SGD) and Discriminant analysis (LDA) emerge as the leaders, as their exceptional speed outweighs the marginal accuracy gains of more complex algorithms. This reveals a critical deployment pattern: while sophisticated tree-based ensembles often achieve higher raw accuracy, their substantial processing overhead makes them less optimal for high-volume applications requiring a balance of speed and precision.

Dataset Complexity and Performance Drivers

Analysis of 22 dataset complexity measures revealed that neighborhood-based complexity dominates as the primary predictor of classification difficulty. The three neighborhood measures (N1, N3, N4) exhibited the strongest negative correlations with accuracy (r = -0.78, -0.77, -0.73 respectively, all p < 0.001), indicating that datasets with poorly defined class boundaries and overlapping decision regions consistently challenge model performance. Linearity measures (L2, L3) also demonstrated strong predictive power (r = -0.71, -0.69), confirming that non-linear decision boundaries substantially increase classification difficulty.

Feature importance analysis using Random Forest and Linear Regression methods validated these results. Each method independently ranked N1 as the single most important predictor and consistently placed neighborhood (N1, N3, N4) and linearity (L2, L3) among the top seven drivers of difficulty.

Class imbalance measures (C1, C2) showed weak, counter-intuitive positive correlations with accuracy (r = +0.36, +0.35). However, this relationship proved spurious: C1 exhibited systematic negative correlations with 17 of the 22 complexity measures[1] *(p < 0.05)*, including strong negative relationships with N1 (r = -0.47), L2 (r = -0.47), and N3 (r = -0.43). High imbalance datasets in this project's collection coincidentally exhibited lower geometric complexity, creating a misleading positive link with accuracy. Feature importance analysis confirmed C1's unreliability, ranking it 19th of 22 measures with less than 1% importance. The results established geometric and topological complexity, rather than class distribution, as drivers of classification difficulty for the algorithms and datasets examined here.

[1] C1 also had a strong negative correlation (p < 0.05*)* with the mean of all 22 complexity measures ("score" field from the Python problexity library that calculates the 22 Lorena et al. [1] measures used in this project).

Model-Complexity Matching

Model performance was evaluated across the six complexity categories: feature-based, linearity, neighborhood, network, dimensionality, class imbalance. The review found some distinct performance tiers. Models were rated *Strong*, *Moderate*, or *Weak* based on performance within the top 25% most complex datasets in each category. Random Forest and XGBoost achieved *Strong* ratings in all six complexity categories, while Extra Trees, LightGBM, and MLP earned *Strong* ratings in five of six, forming a top tier of five consistently high-performing models. In contrast, five models (SGD, Naive Bayes, LDA, QDA, and Linear SVC) received predominantly *Weak* ratings across categories. This confirmed substantial variation in performance, beyond marginal differences across models.

Neighborhood complexity emerged as the primary discriminator of model capability, with only five models achieving *Strong* ratings on high-neighborhood-complexity datasets. Models weak on neighborhood complexity generally struggled elsewhere as well, indicating that handling complex decision boundaries is a key indicator of overall model quality. The five top-tier models demonstrated strong performance across all complexity types, indicating a group of reliable algorithm choices regardless of underlying dataset characteristics.

Computational Trade-offs

Model throughput varied widely, from 40,000 predictions per second with KNN to 7 million with LDA, a 176-fold difference. Linear models (Logistic Regression, SGD, Linear SVC) and LDA were fastest (6-7 million predictions/second) but delivered lower accuracy in the 0.814-0.830 range, placing them in the bottom half of performance rankings. In contrast, tree ensemble methods (Random Forest, Extra Trees) achieved top-tier accuracy (0.855-0.859) but ran nearly 100 times slower than linear models at approximately 70,000 predictions per second.

Gradient boosting models (XGBoost, LightGBM) provided the best balance of speed and accuracy. These two had higher accuracy (0.858-0.861) while maintaining throughput 14-17 times faster than Random Forest and Extra Trees and only 5-7 times slower than linear models. XGBoost was most accurate (0.861) at 1.3 million predictions per second, offering moderate speed compared to linear models, with a 4.8-point accuracy gain.

For most applications, the slower speed of top-tier models is justified by their substantially higher accuracy. Linear models remain a practical choice when sub-second response times are critical and modest accuracy reductions (3–5 percentage points) are acceptable.

Practical Implications

Seven models (XGBoost, Random Forest, LightGBM, MLP, Extra Trees, GBM, SVC) formed a statistically equivalent top-tier performance across all six complexity categories. Selection among these models can be guided by secondary criteria such as training time, interpretability, or infrastructure, since their performance differences were statistically insignificant. Models ranking outside this top tier exhibited substantially weaker performance across most complexity types.

Geometric and topological complexity measures proved to be the primary drivers of classification difficulty. Neighborhood complexity (overlapping decision regions, poorly defined boundaries) and non-linear decision boundaries showed strong negative correlations with accuracy, while class imbalance showed minimal impact on model performance. Datasets with high geometric complexity presented significant challenges for lower-tier algorithms.

XGBoost, LightGBM, and MLP achieved the best balance of predictive performance and speed, delivering highest-tier accuracy while maintaining throughput 14-17 times faster than Random Forest and Extra Trees. These three models processed predictions at approximately 1-1.3 million per second while achieving equivalent accuracy, making them well-suited for applications where both performance and speed matter. Linear models achieved throughput (6-7 million predictions/second) that was 2-100 times faster than all other models, but with 3-5 percentage points lower accuracy, making them appropriate when response time requirements outweigh performance considerations.

1. Introduction

This section is organized as follows:

1.1 Overview and Motivation

1.2 Dataset Complexity and Model Performance

1.3 Tabular Datasets

1.4 Report Organization.

1.1. Overview and Motivation

The primary goal of a machine learning model is to make accurate predictions on new, unseen data. Machine learning practitioners consistently face a fundamental challenge: selecting which algorithm will perform best on a specific dataset. Traditional benchmarking studies evaluate models across multiple datasets but rarely examine how dataset characteristics affect algorithm effectiveness. As a result, practitioners often rely on trial-and-error experimentation with limited insight into why certain algorithms succeed or fail.

This study addresses this gap by systematically linking algorithm performance to measurable dataset complexity. By characterizing 159 binary classification datasets across 22 complexity dimensions and evaluating 15 diverse machine learning models, this work establishes which complexity factors drive classification difficulty, and which models handle specific complexity types most effectively. These findings enable evidence-based algorithm selection tailored to dataset characteristics rather than generic "best practices" that may not apply to specific problem contexts.

1.2. Dataset Complexity and Model Performance

Dataset complexity measures quantify characteristics that influence classification difficulty. The measures span geometric structure, feature relationships, class separability, and data topology.

Ho and Basu in 2002 [2], introduced 12 complexity measures to characterize the difficulty. Ho and Basu defined the 12 measures, grouping them into overlap measures (F1–F3), separability measures (L1–L2, N1–N3), and geometry/topology measures (L3, N4, T1, T2). Together, these measures capture how features, class boundaries, and structural properties affect classification difficulty.

In their 2018 study [1], Lorena et al. added measures for feature correlation, linearity, neighborhood, network, dimensionality, and class balance. Lorena et al. developed 22 measures of dataset complexity, organized in six categories.

This project applies all 22 measures to characterize dataset complexity. These state-of-the-art measures detect class overlap and relate directly to classification outcomes. [3] The 22 measures are organized into six categories:

- **Feature-based**: discriminative power of individual features
- **Linearity**: decision boundary linearity
- **Neighborhood**: local class overlap
- **Network**: data as a graph theoretic connectivity
- **Dimensionality**: feature-to-sample ratio and sparsity effects
- **Class imbalance**: minority class representation.

By correlating these complexity measures with model performance across 159 datasets, this work identifies which complexity dimensions most strongly predict classification difficulty, and which models demonstrate robustness across different complexity profiles. This complexity-aware approach enables practitioners to assess dataset characteristics before model selection, improving algorithm choice beyond generic accuracy rankings.

1.3. Tabular Datasets

Tabular datasets, organized as rows of samples and columns of features, appear in spreadsheets, databases, and data warehouses, and may be structured or semi-structured. They underpin mission-critical applications across agriculture, biology, education, engineering, finance (Nureni & Adekola, 2022), healthcare (Johnson et al., 2016; Ulmer et al., 2020), human resources, manufacturing (Chen et al., 2023), and retail. Use cases span click through rate prediction, credit risk analysis, customer churn prediction, drug efficacy, fraud detection, identity protection, investment analysis, medical diagnosis, and quality control, among many others. [4] [5]

Despite their ubiquity, tabular datasets pose distinct challenges that can significantly affect model performance. Machine learning models are sensitive to data quality issues that can undermine performance. Common issues include missing values, mislabeled records, non-numeric (i.e., categorical) features, value ranges, class imbalances, sparsity, outliers, redundancy among features, high dimensionality (many features relative to number of records), and noise.

Missing values bias training and reduce reliability, while mislabeled records mislead algorithms and degrade accuracy. Categorical features distort relationships if not properly encoded, and inconsistent value ranges skew optimization when features differ in scale. Class imbalances can bias models toward majority outcomes, and sparsity limits signal strength and hinders generalization.

Outliers disproportionately influence models, while redundant features add noise without improving accuracy. High dimensionality, with many features relative to records, overwhelms algorithms and hinders performance, while noise obscures true patterns and reduces predictive power.

Because of these unique challenges, tabular datasets require dedicated benchmarking to evaluate model performance under realistic conditions. Unlike image or text domains, where standardized benchmarks are well established, tabular data vary widely in scale, feature types, and class distributions. Without consistent testing like this, it is hard to fairly evaluate models or understand which ones are right for different business needs.

1.4. Report Organization

The remainder of this report is organized as follows:

- **Section 2, Methodology** – machine learning models evaluated, datasets used for benchmarking, dataset complexity measures, evaluation procedures, and benchmarking workflow
- **Section 3, Performance Metric Validation** – correlations between accuracy, F1, and AUC, with justification for using accuracy as the primary metric
- **Section 4, Model Performance** – model performance results across all datasets
- **Section 5, Complexity Analysis** – dataset complexity characteristics and their impact on performance
- **Section 6, Model–Complexity Analysis** – model performance examined by complexity measure.
- **Section 7, Computational Efficiency** – throughput results and accuracy–throughput tradeoffs
- **Section 8, Conclusion** – summary of results and key takeaways
- **Section 9, Limitations and Future Work** – constraints of this study and directions for extension.

Appendices provide supporting details:

- **Appendix A, Dataset Complexity Measures** – descriptions of each measure, organized by the six categories of Lorena et al. measures [1]
- **Appendix B, Dataset Characteristics and Sources** – full list of datasets with name, records, features, source, and URL

- **Appendix C, Computational Infrastructure** – Python environment, hardware specifications, and code availability statement
- **Appendix D, Pearson Correlations Between Complexity Measures and Accuracy by Model Family** – Pearson correlations between all 22 complexity measures and accuracy computed separately for each of the eight model families.

Finally, the **References** section lists all cited studies.

2. Methodology

This section is organized as follows:

2.1 Machine Learning Models Evaluated

2.2 Datasets Used for Benchmarking

2.3 Dataset Complexity Measures

2.4 Evaluation Procedures

2.5 Benchmarking Workflow.

2.1. Machine Learning Models Evaluated

This study evaluated 15 machine learning models selected to represent a broad cross-section of classification algorithms. These models span diverse learning strategies, from simple linear classifiers to complex ensemble methods, demonstrating the range of techniques available for structured tabular data.

All 15 models are widely adopted in practice and well-supported by mainstream machine learning libraries such as scikit-learn, XGBoost, and LightGBM. The selection emphasizes established, production-ready algorithms rather than experimental or domain-specific approaches, ensuring findings are directly applicable to real-world classification tasks.

The models are organized into eight algorithm families based on their core learning mechanisms: tree-based (single decision tree), tree ensemble (bagging and boosting methods), linear (logistic regression and variants), kernel (support vector machines), instance-based (nearest neighbors), probabilistic (Naive Bayes), discriminant analysis, and neural network. This taxonomy supports systematic comparison of how different algorithms respond to varying dataset characteristics and complexity profiles. The models are listed on the next page by family.

Model names are shortened for readability. For example, “Classifier” is dropped once the context is clear, and widely recognized abbreviations (e.g., “XGBoost,” “SVC”) are used.

A. Tree-based

1. Decision Tree Classifier

B. Tree Ensemble

2. Random Forest Classifier
3. Extra Trees Classifier
4. Gradient Boosting Classifier
5. LGBM Classifier
6. XGBoost Classifier

C. Linear

7. Logistic Regression
8. Stochastic Gradient Descent (SGD) Classifier
9. Linear SVC

D. Kernel-based

10. Support Vector Classifier

E. Instance-based

11. KNeighbors Classifier

F. Probabilistic

12. Gaussian Naive Bayes (GaussianNB)

G. Discriminant

13. Linear Discriminant Analysis
14. Quadratic Discriminant Analysis

H. Neural Network

15. Multi-layer Perceptron (MLP) Classifier

2.2. Datasets Used for Benchmarking

This project used 159 publicly available datasets selected based on three criteria:

- A target field existed and contained no more than three values
- No more than 15% of the records had missing values
- Did not involve games like poker or chess (where the target is a deterministic function of the data).

Of the 159 datasets, 143 had no missing values, 6 had less than 3% missing values, 8 more had less than 10% missing values, and 2 had less than 15% missing values. All are available from repositories of datasets not intended to represent the diverse, unclean, and complex data found in real-world datasets. Many are preprocessed before release (e.g., missing values removed, outliers filtered). As noted by [6], datasets may also have well-engineered features that simplify the modelling process and enhance prediction performance.

A simple measure of class imbalance is the ratio of records in the majority class to those in the minority class. A perfectly balanced dataset has a ratio of 1.0, while larger values indicate increasing imbalance. Across the 159 datasets, the median imbalance was 1.68, with values ranging from 1.0 to 577.9 (creditcard dataset).

Thirty-eight datasets included categorical features; the remaining 121 did not. All categorical features were one-hot encoded. Encoding produced between 2 and 6,572 features per dataset, with an average of 256 encoded features.

Due to time limits, the 159 datasets represent a small set of all datasets meeting the selection criteria. The full list of datasets is provided in **Appendix B**. **Figure 2.1** summarizes the sources from where the datasets were obtained. [2]

Figure 2.1 Dataset Sources

Source	Count
OpenML	111
UCI	27
Kaggle	15
Other	6
Total	**159**

Figure 2.2 profiles of the 159 datasets. The model benchmarks capture performance across both small and large, simple and complex datasets.

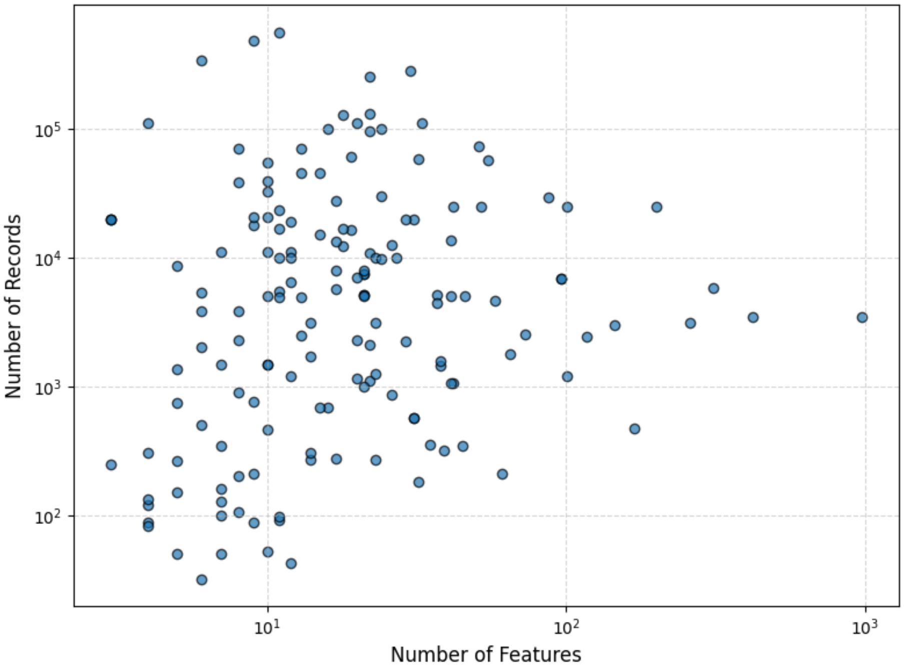
Figure 2.2 Dataset Characteristics: Size and Feature Counts

Attribute	Count
Record Count Median	5,000
Record Count Range	32 – 566,602
Feature Count Median	17
Feature Count Range	2 – 2,569
Numeric Features:	
Count Mean	50
Count Median	13
Count Range	1 – 2,568
Categorical Features:	
Datasets with at Least One Categorical	38
Count Mean: All 159 Datasets The 38 with At Least One Categorical	 1 5
Count Median: All 159 Datasets The 38 with At Least One Categorical	 0 3
Encoded Count Median	11
Encoded Count Range	2 – 6,572

[2] The four sources of the datasets are: OpenML, UCI Machine Learning Repository, Kaggle, and Scikit_Learn .

Three of the four smallest datasets, with two features each, were the three datasets from scikit-learn's suite of toy datasets[3]. The malware classification EMBER2024 dataset available on GitHub has 2,568 features and 3.2 million labeled samples. Because of a compute limit (32GB RAM), 50,000 samples were randomly selected for benchmarking. **Figure 2.3** illustrates dataset size and dimensionality, shown on a logarithmic scale for both record counts and feature counts.

Figure 2.3 Dataset Characteristics: Number of Records and Dimensionality (log$_{10}$)



2.3. Dataset Complexity Measures

Complexity measures quantify dataset traits that affect classification difficulty. They capture aspects of geometric structure, feature relationships, class separability, and data topology. This project employs the Lorena et al. (2019) framework, [1] which defines 22 measures organized into six categories: feature-based, linearity, neighborhood, network, dimensionality, and class imbalance. By correlating these measures with model performance, we identify which dimensions most strongly predict classification difficulty and which models demonstrate robustness.

Ho and Basu (2002), [2] introduced 12 measures focused on the geometric characteristics of class distributions. They emphasized that the way classes are separated or interleaved is critical for classification accuracy. Their measures fell into three groups:

- Overlap of feature values: Fisher's discriminant ratio (F1), Volume of overlap region (F2), Feature efficiency (F3).

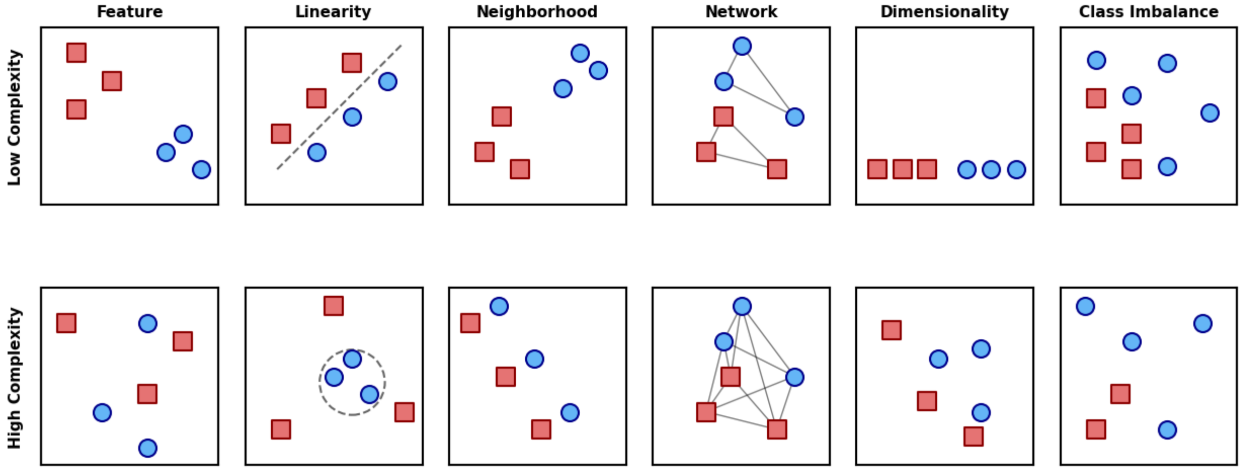
[3] The three scikit-learn toy datasets are: make_blobs, make_circles, and make_moons.

- Separability: Linear separability (L1, L2), Mixture identifiability (N1, N2, N3).
- Geometry, topology, and density: Nonlinearity (L3, N4), Proportion of hyperspheres covering data (T1), Average number of samples per feature (T2).

Lorena et al. [1] expanded the framework to include feature correlation, linearity, neighborhood, network, dimensionality, and class balance. These 22 measures, considered state-of-the-art for detecting class overlap and classification difficulty, are summarized in **Exhibit 2.1**, on the following pate. Full names, acronyms, and descriptions are provided in **Appendix A**.

Figure 2.5 illustrates the six complexity categories with stylized examples, adapted but then modified from [7] . Low-complexity datasets (top row) exhibit clear class separation and simple decision boundaries, while high-complexity datasets (bottom row) show overlapping classes, non-linear boundaries, and intricate topological structures. These visual differences translate into values calculated for the 22 Lorena et al. measures.

Figure 2.5 Examples of Low and High Dataset Complexity by Category



2.4. Evaluation Procedures

Model performance was evaluated using stratified 5-fold cross-validation repeated five times, yielding 25 independent measurements for each model–dataset combination. Stratification preserved binary class distributions across all folds, ensuring representativeness even in imbalanced targets. Each model was trained on one portion of the dataset, and its accuracy was measured by predicting the target variable on a separate, unseen portion of data that was reserved solely for testing. In total, the strategy produced 2,384 performance observations (15 models × 159 datasets × 1 average per combination, minus one SVC dataset pair[4]).

[4] SVC could not complete training and testing on one of the datasets.

Exhibit 2.1 Dataset Complexity Measures

Complexity Measures

Feature-based

- **F1**: Maximum Fisher's discriminant ratio – overlap between the values of the features in different classes
- **F1v**: Directional-vector Maximum Fisher's Discriminant Ratio – version of F1 that searches for a separating vector
- **F2**: Volume of overlapping region – overlap of the distributions of the features values within the classes
- **F3**: Maximum individual feature efficiency – maximum value of most discriminative feature
- **F4**: Collective feature efficiency – combined discriminative power of features

Linearity

- **L1**: Sum of the error distance by linear programming – ability to separate classes with a linear boundary.
- **L2**: Error rate of linear classifier – misclassification rate under linear separation.
- **L3**: Nonlinearity of a linear classifier – error rate on new data points

Neighborhood

- **N1**: Fraction of borderline points – proportion of samples near class boundaries.
- **N2**: Ratio of intra/extra class nearest neighbor distance – compares the distances inside a class with those between classes

(continued above)

Neighborhood (continued)

- **N3**: Error rate of nearest neighbor classifier – ratio of misclassified points using leave one out 1-NN
- **N4**: Nonlinearity of nearest neighbor classifier – ratio of the misclassified interpolated points
- **T1**: Fraction of hyperspheres covering data – how intermixed the different classes are
- **LSC**: Local set average cardinality – average of the set of points whose distance is smaller than the distance of the other class

Network

- **Density**: Average density of the network – density relative to dimensionality
- **ClsCoef**: Clustering coefficient – the tendency to create cliques
- **Hubs**: Network density – graph connectedness of the vertices

Dimensionality

- **T2**: Average number of features per points – ratio of dimensionality to number of samples
- **T3**: Average number of PCA dimensions per points – ratio of PCA components to number of samples
- **T4**: Ratio of the PCA dimension to the original dimension – proportion of relevant dataset dimensions

Class imbalance

- **C1**: Entropy of classes proportion– normalized entropy of the class size distribution
- **C2**: Imbalance ratio – and index of class imbalance

Three performance metrics were recorded for each model-dataset pair:

- **Accuracy** (proportion of correct predictions)
- **F1-weighted score** (harmonic mean of precision and recall, weighted by class support)
- **AUC** (area under the ROC curve).

Preliminary analysis revealed very high correlation between accuracy and F1-weighted scores (Pearson's $r = 0.988$, $R^2 = 0.976$), indicating these two performance metrics were redundant for this dataset collection. Accuracy was selected as the primary performance metric due to its interpretability and the demonstrated equivalence with F1 in model rankings (detailed in Section 3).

Statistical significance testing used the Friedman test to detect overall performance differences across models, followed by the Nemenyi post-hoc test for pairwise comparisons when differences were significant. The Friedman test is a non-parametric alternative to repeated-measures ANOVA, appropriate for comparing multiple algorithms across multiple datasets without assuming normally distributed performance scores. Critical distance diagrams and compact letter groupings show statistically equivalent model groups at $\alpha = 0.05$.

Computational efficiency was measured as prediction throughput (predictions per second) and peak RAM usage during model training and prediction. Throughput was calculated by running predictions five times and dividing total predictions by elapsed time, reducing interference from background tasks. Measurements were taken across all test folds within each cross-validation run to ensure representative timing. Preprocessing was excluded, and all timing was conducted on consistent hardware (specifications in Appendix D) for comparability.

All numerical features were standardized to zero mean and unit variance prior to model training, ensuring fair comparison across scale-sensitive (linear models, SVC, KNN, discriminant analysis, MLP) and scale-invariant (tree ensembles) algorithms. Hyperparameters were optimized using grid search with stratified 3-fold cross-validation on each training set. Search ranges for each model were defined based on scikit-learn documentation defaults and common practice, balancing computational feasibility with performance optimization. Optimal hyperparameters were selected independently for each dataset, allowing models to adapt to dataset-specific characteristics while maintaining consistent search procedures across all experiments.

2.5. Benchmarking Workflow

The benchmarking workflow followed sequential tasks, each building on the previous to ensure reproducibility. These activities are outlined below:

1. **Dataset Acquisition**
    - Locate and download datasets from public repositories (OpenML, UCI, Kaggle)
    - Log metadata (source, license, citation)
    - Convert to CSV format
    - Inspect structure and verify integrity

2. **Data Preprocessing**
    - Identify feature types (numeric, categorical)
    - Remove non-informative features (unique identifiers, constant values)
    - Extract temporal components from datetime features (year, month, day, hour)
    - Detect and remove duplicates
    - Clean numeric features (remove non-numeric characters)
    - Drop rows with missing values
3. **Feature Engineering**
    - One-hot encode categorical features
    - Standardize numerical features (zero mean, unit variance)
    - Log all transformations for reproducibility
4. **Target Standardization**
    - Normalize target column name to "target"
    - Binarize target values (0 or 1)
    - Verify class distribution for stratification
5. **Dataset Complexity Measurement**
    - Calculate 22 complexity measures defined in the Lorena et al. framework: [1]
        1. Feature-based (5 measures)
        2. Linearity (3 measures)
        3. Neighborhood (6 measures)
        4. Network (3 measures)
        5. Dimensionality (3 measures)
        6. Class imbalance (2 measures)
    - Compute summary statistics and distributions
6. **Model Training and Evaluation**
    - Train 15 ML algorithms across 159 datasets
    - Apply stratified 5-fold cross-validation, repeated 5 times (25 measurements per model-dataset pair)
    - Perform grid search hyperparameter tuning
    - Record performance metrics (accuracy, F1-weighted, and ROC-AUC)
    - Record computational metrics: throughput (predictions per second) and peak memory usage (RAM)
7. **Statistical Analysis**
    - Calculate Pearson correlations (complexity measures vs. performance)
    - Perform Random Forest and Linear Regression feature importance analysis
    - Perform stratified complexity analysis (low, medium, high)
    - Develop model selection guide by complexity type
    - Conduct Friedman test for overall model differences
    - Conduct post-hoc Nemenyi test for pairwise model comparisons

- Generate critical difference diagram and statistical grouping letters

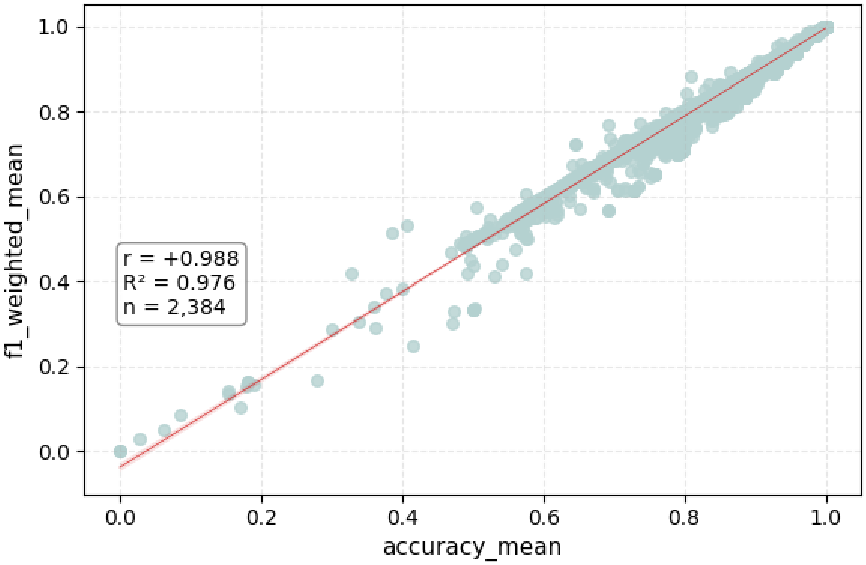
8. **Visualization and Synthesis**
    - Create correlation matrices, boxplots, scatter plots, bar charts
    - Develop model performance tables by complexity category
    - Generate scatter plots for validation
    - Compile computational efficiency comparisons
    - Document findings and conclusions.

3. Performance Metric Validation

Each model–dataset pair was evaluated using three metrics: accuracy, F1-weighted score, and area under the ROC curve (AUC). To assess redundancy and select a primary metric, we examined correlations across all 2,384 model–dataset combinations.

Accuracy and F1-weighted scores were highly correlated (Pearson's *r* = 0.988, R^2 = 0.976), indicating that both metrics ranked models similarly. **Figure 3.1** illustrates this relationship, with each point representing a model–dataset pair and a trend line illustrating the near-linear association.

Figure 3.1 Accuracy vs. F1-Weighted Score Across All Model–Dataset Pairs



Given the redundancy, accuracy was chosen as the primary metric. It is easy to interpret and aligns with F1-weighted rankings, making it suitable for summarizing model performance across diverse datasets.

AUC was retained for reference but not used as the primary metric due to interpretability and availability constraints. Although AUC reflects ranking quality independent of threshold, it is less intuitive for binary tasks with balanced classes. Additionally, three models (LinearSVC,

SVC, and SGDClassifier) do not produce probability estimates required for AUC, making it unavailable for those model–dataset pairs.

4. Model Performance

This section is organized as follows:

4.1 Model Performance

4.2 Statistical Significance Testing

4.3 Memory Efficiency and Stability.

4.1. Model Performance

Figure 4.1 summarizes model performance across 159 datasets, ordered by mean accuracy.[5]

Figure 4.1 Model Performance

Model	Accuracy	Median	Std. Dev. (pp)	Min.	Max.
XGBoost	**86.1%**	**88.3%**	12.6%	6.2%	100.0%
Random Forest	85.9	87.3	12.5	15.2	100.0
LightGBM	85.8	87.0	11.7	47.1	100.0
MLP	85.6	87.6	12.1	37.6	100.0
Extra Trees	85.5	87.1	13.7	0.0	100.0
SVC	85.1	87.2	12.6	17.1	100.0
GBM	85.0	86.9	13.9	0.0	100.0
Decision Tree	83.1	84.4	13.4	2.9	100.0
KNN	83.1	85.0	13.7	8.6	100.0
Logistic Regression	83.0	85.3	12.4	**50.0**	100.0
Linear SVC	82.7	84.9	13.3	15.2	100.0
SGD	81.9	84.2	13.2	33.8	100.0
LDA	81.4	83.9	13.6	18.1	100.0
QDA	81.1	82.8	14.1	0.0	100.0
Naive Bayes	72.7	73.7	**17.2**	18.0	100.0

Fourteen of the fifteen models (excluding Gaussian Naive Bayes) were tuned using standard hyperparameter optimization procedures. On average, tuning improved accuracy by 1.38 percentage points across the 159 datasets, and the reported accuracy values reflect results from these tuned models.

[5] The F1 macro weighted score for each model averaged 1.0 percentage points different from its accuracy score. The differences ranged from 0.5 percentage points less than accuracy to 1.4 percentage points less than accuracy.

Four key patterns emerge from these results.:

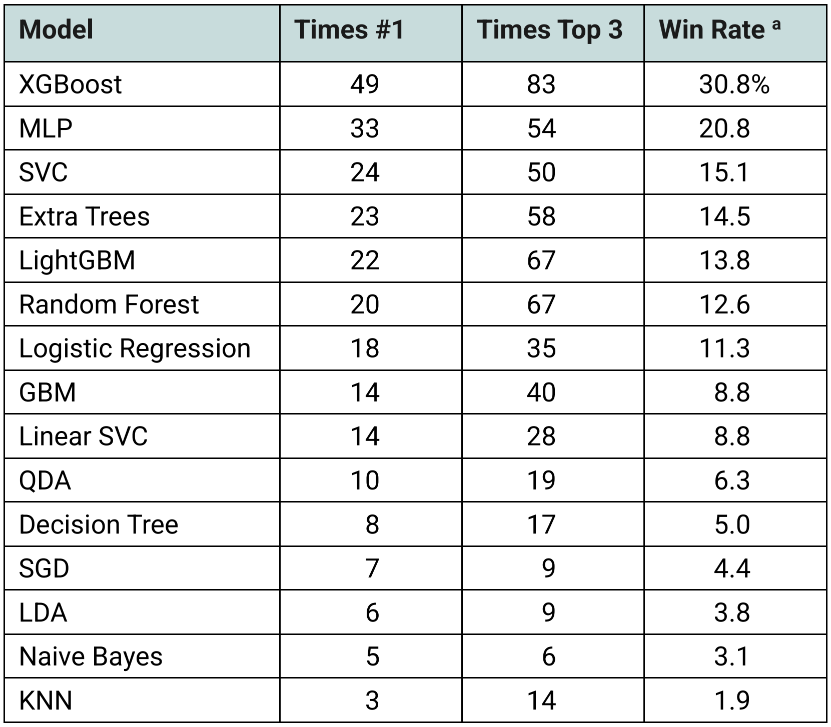
- **Dominance of ensemble and non-linear models**: The top seven models (XGBoost down through GBM) are all highly effective ensemble methods or non-linear, generally outperforming the best pure linear model by 2–3 percentage points (Logistic Regression, 83.0%).
- **Comparable stability**: Most models show similar variability (ranging from 12.1 - 14.1 pp), indicating they all experience a similar degree of performance variance across the 159 datasets. This suggests that dataset difficulty affects most algorithms similarly, rather than creating model-specific challenges.
- **High potential**: Every model achieved perfect accuracy on at least one dataset, confirming that the benchmarking datasets include problems that are easily separable.
- **Weak baseline performance of Naive Bayes**: Naive Bayes is the clear outlier, with the lowest mean accuracy (72.7%) and highest variability (17.2 pp). Struggling with a large portion of the datasets is likely due to violations of its core assumption of feature independence.

Twelve of the 15 models have mean accuracies within a span of five percentage points (≈81% to 86%). All models achieved perfect accuracy on at least one dataset, and all struggled uniformly on the hardest datasets. This convergence, despite models representing fundamentally different learning algorithms, suggests that modern machine learning has largely found effective, general-purpose solutions for tabular binary classification. While statistical tests (Section 4.2) confirm some distinct performance groups for the 15 models, the data indicates the datasets intrinsic complexity, not the choice of algorithm, is the primary factor determining classification performance.

Figure 4.2 summarizes model ranking frequencies across the 159 datasets, reporting the number of times each model ranked first, the frequency of Top-3 finishes, and the overall win rate. Several patterns emerge:

- **XGBoost as the most frequent winner**: XGBoost ranked first on 49 datasets (30.8% win rate) and appeared in the Top 3 83 times, confirming its consistent dominance.
- **Strong but secondary performers**: MLP (33 wins, 20.8%), SVC (24 wins, 15.1%), Extra Trees (23 wins, 14.5%), LightGBM (22 wins, 13.8%), and Random Forest (20 wins, 12.6%) all achieved frequent top placements, though all trailed XGBoost in win rate.
- **Competitive linear baselines**: Logistic Regression secured 18 wins (11.3%) and 35 Top 3 finishes, showing that despite lower mean accuracy, it occasionally outperforms non-linear methods on certain datasets. Linear SVC matched GBM with 14 wins (8.8%), though with fewer Top 3 appearances.
- **Lower-tier models**: QDA, Decision Tree, SGD, LDA, Naive Bayes, and KNN each ranked first on fewer than 10 datasets with win rates below 6.5%. KNN was the least frequent winner, ranking first only three times (1.9%).

Figure 4.2 Model Ranking Frequencies



Model	Times #1	Times Top 3	Win Rate [a]
XGBoost	49	83	30.8%
MLP	33	54	20.8
SVC	24	50	15.1
Extra Trees	23	58	14.5
LightGBM	22	67	13.8
Random Forest	20	67	12.6
Logistic Regression	18	35	11.3
GBM	14	40	8.8
Linear SVC	14	28	8.8
QDA	10	19	6.3
Decision Tree	8	17	5.0
SGD	7	9	4.4
LDA	6	9	3.8
Naive Bayes	5	6	3.1
KNN	3	14	1.9

[a] Win rate = Times #1 / 159 datasets

For each dataset, the accuracy of each model was determined, and the models were ranked 1 (highest accuracy) to 15 (lowest accuracy). Following McElfresh et al. [8] and Ye et al. [9], an "average rank" then was determined for each model by summing their ranks across datasets and dividing by 159**. Figure 4.3**, on the following page, identifies the average rank of each model across all datasets.

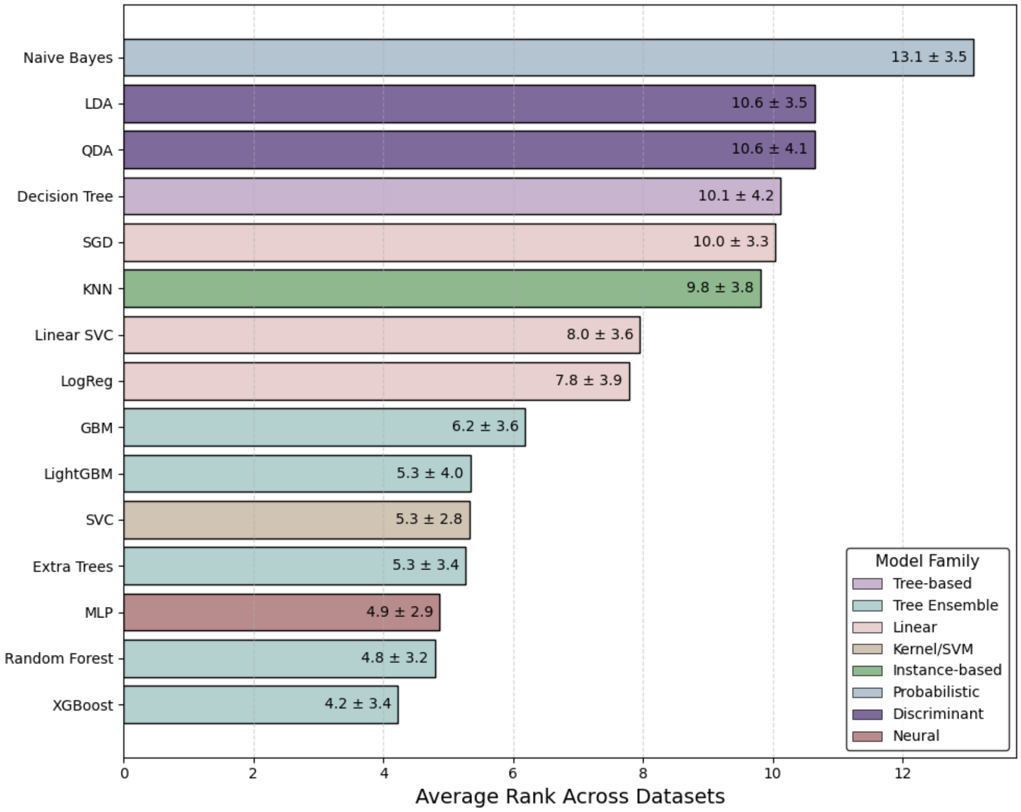
The top seven models cluster within a narrow band (average ranks 4.2 to 6.2). Their overlapping standard deviations indicate these models frequently traded positions across datasets. This suggests that any top-tier model could rank first on a substantial share of datasets. Performance differences are driven more by dataset characteristics than by algorithmic superiority. A clear gap separates the top tier from lower-performing models (Decision Tree through Naive Bayes, ranks 9.8 to 13.1).

4.2. Statistical Significance Testing

he Friedman test assessed whether the 15 models showed statistically significant performance differences across the 159 datasets. The test evaluates whether observed rank differences exceed what would be expected by chance. A significant result ($p < 0.05$) indicates that meaningful performance variation exists, which warrants pairwise comparisons through the Nemenyi post-hoc test.

The Friedman test revealed clear differences among models ($\chi^2 = 846.45$, df = 14, $p < 0.001$), rejecting the null hypothesis of equal performance. This outcome confirms that meaningful differences exist and justifies an evaluation of specific model pairs.

Figure 4.3 Average Model Rank Across 159 Datasets (Shorter Is Better)



To determine which models differ significantly, the Nemenyi post-hoc test was applied. This rank-based method adjusts significance thresholds to account for multiple comparisons (family-wise error rate) across all 105 model pairs (15 × 14 / 2). Model pairs whose average rank difference exceeded the critical threshold at $\alpha = 0.05$ were considered statistically distinct.

The Nemenyi test found 72 of 105 pairs (68.6%) to be significantly different. XGBoost, the highest-ranked model, differed significantly from eight models: Decision Tree, Logistic Regression, SGD, Linear SVC, KNN, Naive Bayes, LDA, and QDA (all $p < 0.001$). However, XGBoost showed no significant difference from six other models (Random Forest, Extra Trees, LightGBM, SVC, MLP, and GBM, all $p > 0.05$ except GBM at $p = 0.014$), indicating that these seven models form a statistically equivalent top tier (Group A).

Figure 4.4, on the following page, presents the statistical equivalence groups, known as compact letter display (CLD), derived from the Nemenyi post-hoc test. Models within each CLD do not differ significantly in performance at $\alpha = 0.05$, indicating statistically interchangeable performance within each tier. Group A contains the seven highest-ranked models, all statistically indistinguishable from one another. Group B represents moderate-performing models, while Group C consists of the lowest-performing model. Models in higher groups significantly outperform those in lower groups, establishing a clear performance hierarchy.

Figure 4.4 Compact Letter Display (CLD) from Nemenyi Post-Hoc Test

Group	Models	Mean Accuracy Range
A	XGBoost, Random Forest, LightGBM, MLP, Extra Trees, SVC, GBM	0.850–0.861
B	Decision Tree, KNN, Logistic Regression, Linear SVC, SGD, LDA, QDA	0.811–0.831
C	Naive Bayes	0.727

4.3 Memory Efficiency and Stability

Beyond accuracy, models were also evaluated for memory efficiency. Model memory footprint is a critical factor in real-world machine learning deployment, directly impacting cloud compute costs and system scalability. However, absolute maximum RAM usage is not a reliable comparison point, as it is influenced by many external factors. These include the dataset's size, data types, and sparsity; the operating system, library implementations, hardware configuration, available RAM, and background processes or memory management overheads. To provide a fairer assessment, two comparative measures were used:

- ***Multiple of Minimum RAM***: For each dataset, the ratio of a model's RAM usage to the minimum observed was calculated and then averaged across all 159 datasets. This shows how much more memory a model typically requires compared to the most efficient option. Values close to 1.0 indicate near-optimal efficiency, while larger values point to models that consistently demand more resources.
- ***Stability Factor***: For each dataset, overhead was defined as the difference between a model's RAM usage and the minimum observed for that dataset. A model's Stability Factor is the standard deviation of these overhead values across all 159 datasets. A low value means predictable memory use, while a high value indicates large swings in demand and greater risk in deployment.

The results in **Figure 4.5**, on the following page, reveal a trade-off between efficiency (cost) and stability (risk). Several models, including SVC, MLP, Naive Bayes, KNN, GBM, and LDA, required less than 1.34 times the minimum RAM, making them efficient relative to other models. Among these, KNN and LDA stood out as the most stable, with Stability Factors near 1.0, indicating highly predictable memory use. At the other extreme, Random Forest, Decision Tree, and Extra Trees were both the most inefficient and most unstable of all models, with RAM usage three to five times more variable than the most stable models. This combination of high overhead and volatility suggests that, despite strong accuracy, these methods may require costly hardware buffers for reliable production use.

Figure 4.5 Model RAM Efficiency and Stability Factors (most efficient first)

Model	Multiple of Minimum RAM	Stability Factor	Family
SVC	**1.32**	1.82	Kernel/SVM
MLP	1.33	1.79	Neural
Naive Bayes	1.33	1.65	Probabilistic
KNN	1.34	**1.00**	Instance-based
GBM	1.34	1.76	Tree Ensemble
LDA	1.36	1.17	Discriminant
QDA	1.37	1.49	Discriminant
LightGBM	1.39	1.83	Tree Ensemble
XGBoost	1.42	1.98	Tree Ensemble
Linear SVC	1.43	1.27	Linear
Logistic Regression	1.44	1.87	Linear
SGD	1.45	1.91	Linear
DecisionTree	1.45	3.78	Tree-based
RandomForest	1.68	3.19	Tree Ensemble
ExtraTrees	1.79	4.56	Tree Ensemble

5. Dataset Complexity Assessment

Dataset complexity was measured using 22 indicators from the Lorena et al. framework. [1] The analysis identifies which structural characteristics influence classification difficulty. Results show that geometric and neighborhood complexity drive classification difficulty more than dataset size or class imbalance.

This section is organized as follows:

5.1 Dataset Complexity Landscape

5.2 Dataset Complexity Impact on Model Performance

5.3 Class Imbalance Paradox: Evidence and Resolution.

5.1. Dataset Complexity Landscape

This project evaluates model performance across 159 binary classification datasets drawn from public repositories. Understanding dataset complexity helps explain variation in model performance.

Dataset complexity was measured using 22 indicators defined by Lorena et al. (2019) [1], grouped into six categories:

- **Feature-based**: discriminative power of individual attributes
- **Linearity**: decision boundary complexity

- **Neighborhood**: local class overlap and boundary definition
- **Network**: graph-theoretic connectivity patterns
- **Dimensionality**: effects of high-dimensional spaces
- **Class imbalance**: representation of minority classes.

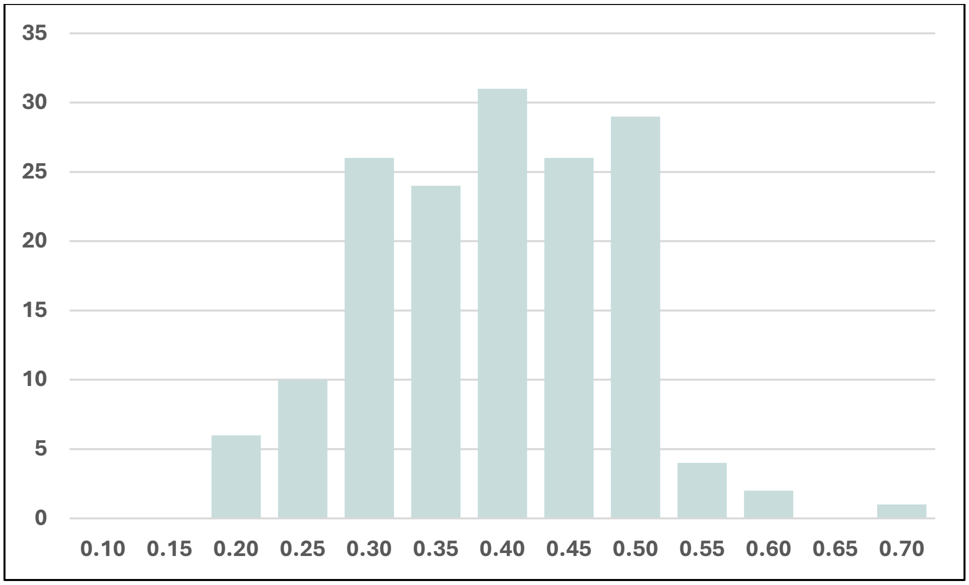
These measures are described in **Appendix A** to this report. The measures capture geometric, topological, and distributional properties that affect how easily algorithms can separate classes. An overall complexity score also was computed for each dataset as the arithmetic mean of all 22 standardized measures. All measures were determined using the Python library *problexity*.

Exhibit 5.1, on the following page, summarizes the distributions of all 22 complexity measures across the 159 datasets, grouped by category. There are substantial differences in geometric, topological, and distributional characteristics. These are discussed below.

1. Overall Difficulty and Key Challenges

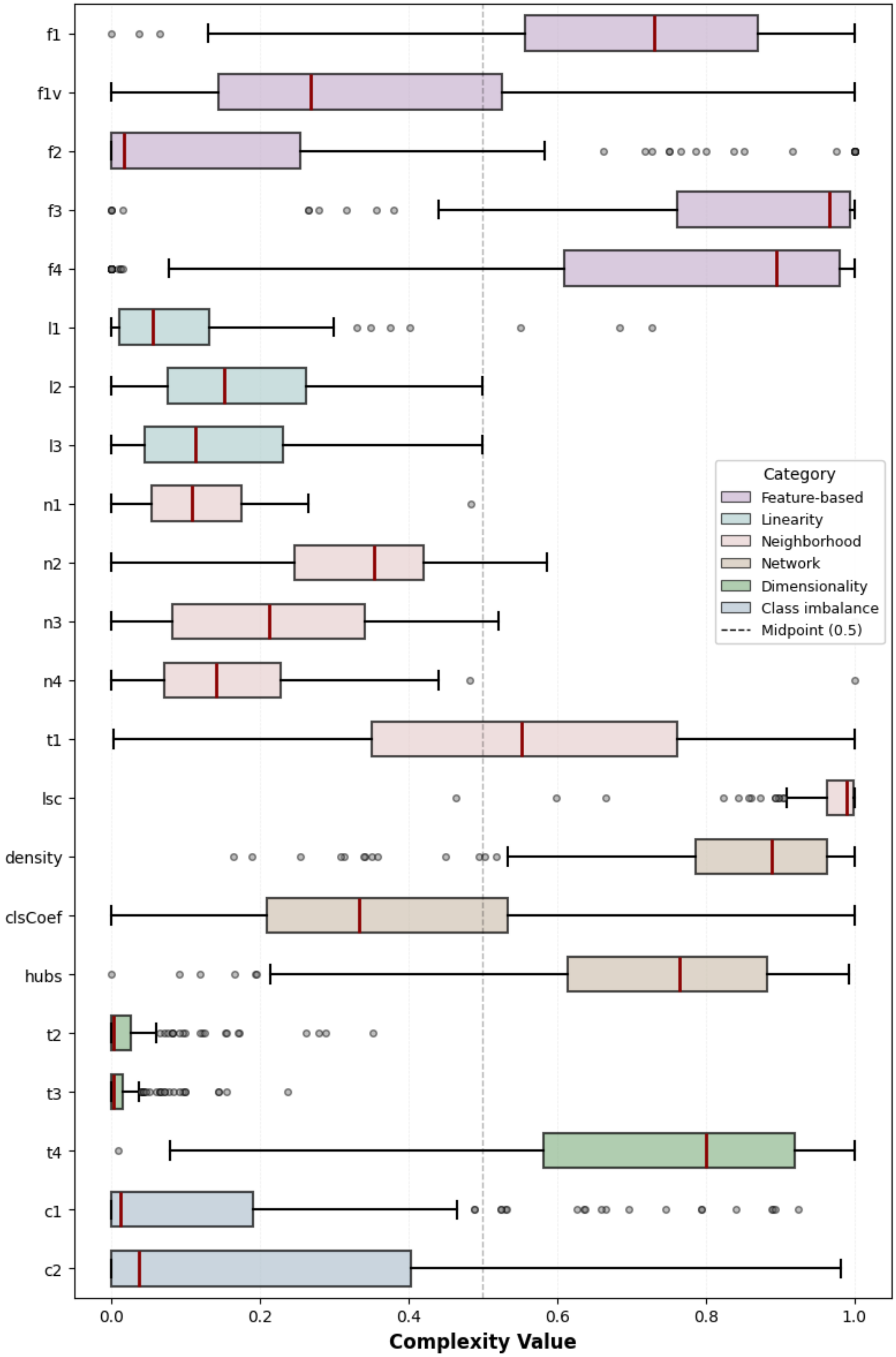
The overall complexity score (median = 0.40, IQR = 0.15) suggests that the benchmark collection consists mostly of moderate-difficulty datasets, with relatively consistent complexity across the 159 cases. The minimum overall score is 0.181, the maximum is 0.683. **Figure 5.1** displays the distribution of 159 benchmark datasets by their total complexity score.[6]

Figure 5.1 Distribution of Benchmark Datasets by Complexity Score



[6] Complexity "score" is the arithmetic mean of the dataset's 22 complexity measure values.

Exhibit 5.1 Distribution of Complexity Measures Across 159 Datasets



While the datasets span the full range of structural difficulty, they are drawn from curated repositories (UCI, Kaggle, OpenML) and therefore lack many of the irregularities of real-world data, such as significant missing values, mislabeled targets, noisy categorical features, redundant features, feature interaction, and concept drift. These datasets serve as a controlled environment for comparing machine learning models, which is not the same as working with raw, messy data.

Complexity varies substantially across the six categories, with network and feature-based measures emerging as the dominant challenges. These are the only two categories above the overall complexity median (category medians = 0.66 and 0.58, respectively). The primary obstacles to classification accuracy stem from graph-theoretic connectivity patterns and feature ambiguity rather than dataset size, class imbalance, or boundary linearity.

2. Dominant Challenges: Network and Feature Complexity

Network and feature-based complexity are the primary classification barriers, as the only categories exceeding the overall median. Network measures indicate that graph-theoretic connectivity patterns create difficult separation problems when class-specific subgraphs overlap or exhibit weak clustering. Feature-based complexity reflects that individual attributes often lack sufficient discriminative power to distinguish classes effectively. These intrinsic data characteristics drive classification difficulty more than surface-level properties like dataset size or class distribution.

3. Moderate Barriers: Neighborhood Structure

Neighborhood complexity is near the overall median, reflecting moderate challenges from local class overlap and boundary ambiguity. Among all 22 measures, LSC (local set cardinality) has the highest median complexity and shows notable stability. Its median value is roughly four times greater than the other neighborhood measures, indicating that the datasets consistently feature narrow, irregular decision boundaries.

The remaining neighborhood measures (N1–N4, T1) highlight variability in how instances cluster near boundaries: some datasets show substantial class overlap, while others maintain clearer separation. This variability makes neighborhood structure a key factor in explaining why dataset difficulty differs across the benchmark.

4. Limited Obstacles: Linearity, Dimensionality, and Class Imbalance

Linearity, dimensionality, and class imbalance all fall well below the overall complexity median, making them less common sources of difficulty in the benchmark. Linearity measures cluster at low values (median = 0.107, std = 0.124), indicating that most datasets require non-linear decision boundaries but avoid the extreme convolution that would demand highly flexible separators. Dimensionality measures also remain modest for typical datasets (median = 0.269), though extreme variability (CV = 200% for T2 and T3) reveals that a small subset experiences pronounced curse-of-dimensionality effects while the majority does not.

Class imbalance shows the lowest median of all categories (0.025) yet shows the highest variability (std = 0.277). Most datasets maintain balanced or near-balanced distributions (Q1 = 0.00), but long right tails extend to severe imbalance in outlier cases. This heterogeneity underscores that the collection contains two distinct dataset types: a majority with balanced classes and a minority with severe imbalance.

5.2. Dataset Complexity Impact on Model Performance

Section 5.1 profiled the complexity profile of the 159 datasets. This section examines which complexity dimensions most strongly influenced classification difficulty by correlating each with model accuracy. These relationships address the second research question: what structural traits make datasets difficult to classify?

Pearson correlation coefficients were calculated by pairing each complexity measure with model accuracy across all 2,384 model-dataset pairs (15 models × 159 datasets, minus one SVC-dataset pair). This model-agnostic approach pools results across all 15 models to identify dataset complexity traits that consistently influence classification difficulty. The results show that geometric and topological complexity, particularly neighborhood overlap and decision boundary non-linearity, dominate classification difficulty, while traditionally emphasized factors like class imbalance and dataset size show surprisingly weak and positive relationships with performance.

Pearson's correlation coefficient (*r*) was used to quantify the relationship between dataset complexity and model performance. For each of the 22 dataset complexity measures, *r* was calculated by correlating the measure's value with accuracy across all 2,384 model-dataset pairs.

The formula for Pearson's *r* is:

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\ \sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

Where:

x_i = complexity measure value (e.g., N1, L2, C1) for a given dataset

y_i = model performance (accuracy) for a specific model-dataset pair

$\bar{x}$ = mean complexity measure value across all 2,384 pairs

$\bar{y}$ = mean model performance across all 2,384 pairs

n = number of paired observations (2,384 model-dataset pairs:
15 models × 159 datasets, minus one SVC dataset pair)

Statistical significance was assessed using the *t* statistic:

$$t = r\ \frac{\sqrt{(n-2)}}{\sqrt{(1-r^2)}}$$

Where:

r is Pearson's correlation coefficient (calculated above)

n is number of paired observations (2,384)

df = degrees of freedom (n - 2 = 2,382)

The p-value was obtained from the t-distribution with 2,382 degrees of freedom using a two-tailed test. Correlations with $p < 0.05$ were considered statistically significant.

1. Correlations Between Complexity Measures and Model Accuracy

Figure 5.2 provides the correlation of each dataset complexity measure with model prediction accuracy. All correlations were statistically significant ($p < 0.0001$), confirming that each complexity measure relates to classification performance, though the strength and direction of these relationships vary substantially. Ten measures have **strong** correlations ($|r| \geq 0.40$), while the remaining twelve showed **weaker** associations.

Figure 5.2 Complexity Measure Correlations with Model Accuracy (all *p* <0.0001, N = 2,384) (Sorted in Descending Absolute Value)

Measure	Correlation	Strong
N1	(0.7768)	**Yes**
N3	(0.7712)	**Yes**
N4	(0.7326)	**Yes**
L2	(0.7089)	**Yes**
L3	(0.6874)	**Yes**
F1v	(0.6374)	**Yes**
T1	(0.5673)	**Yes**
F1	(0.4611)	**Yes**
L1	(0.4451)	**Yes**
N2	(0.4271)	**Yes**
F3	(0.3891)	No
F4	(0.3849)	No
C1	0.3582	No
C2	0.3478	No
F2	(0.3105)	No
LSC	(0.2870)	No
Density	(0.2237)	No
T3	(0.1933)	No
ClsCoef	(0.1812)	No
T4	(0.1687)	No
Hubs	(0.1010)	No
T2	(0.0831)	No

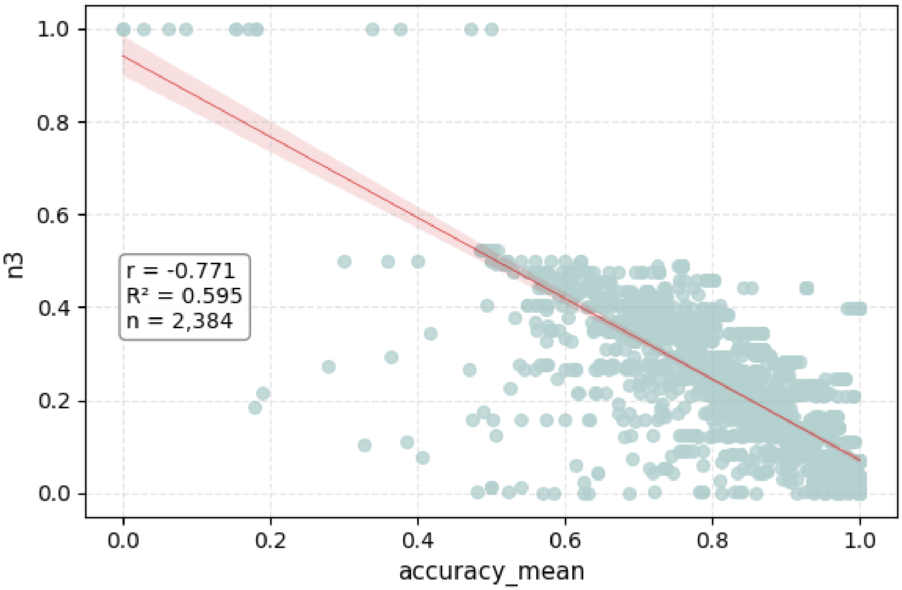
Neighborhood complexity emerged as the strongest predictor of classification difficulty, with three measures ranking highest: N1 (r = -0.78), N3 (r = -0.77), and N4 (r = -0.73). This confirms that datasets with extensive class overlap near decision boundaries consistently challenge models across all learning algorithms. Linearity measures followed closely, with L2 (r = -0.71) and L3 (r = -0.69) demonstrating that non-linear decision boundaries substantially increase classification difficulty. Feature-based measures showed moderate correlations (F1v: r = -0.64, F1: r = -0.46), indicating that weak individual feature discriminability contributes meaningfully to dataset difficulty, though less powerful than geometric properties.

Class imbalance measures showed a counterintuitive pattern: both C1 (r = +0.36) and C2 (r = +0.35) showed positive correlations with accuracy, indicating that higher imbalance associates with better performance rather than worse. This counterintuitive finding contradicts conventional wisdom that imbalanced datasets are harder to classify and requires deeper investigation (and addressed further in Section 5.3).

Dimensionality measures showed the weakest correlations with performance, with T2 (r = -0.08), T3 (r = -0.19), and T4 (r = -0.17) all well below the strong correlation threshold. This pattern suggests that curse-of-dimensionality effects, while present in isolated cases (as shown in Section 5.1), do not influence classification difficulty across the benchmark collection.

Figure 5.3 illustrates the relationship between neighborhood complexity measure N3 (error rate of nearest neighbor classifier) and model accuracy. N3 demonstrated a strong negative correlation with model accuracy (r = -0.771, p < 0.001), explaining nearly 60% (R^2 = 0.595) of the variance in performance. The remaining 40% reflects variation in how models handle neighborhood complexity.

Figure 5.3 N3 (Neighborhood Complexity) vs. Model Accuracy

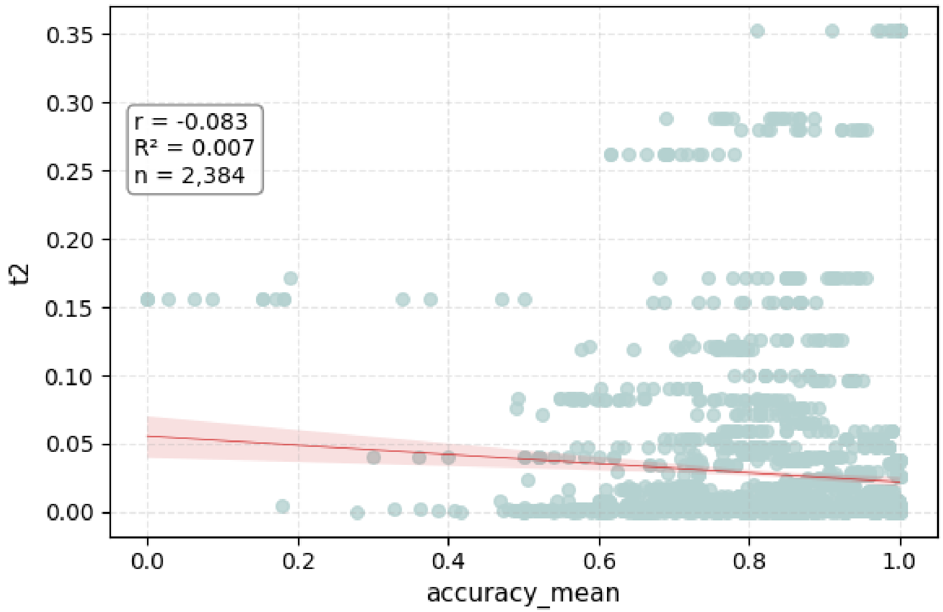


The N3 plot shows a single outlier dataset, where N3 = 1.00 (the horizontal row of dots at the top of the figure), with the next-highest value at N3 = 0.522. The outlier dataset (parity5) contains just 32 samples and 6 features, the smallest in the collection. While most models struggled with the dataset, two models recorded moderate accuracy (≈0.4-0.5): Logistic Regression and LightGBM. This divergence shows that some algorithms are more resilient to extreme complexity.

In contrast to N3, the dimensionality measure T2 (average number of features per point) demonstrated the weakest correlation of all 22 measures (r = -0.083, R^2 = 0.007). **Figure 5.4** displays the comparison. T2 reflects data sparsity in the feature space: when datasets have many features relative to sample size, data points become sparsely distributed in the input space.

Lower T2 values indicate less sparsity and therefore simpler problems. The nearly horizontal regression line and negligible R² confirm that T2 explains virtually none of the accuracy variation.

Figure 5.4 T2 (Average Number of Features per Point) vs. Model Accuracy



The T2 scatter plot reveals substantial horizontal spread at specific T2 values, most notably at T2 = 0.154 where a series of 15 dots represents a single dataset. On this dataset, models achieved highly divergent accuracies ranging from near-zero to moderate performance. This pattern confirms that while T2 itself does not determine difficulty, underlying geometric characteristics cause some models to struggle while others perform effectively on the same problem.

2. Validation Through Easy vs. Hard Dataset Comparison

To validate the correlation findings, datasets were stratified by model consensus. Eight datasets where at least 12 of 15 models achieved ≥99% accuracy were classified as "easy," while the remaining 151 datasets were classified as "hard." For each complexity measure, mean values were calculated separately for the easy and hard groups. The hard mean was divided by the easy mean to show how many times higher the measure was in hard datasets (e.g., 4.9× means hard datasets averaged 4.9 times higher). This stratification enables direct comparison of complexity characteristics between trivially easy and genuinely difficult problems.

Dataset size showed no consistent pattern distinguishing easy from hard datasets. Median record counts differed moderately (6,871 easy vs. 4,970 hard, a 28% gap). Feature counts presented mixed results: hard datasets *averaged* more features (54 vs. 27) yet had a lower *median* (16 vs. 28). These mixed results indicate that scale, whether measured by records or features, did not determine classification difficulty.

Dataset size did not explain difficulty, but complexity measure comparisons revealed a consistent pattern: geometric complexity (boundary shape and linearity) and topological complexity (neighborhood structure and connectivity) drive dataset difficulty. **Figure 5.5** shows that ten measures exhibited 2-5 fold increases in hard datasets, all statistically significant ($p \leq 0.05$).

Figure 5.5 Complexity Measures with Largest Correlation Differences Between Easy and Hard Datasets (r_{hard} / r_{easy})

Measure	Easy	Hard	Multiple	Complexity Category	*p*-value
L2	0.038	0.186	**4.9x**	Linearity	0.000
L3	0.032	0.157	**4.8x**	Linearity	0.001
L1	0.023	0.096	**4.2x**	Linearity	0.001
N3	0.058	0.226	**3.9x**	Neighborhood	0.001
N1	0.031	0.119	**3.9x**	Neighborhood	0.001
T1	0.153	0.572	**3.7x**	Neighborhood	0.000
N4	0.045	0.164	**3.7x**	Neighborhood	0.001
F1v	0.107	0.365	**3.4x**	Feature-based	0.001
F4	0.221	0.748	**3.4x**	Feature-based	0.001
F2	0.098	0.193	**2.0x**	Feature-based	0.019

Linearity measures showed the strongest effects (L2: 4.9×, L3: 4.8×, L1: 4.2×), followed closely by neighborhood complexity and feature-based complexity. The overall complexity score[7] (not shown in figure) increased 52% in hard datasets (0.264 to 0.400, $p < 0.001$), confirming that multiple complexity dimensions compound classification difficulty.

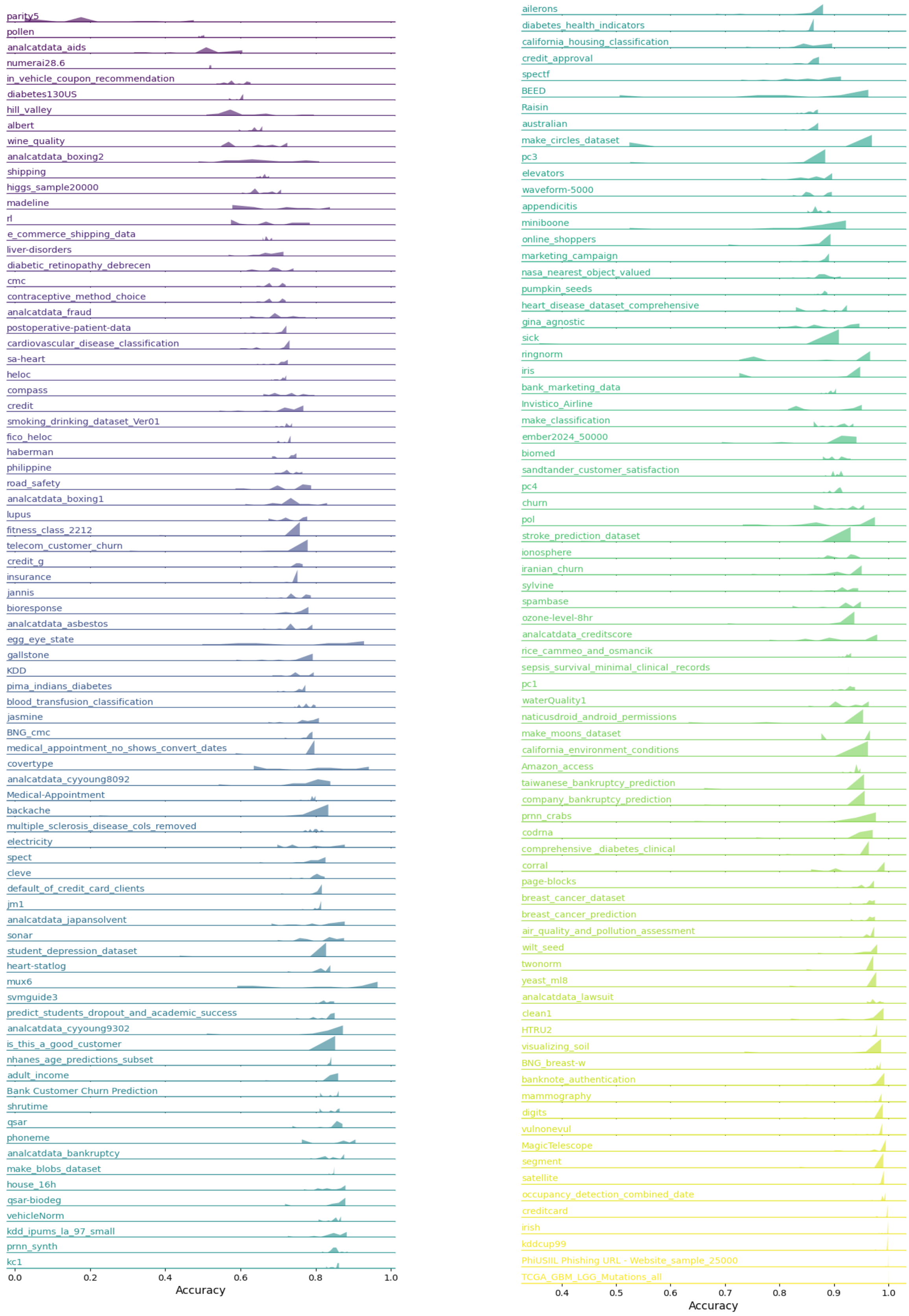
3. Distribution of Dataset Difficulty (Ridgeline View)

Exhibit 5.2, on the following page, presents ridgeline plots displaying all 159 datasets sorted by mean accuracy (hardest at top left, easiest at bottom right). Each ridge represents the distribution of accuracy achieved by the 15 models on a single dataset. Ridge width reveals the degree of model agreement or disagreement on prediction accuracy.

The collection spans a wide difficulty range: mean accuracy across datasets ranges from 0.18 (parity5) to 0.999 (PhiUSIIL). Model disagreement (standard deviation) varies substantially, ranging from 0.0001 to 0.221. The median minimum across datasets is 0.719, and the median maximum accuracy is 0.889, indicating that most datasets are solvable problems as even the weakest model achieves reasonable performance. This distribution validates the moderate-difficulty characterization established in Section 5.1.

[7] The overall complexity "score" produced by Python's problexity is the arithmetic mean of the 22 complexity measures.

Exhibit 5.2 Ridgeline Plots Showing Model Accuracy Distributions Across 159 Datasets



Ridge width patterns are minimally associated with accuracy (r = -0.16, p = 0.044). The easiest datasets consistently show narrow ridges with low standard deviations (e.g., irish: std = 0.006, PhiUSIIL: std = 0.002), reflecting near-perfect model consensus. In contrast, harder datasets exhibit mixed patterns with both high variance (parity5: std = 0.168) and low variance (numerai28.6: std = 0.004), indicating that some challenging problems affect all models equally while others reveal differences in algorithm strengths.

4. Model Family Sensitivity to Dataset Complexity

To assess whether complexity dimensions affect model families differently, Pearson correlations were calculated between each of the 22 complexity measures and accuracy for each of the eight model families separately. Results are presented in **Appendix D**.

This analysis reveals which complexity measures distinguish model algorithms and which pose more consistent challenges across all model types. The different sensitivities reflect model constraints: linear models struggle fundamentally with non-linear boundaries, instance-based methods depend critically on neighborhood structure, while ensemble methods' flexibility enables robust performance across diverse complexity types.

- **Universal Neighborhood Dominance.** N1, N3, and N4 show the strongest negative correlations across all families (r = −0.73 to −0.89). While the instance-based family is most sensitive (N3: r = −0.90), even the weakest responder, the probabilistic family (r = −0.47 to −0.49), still ranks neighborhood measures as top predictors.
- **Probabilistic Family as Outlier.** Probabilistic family exhibits dramatically weaker correlations (N1: r = -0.49) compared to other families (r = -0.78 to -0.89), reflecting Naive Bayes' strong independence assumption, which limits adaptability to specific complexity patterns.
- **Linear Models Most Sensitive to Linearity.** The linear family shows the strongest L2 and L3 correlations (r = −0.86, −0.84), reflecting these models' architectural constraint to linear decision boundaries.
- **C1/C2 Paradox is Universal.** All families display positive C1/C2 correlations except the probabilistic family (near zero), with most architectures clustering in the r = +0.35 to +0.44 range, reinforcing the cross-family consistency of this benchmark's paradox.
- **Dimensionally Complex Datasets Missing.** T2, T3, and T4 show the weakest correlations across all families (minimum r range = −0.06 to −0.26, and avg. r = 0.148). T2 and T3 complexity was virtually absent (means = 0.028, 0.017), while T4 showed substantial complexity (mean = 0.742). Despite T4's high complexity, all three measures showed weak correlations, suggesting dimensionality exerts limited influence on model performance in this collection.
- **Instance-Based Most Sensitive Overall.** KNN exhibits the highest absolute correlations across most measures, consistent with its reliance on local neighborhood structure where class overlap directly determines nearest-neighbor label accuracy.

- **Naive Bayes Drives Differential Effects.** The 14 largest variations in family correlations are driven only by Naive Bayes' weak correlations. Excluding NB reduces maximum range from 0.44 to 0.19, revealing the remaining model families respond relatively consistently to complexity dimensions.

5.3 Class Imbalance Paradox: Evidence and Resolution

An imbalanced binary classification dataset has significantly more examples of one class than the other. Class imbalance measures (C1, C2) showed unexpected positive correlations with model performance (r = +0.36, +0.35, p < 0.001), contradicting the conventional expectation that greater imbalance should make classification more difficult. However, comprehensive correlation analysis revealed this relationship to be spurious. **Figure 5.6**, on the following pate, shows that C1 has negative correlations with 20 of 22 other complexity measures. This pattern indicates that highly imbalanced datasets in this collection also showed lower geometric and topological complexity.

Correlation analysis confirmed the positive C1-performance relationship was spurious. C1 showed negative correlations with 20 of 22 complexity measures, including neighborhood complexity (N1: r = -0.47, N3: r = -0.43), linearity (L2: r = -0.47, L3: r = -0.38), and network structure (Density: r = -0.44, Hubs: r = -0.42). When C1 is high (more imbalance), geometric complexity tends to be low, resulting in easier classification. When C1 is low (more balanced), geometric complexity is higher, resulting in harder classification. The positive C1 correlation therefore reflects confounding with geometric simplicity rather than any benefit of imbalance.

Feature importance analysis supported this interpretation: C1 ranked 17th-18th of 22 measures with less than 1% combined importance, while geometric measures (N1, N3, L2) consistently ranked in the top seven across both Random Forest and Linear Regression methods (Section 5.4). These findings establish that geometric and topological complexity, not class distribution, determines classification difficulty. The weak influence of imbalance likely reflects ensemble methods' built-in mechanisms for handling skewed class ratios, including stratified sampling and class weighting. The C1 positive correlation is an artifact of this dataset collection: easier datasets happen to exhibit both high imbalance and low geometric complexity.

Both accuracy and F1 increased as class ratios became more extreme, with the most imbalanced datasets (95/5 majority/minority) showing the highest average performance. **Figure 5.7**, below Figure 5.6, summarizes these results. Accuracy and F1 also moved together across all bins (r = 0.988 across all model–dataset pairs), remaining nearly identical even under severe imbalance. If a model were simply predicting the majority class, the result would be high accuracy but low F1, because F1 penalizes ignoring the minority class. But the models were correctly identifying both majority and minority classes, not collapsing into majority-class prediction.

Figure 5.6 C1's Correlations with Each of the 22 Complexity Measures (in descending |*r*|)

Measure	Correlation (*r*)	*p*_Value	Significant?
C2	0.9722	0.000000	Yes
N1	(0.4741)	0.000000	Yes
L2	(0.4730)	0.000000	Yes
T1	(0.4466)	0.000000	Yes
Density	(0.4385)	0.000000	Yes
LSC	(0.4345)	0.000000	Yes
N3	(0.4288)	0.000000	Yes
Hubs	(0.4200)	0.000000	Yes
N4	(0.4115)	0.000000	Yes
L3	(0.3846)	0.000001	Yes
F3	(0.2711)	0.000548	Yes
F1v	(0.2619)	0.000854	Yes
F4	(0.2426)	0.002058	Yes
N2	(0.2379)	0.002528	Yes
F2	(0.2077)	0.008615	Yes
Score	(0.2045)	0.009711	Yes
ClsCoef	(0.1728)	0.029427	Yes
T4	(0.1605)	0.043244	Yes
F1	0.1130	0.156318	No
L1	(0.0939)	0.238990	No
T2	(0.0881)	0.269313	No
T3	(0.0786)	0.324686	No

Figure 5.7 Accuracy and F1 Scores by Imbalance Ratio

Ratio	Bin Count	Accuracy	Accuracy Std.	F1Score	F1Score Std.	Bin Percent
50/50	1,154	0.785	0.154	0.780	0.161	48.4%
60/40	285	0.856	0.107	0.849	0.116	12.0%
70/30	345	0.814	0.089	0.786	0.107	14.5%
80/20	285	0.881	0.080	0.863	0.086	12.0%
90/10	165	0.947	0.060	0.939	0.055	6.9%
95/5 or worse	150	0.966	0.080	0.966	0.063	6.3%
	2,384	**0.832**	**0.137**	**0.822**	**0.144**	**100.0%**

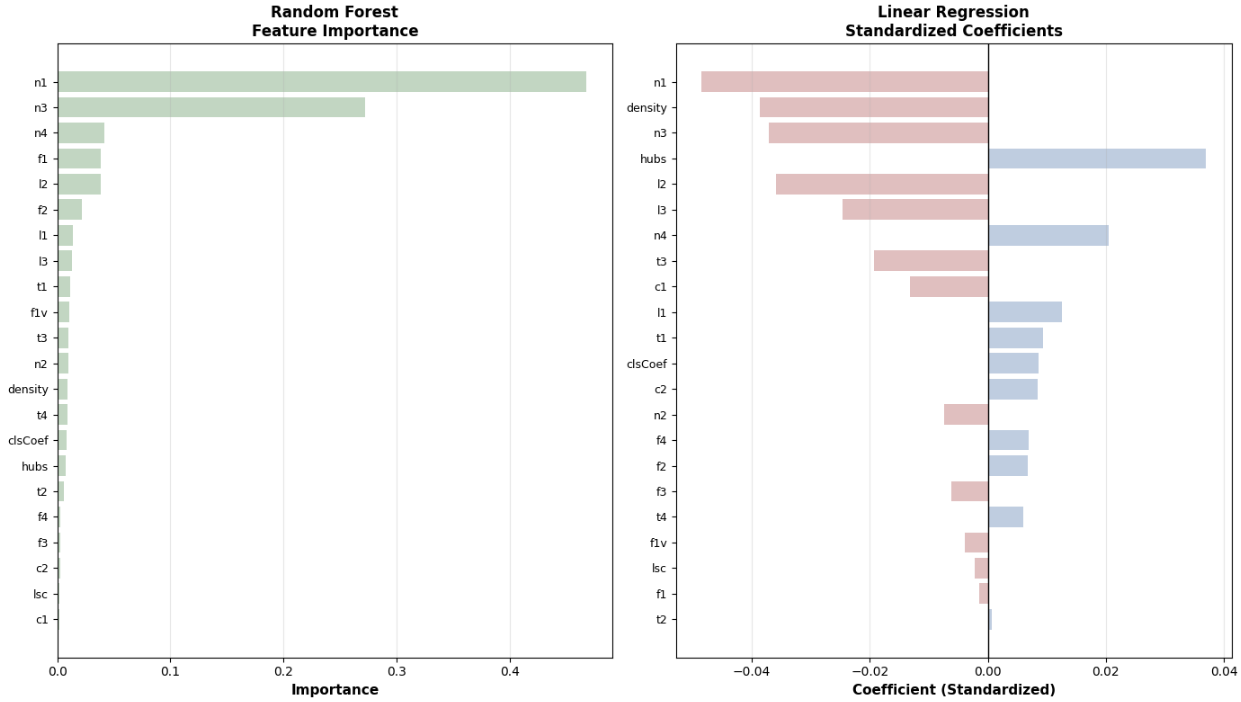
Performance became increasingly stable as imbalance grew, with accuracy and F1 showing much lower variability in the most imbalanced bins. This greater stability suggests that the highly imbalanced datasets are also easier to classify. It reinforces the earlier C1 result that imbalance in this collection tracks with lower geometric complexity.

5.4 Feature Importance Validation

To validate the correlation results, two feature importance methods were applied: Random Forest (RF) importance and Linear Regression (LR) standardized coefficients. RF captures nonlinear relationships and interaction effects, while LR highlights interpretable linear effects. Both methods independently assessed the influence of all 22 complexity measures on classification accuracy.

Figure 5.8 shows strong consensus on the most influential measures. N1 (neighborhood overlap) ranked first in both methods, accounting for about half of RF importance and showing the largest absolute coefficient in LR. The top six consensus measures, based on average rank, were N1, N3, L2, N4, L3, and Hubs (all ≤6.5 average rank). Neighborhood measures (N1, N3, N4) and linearity measures (L2, L3) accounted for five of the top seven positions. In contrast, class imbalance measures ranked much lower: C1 averaged 14.0 (19th in RF, 9th in LR) and C2 averaged 17.5 (22nd in RF, 13th in LR), confirming their weaker role once geometric complexity is considered.

Figure 5.8 Dataset Complexity Measures Feature Importance



Consensus rankings show broad agreement on the main drivers of classification difficulty, despite differences between RF and LR. Perfect alignment occurred for N1 (rank 1), L2 (rank 5), and L1 (rank 10). The largest disagreements occurred for Density (rank difference = 14) and F1 (rank difference = 18), illustrating how tree-based and linear methods can weigh variables differently. Overall, consistency across Pearson correlations, Random Forest, and Linear Regression shows that neighborhood complexity and decision boundary non-linearity are the strongest contributors to classification difficulty in the 159 datasets.

6. Model Performance by Dataset Characteristics

Section 5 established that geometric and topological complexity, particularly neighborhood overlap and decision boundary non-linearity, determine classification difficulty across the benchmark collection. This section examines how different models respond to these six complexity categories, identifying which models maintain performance under specific challenging conditions and which ones show fundamental limitations.

This section is organized as follows:

6.1 Model Performance by Complexity Category

6.2 Model Performance by Dataset Size

6.1 Model Performance by Complexity Category

To evaluate which models maintain performance under challenging conditions, all 15 models were assessed across the six complexity categories using the top 25% most complex datasets in each category (40 datasets per category). Performance on these high-complexity subsets was ranked and classified as Strong (top 5 models), Moderate (middle 5), or Weak (bottom 5) based on mean accuracy. This approach tests model robustness when facing substantial geometric, topological, or distributional challenges rather than overall benchmark performance.

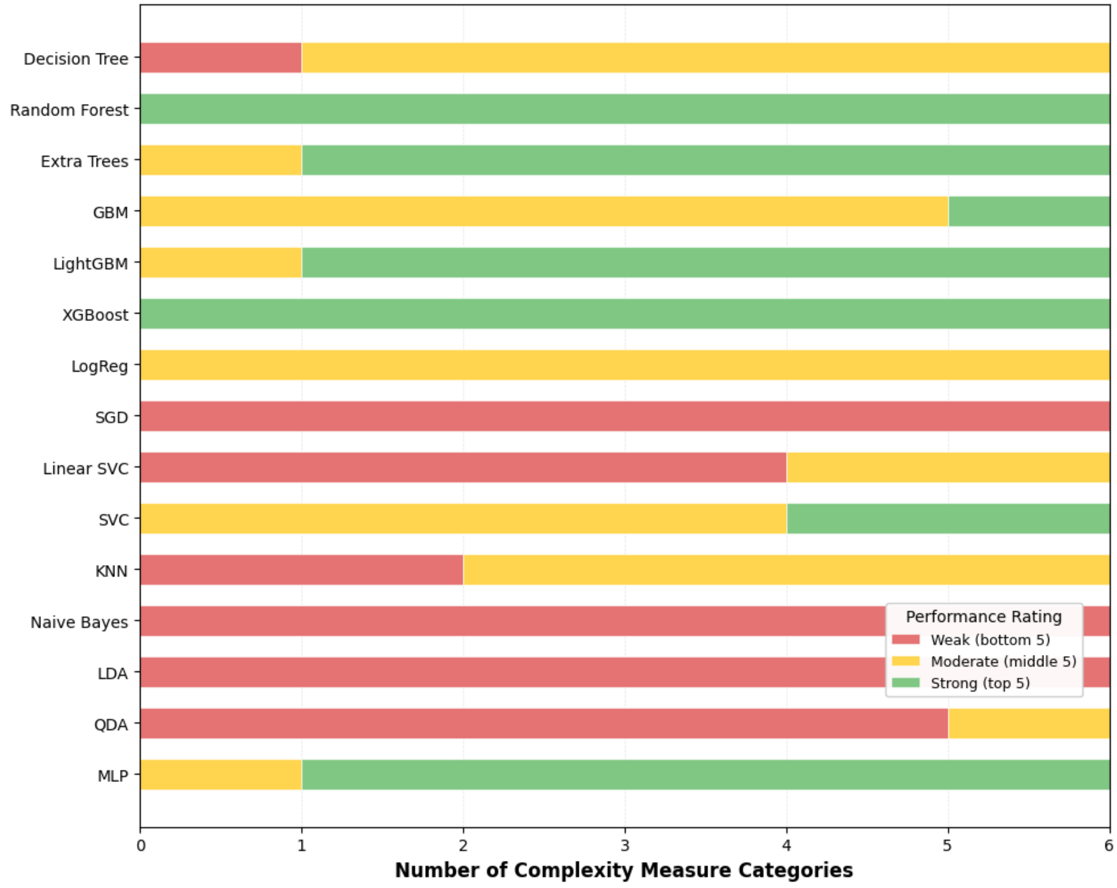
The analysis reveals a clear performance hierarchy (**Figures 6.1** and **6.2**). Five models consistently performed well across all complexity types: Random Forest and XGBoost achieved Strong ratings in all six categories (6/6), while Extra Trees, LightGBM, and MLP achieved Strong ratings in five of six categories (5/6). These top-tier models maintained performance regardless of complexity type, establishing them as reliable generalists. In contrast, SGD, Naive Bayes, and LDA received Weak ratings across all six categories (6/6), indicating consistent limitations under high-complexity conditions.

To quantify performance consistency, ratings were converted to numeric scores (Strong = 3, Moderate = 2, Weak = 1) and averaged by family. Model families showed consistent performance across all complexity categories, revealing no evidence of specialization. Top-tier families (Tree Ensemble: 2.6-3.0, Neural: 2.0-3.0) maintained high scores regardless of complexity type, while weak families (Probabilistic, Discriminant, Linear: ≤1.67 in all 6 categories) struggled uniformly. These results challenge the assumption that simpler models excel on easier datasets. In this benchmark, top models dominated across all complexity types, including those with low geometric complexity.

Figure 6.1 Model Strength on Most Complex Datasets (sorted by strong count, descending)

Model	Feature	Linearity	Neighbor-hood	Network	Dimen.	Class Imb.	Strong Count	Weak Count
Random Forest	Strong	Strong	Strong	Strong	Strong	Strong	**6**	0
XGBoost	Strong	Strong	Strong	Strong	Strong	Strong	**6**	0
Extra Trees	Strong	Strong	Strong	Strong	Moderate	Strong	5	0
LightGBM	Strong	Strong	Strong	Strong	Strong	Moderate	5	0
MLP	Strong	Strong	Strong	Moderate	Strong	Strong	5	0
SVC	Moderate	Moderate	Moderate	Moderate	Strong	Strong	2	0
GBM	Moderate	Moderate	Moderate	Strong	Moderate	Moderate	1	0
Decision Tree	Moderate	Moderate	Moderate	Moderate	Moderate	Weak	0	1
Log. Reg.	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	0	0
SGD	Weak	Weak	Weak	Weak	Weak	Weak	0	**6**
Linear SVC	Weak	Weak	Moderate	Weak	Weak	Moderate	0	4
KNN	Moderate	Moderate	Weak	Weak	Moderate	Moderate	0	2
Naive Bayes	Weak	Weak	Weak	Weak	Weak	Weak	0	**6**
LDA	Weak	Weak	Weak	Weak	Weak	Weak	0	**6**
QDA	Weak	Weak	Weak	Moderate	Weak	Weak	0	5

Figure 6.2 How Each Model Performs Under High Complexity Conditions



6.2 Model Performance by Dataset Size

This subsection evaluates how dataset size affects accuracy and stability, highlighting consistent performance patterns and notable exceptions. To evaluate model scalability, the 159 datasets were partitioned into three size buckets: small (≤500 records), medium (501–7,400), and large (>7,400). Each of the 15 models was assessed on its average performance within each bucket using mean accuracy and coefficient of variation (CV%) as metrics. This design tests whether models maintain accuracy and consistency as dataset size increases, or whether performance degrades on larger, more complex conditions. Buckets varied in sample count, feature dimensionality, and average complexity score, allowing for a robust comparison across scale. **Figure 6.3** presents a summary of the results.

Figure 6.3 Model Performance Summary by Dataset Size

Records	Datasets	Complex-ity Score	Mean Accuracy	Top 3 Models	Bottom 3 Models
≤500	35	0.401	79.4%	MLP, SVC, LightGBM	SGD, LDA, Naive Bayes
501-7,400	61	0.379	85.1%	XGBoost, LightGBM, Random Forest	LDA, QDA, Naive Bayes
>7,400	63	0.403	83.4%	XGBoost, Random Forest, Extra Trees	LDA, QDA, Naive Bayes
All	159	0.393	83.2%	XGBoost, Random Forest, LightGBM	LDA, QDA, Naive Bayes

Results show a consistent performance hierarchy across dataset sizes, with notable exceptions. On small datasets, MLP (0.828, CV% 16.2) and SVC (0.822, CV% 18.7) emerged as leaders, the only instance where non-ensemble models topped the hierarchy. From small to medium datasets, accuracy increased by roughly six points for every model, a strikingly consistent gain. Ensemble methods then dominated the medium bucket, with XGBoost (0.879, CV% 12.0), LightGBM (0.876, CV% 12.3), and Random Forest (0.875, CV% 12.2) clustered at the top.

On large datasets, accuracy declined slightly (1–3 points) for nearly all models except Decision Tree, confirming a stable scalability curve. Naive Bayes remained weakest throughout, with accuracy below 0.74 and CV% exceeding 20%. The medium bucket had the lowest average complexity score (0.379), slightly below both small (0.401) and large (0.403), which may explain why performance peaked on medium datasets.

The most consistent performers across all sizes were ensemble tree methods, which maintained high accuracy and low variability. This reinforces the broader finding: top-tier models are robust across complexity types and dataset scale. In contrast, models with strong parametric assumptions or limited capacity (e.g., Naive Bayes, LDA, SGD) showed persistent weaknesses. Overall, the scalability analysis highlights that dataset size magnifies model differences, with tree ensembles remaining the most reliable choice for large-scale classification tasks.

7. Model Performance versus Throughput

This section examines the computational costs of the 15 models, specifically predictions per second. It identifies which models are both fast and accurate, and which trade speed for predictive strength.

This section is organized as follows:

7.1 Model Throughput (Predictions per Second)

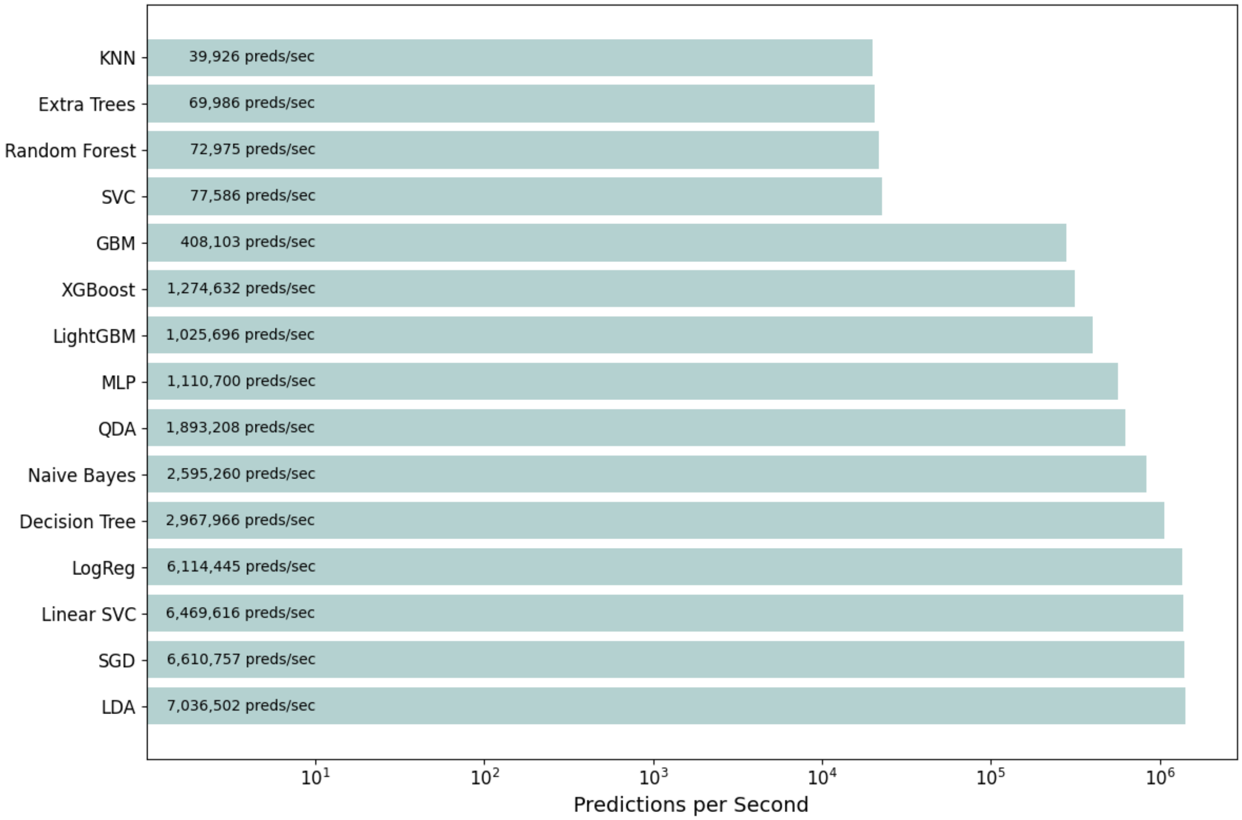
7.2 Performance-Throughput Trade-offs

7.3 Integrated Performance Score

7.1 Model Throughput (Predictions per Second)

Prediction speed is a critical factor for real-time and high-volume deployment scenarios. Each model's baseline throughput was measured as its average predictions per second. While actual speed depends on compute resources, the relative differences between models are key, revealing intrinsic computational efficiency. **Figure 7.1** displays these results, with significant variability immediately apparent on the logarithmic scale.

Figure 7.1 Model Predictions per Second ($\log_{10}$)



Throughput varied by as much as approximately 175× across models. Linear models (LDA, SGD, Logistic Regression) and Naive Bayes consistently achieved the highest prediction rates, all four are at least twice as fast as the next fastest group of models. Kernel-based methods (SVC) and neural networks (MLP) are among the slowest. Tree ensembles fall into a middle tier, with XGBoost and LightGBM outperforming Random Forest and Extra Trees in speed.

These differences reflect each model's underlying algorithm. Models with closed-form solutions or simple decision rules scale efficiently, while iterative and memory-intensive methods incur higher cost. Interestingly, the slowest models (KNN, Extra Trees, Random Forest, and SVC) share little in common beyond their computational bottlenecks. Their latency arises from different sources: brute-force distance calculations (KNN), deep ensemble traversal (Extra Trees, Random Forest), and kernel matrix operations (SVC). The range in model speed underscores the need to measure throughput directly rather than assume it from the model type.

In particular, SVC latency appears largely due to the use of the RBF kernel during hyperparameter tuning. GridSearch selected the RBF kernel for 119 of the 158 datasets [8], which requires costly kernel calculations.

7.2 Performance-Throughput Trade-offs

This subsection examines the trade-offs between predictive accuracy and prediction speed. While accuracy is often the primary goal, throughput may determine model choice in real-time or resource-limited deployments. The goal is to identify which models deliver strong performance without excessive computational cost, and which ones force a compromise.

The fundamental trade-off between model accuracy and computational efficiency is illustrated in **Figure 7.2.**, on the following page. **Figure 7.3** further details this relationship, using average model ranking and model family groupings to identify the specific clusters that represent the effective compromises.

[8] SVC was unable to complete training and testing on one dataset.

Figure 7.2 Model Trade-offs: Performance and Prediction Speed (upper right is best)

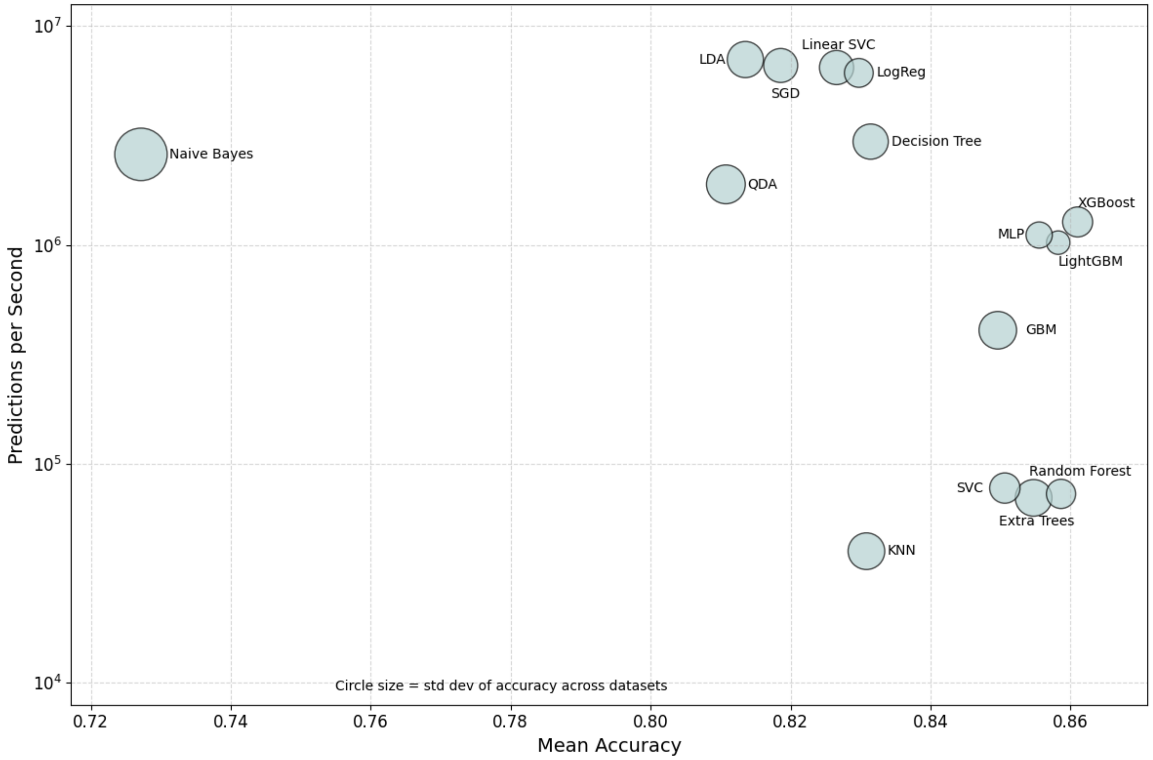
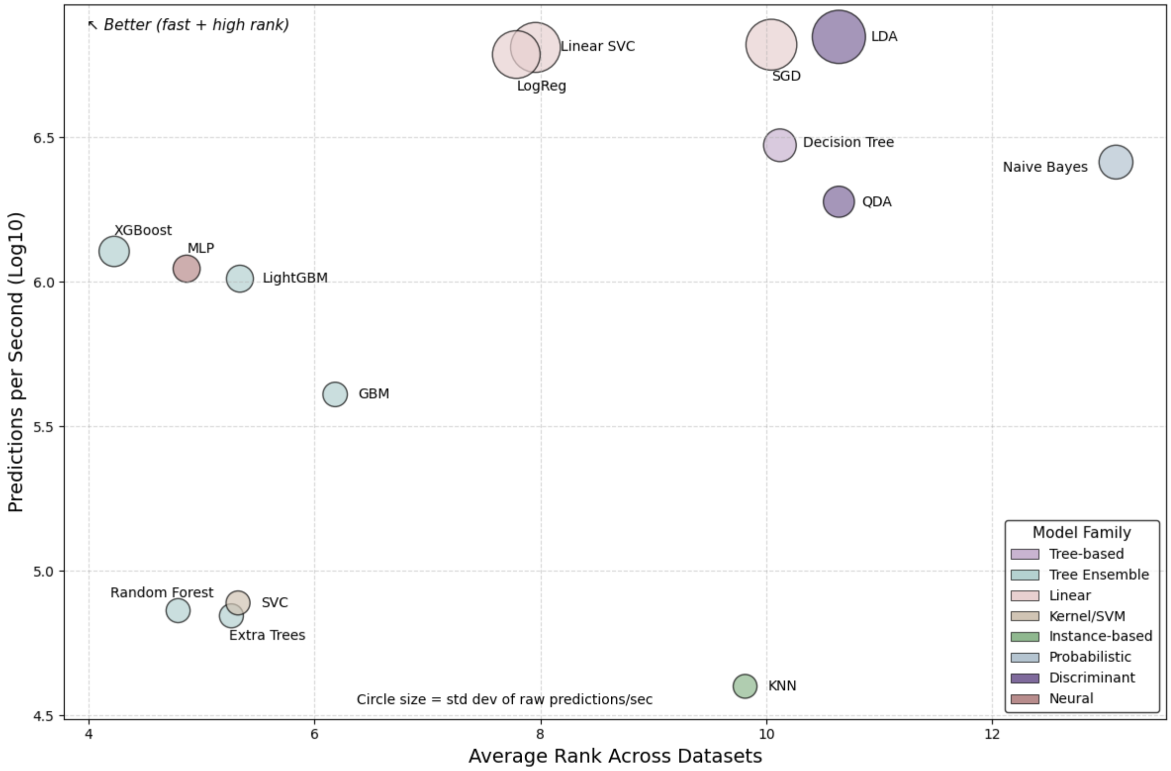


Figure 7.3 Performance Rank vs. Prediction Speed by Model Family (upper left is best)



A few algorithms combine strong accuracy with high throughput. Logistic Regression and Linear SVC stand out as fast, competitive performers, though Linear SVC illustrates a trade-off: it is quicker than Decision Tree but less accurate. KNN is a clear outlier, being both slow and less accurate. Extra Trees and Random Forest present a different trade-off. Despite ranking among the slowest models, they achieve relatively high average accuracy, suggesting that their predictive strength can sometimes justify the computational cost.

The tradeoffs show that throughput cannot be treated as secondary to accuracy. Model differences in predictive strength and computational demands directly affect their suitability for deployment. Latency is critical for online systems, where slow predictions can break real-time requirements, whereas batch systems can tolerate longer runtimes. In production settings, throughput constraints may rule out otherwise strong models. These charts clarify which algorithms are practical when both speed and accuracy matter.

7.3 Integrated Performance Score

This subsection offers a different look at the benchmarking results by combining accuracy and throughput into a single, easily interpretable measure. Rather than evaluating accuracy and throughput separately, this integrated score offers a high-level view that simplifies communication and supports decision-making. The composite score weights accuracy at 60% and throughput at 40%, producing a weighted score for each model.

Figure 7.4 shows each model's underlying normalized values for both metrics.[9] **Figure 7.5** presents the combined weighted score of both measures and indicates model family. This approach merges both metrics into a single interpretable "deployment" score, offering a means to explain and compare models.

The deployment score reshapes the landscape of model performance. Models with high accuracy but low throughput, such as ensemble trees, shift downward when efficiency is factored in. Conversely, linear models gain ground as their speed offsets moderate accuracy. Neural and LDA models occupy middle positions, reflecting balanced but not dominant profiles. The composite view highlights that no single family dominates across both dimensions, and the relative ranking depends on how accuracy and throughput are weighted. Figure 7.5 illustrates these shifts, showing how the balance of strengths and weaknesses changes when the two measures are combined.

This integrated view supports more informed model selection. The 60/40 weighting is illustrative, not prescriptive. If speed is critical, adjusting the weights will elevate fast but less accurate models. If accuracy is paramount, the top tier models remain dominant. Presenting both normalized metrics and the weighted score side by side demonstrates how priorities shape model choices. The optimal balance depends on the use case: whether the goal is maximum accuracy, speed, or a compromise.

[9] Scores are normalized using min-max scaling to ensure comparability across models. Each value is rescaled to a 0–1 range based on the lowest and highest observed mean accuracy or throughput.

Figure 7.4 Model Comparison: Normalized Accuracy vs. Throughput

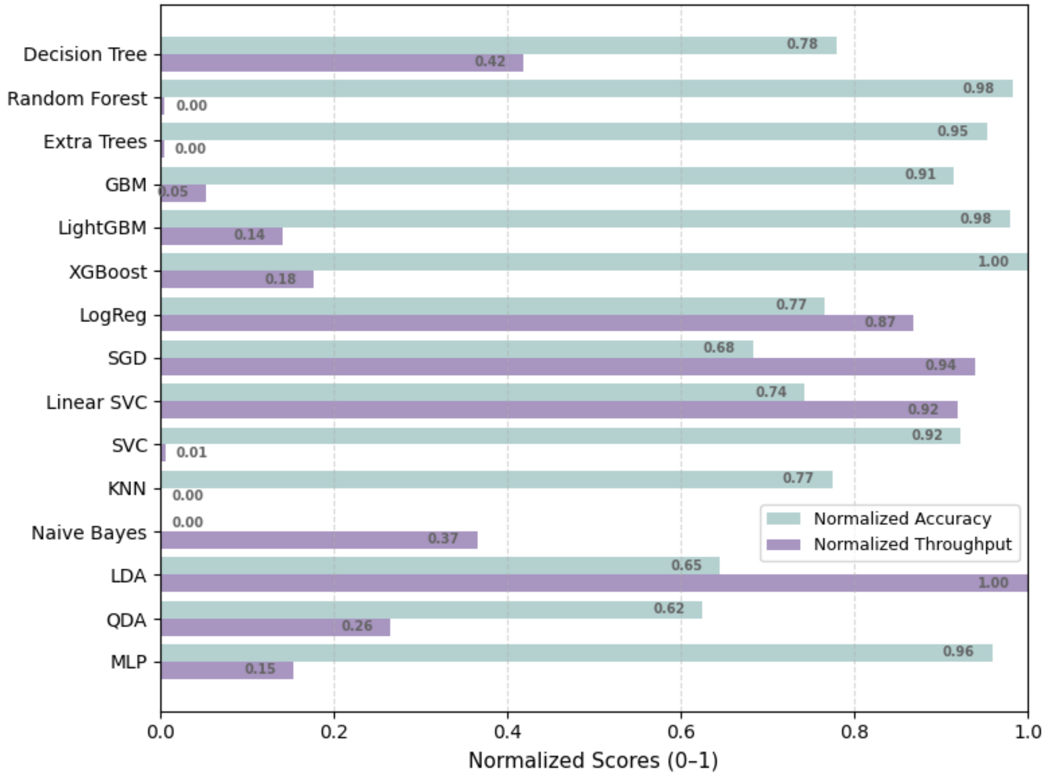
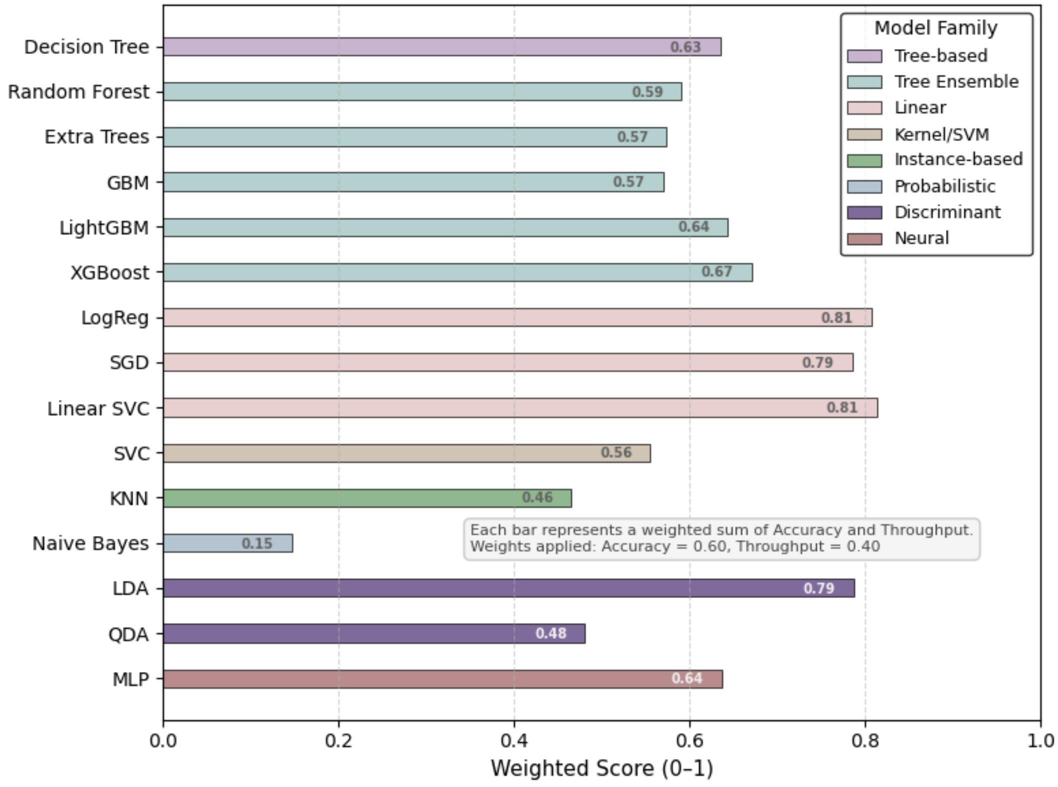


Figure 7.5 Model Deployment Score Based on Combined Accuracy and Speed



8. Conclusion

Across 159 datasets, seven of the fifteen models emerged as the most reliable choices: XGBoost, LightGBM, Random Forest, Extra Trees, Gradient Boosting, SVC, and the Multi-Layer Perceptron. These models are statistically indistinguishable in performance, meaning any of them can be chosen based on secondary factors like ease of use or interpretability. They consistently delivered top-tier accuracy, making them safe choices when predictive performance is the priority.

However, speed is often critical. Linear models such as Logistic Regression, SGD, and Linear SVC were noticeably faster, at least twice as fast as the next fastest group, and more than 75× faster than the slowest four models. However, these models sacrificed several points of accuracy. On the other end, powerful ensemble models like Random Forest, Extra Trees, and SVC deliver top accuracy but can be significantly slower. The sweet spot for balancing speed and accuracy was gradient boosting models (XGBoost and LightGBM), and MLP, which offer high accuracy at a more practical throughput.

By contrast, powerful ensemble models such as Random Forest, Extra Trees, and SVC delivered top accuracy but were significantly slower. In short, the study confirms that boosting methods and tree ensembles dominate when accuracy matters most, while linear methods are more relevant when speed is critical. Ultimately, the "best" model isn't universal; it depends on whether an application prioritizes maximum accuracy, sub-second response times, or a robust balance of both.

The analysis also makes clear what drives model success or failure: dataset complexity. Measures of neighborhood overlap and non-linear boundaries were the strongest predictors of classification difficulty. When classes are tightly intermixed or decision boundaries are curved, accuracy drops sharply across most models. For example, datasets with poorly defined boundaries or overlapping regions consistently posed the greatest hurdles.

Tree ensembles and boosting methods handled these complexity challenges best, while linear and probabilistic models struggled. For example, datasets with high neighborhood complexity consistently reduced accuracy for LDA and Naive Bayes, but XGBoost and Random Forest maintained strong performance. This shows that choosing the right model depends not only on speed and accuracy tradeoffs, but also on the structure of the dataset itself.

Interestingly, factors commonly assumed to be major issues, such as class imbalance or dataset size alone, had a surprisingly minimal or even misleading positive impact on model performance in this context. Top-performing models proved more adept at navigating these complex data structures, while simpler probabilistic and discriminant algorithms consistently struggled when faced with such challenges.

Class imbalance appeared to correlate positively with model performance, but deeper analysis showed this was a spurious effect. Highly imbalanced datasets in the collection also tended to have lower geometric and topological complexity, making these datasets easier to classify. The evidence confirms that geometric and neighborhood complexity, not class distribution, drives classification difficulty, while modern ensemble methods effectively managed imbalance.

9. Limitations and Future Work

Scope of Task: This study focused on binary classification with tabular datasets. Multi-class classification and regression tasks were not covered. *Future Work: Extend benchmarking to multi-class tasks and regression problems.*

Model Coverage: Focused on 15 widely used models suitable for tabular data. Modern deep learning architectures designed for images (CNNs) and sequences (transformers, LSTMs) were not included, as they are not optimized for tabular data. *Future Work: Incorporate emerging tabular-specific neural architectures (e.g., TabNet, FT Transformer, SAINT, Wide & Deep, and TabTransformer) and domain-specific models to compare against traditional approaches.*

Computational Metrics: Captured prediction throughput and RAM usage but did not measure tuning time, training time, or GPU utilization. *Future Work: Benchmark hyperparameter tuning times, training efficiency, memory scaling, and GPU acceleration to provide a fuller picture of computational demands.*

Dataset Collection: Based benchmarks on 159 tabular binary datasets from public repositories (OpenML, UCI, Kaggle). Larger collections, private/proprietary datasets, and domain-specific data were not represented, nor were higher complexity datasets. *Future Work: Expand to larger and more diverse datasets, including domain-specific collections (healthcare, finance, commerce, education), to strengthen generalizability and validate findings across application areas.*

Hyperparameter Optimization: Used grid search with predefined parameter ranges for model tuning. More advanced methods (e.g., Bayesian optimization, AutoML) were not explored, and the search was limited in both the number of parameters considered and the range of values tested. *Future Work: Investigate whether advanced hyperparameter tuning changes model rankings or reduces performance gaps between top-tier models.*

Complexity Frameworks: Focused was on understanding what makes datasets difficult to classify and identifying which models handle specific types of complexity most effectively. Dataset difficulty was characterized using 22 measures from Lorena et al. While other complexity frameworks exist, they were not compared here. *Future Work: Evaluate additional, well-established complexity frameworks to determine which best predict model effectiveness across diverse problem types.*

Class Imbalance: Observed a spurious relationship between the C1 measure and model performance. Class imbalance effects were not isolated from other dataset characteristics. *Future Work: Conduct controlled experiments to isolate class imbalance, systematically vary imbalance ratios, and assess their direct impact on model performance and robustness.*

Principal Component Analysis (PCA): Did not apply PCA, Singular Value Decomposition (SVD), nor any dimensionality reduction techniques in the benchmarking workflow. The potential effects of reduced feature spaces on model performance and computational efficiency were not examined. *Future Work: Evaluate impact of reducing dimensions using PCA on training, model performance, and throughput.*

Domains of Competence Analysis: Summarized model performance within six complexity categories. It did not examine classifier performance across the full multidimensional complexity space to identify domains of competence, as demonstrated in Mansilla & Ho's 9D framework. *Future Work: extend this to multidimensional complexity space analysis, as demonstrated by Mansilla & Ho (2005),* [10] *to identify precise domains of competence where specific geometric combinations favor particular algorithms.*

Appendix A - Dataset Complexity Measures

This appendix provides definitions for the 22 dataset complexity measures used to characterize the 159 datasets in this study. The measures are grouped into the six categories defined in Lorena et al. (2019) [1]. An overview of each category is provided below:

- **Feature-based measures (F1, F1v, F2, F3, F4)**: Evaluate the ability of individual features, or combinations of features, to separate classes. Examples include the maximum Fisher's discriminant ratio and the volume of the overlapping region between classes.
- **Linearity measures (L1, L2, L3)**: Assess how well classes can be separated using linear boundaries, often by utilizing a Linear Support Vector Machine (SVM) and assessing its error rate or the distance of misclassified points from the decision boundary.
- **Neighborhood measures (N1, N2, N3, N4, T1, LSC)**: Focus on local distribution of instances, analyzing class overlap, borderline examples, and decision boundary complexity based on nearest neighbors.
- **Network measures (Density, ClsCoef, Hubs)**: Convert the dataset into a graph structure to evaluate properties like dataset density and connectivity, providing insights into topological complexity.
- **Dimensionality measures (T2, T3, T4)**: Capture sparsity by relating the number of samples to number of features (dimensions), as high dimensionality can increase problem difficulty.
- **Class imbalance measures (C1, C2)**: Quantify imbalance in class sample size, a factor that can significantly affect classification difficulty.

The 22 measures defined in Lorena et al. [1] include the 12 originally developed and described by Ho and Basu [2]. Together, they provide evidence of whether dataset classes are well separated or heavily interleaved, an important determinant of classification accuracy. [2] To standardize interpretation, modifications were introduced [1] into the earlier definitions [2], so that all measures assume values within bounded intervals, with higher values consistently indicating greater complexity and lower values indicating lesser complexity.

For this project, values of all 22 measures were calculated for each of the 159 datasets. Calculating the measures excluded all non-numeric features. Non-numeric features were excluded and remaining numeric features were standardized using scikit-learn’s StandardScaler to ensure comparability across measures.

Because of compute limits,[10] 46 datasets (of 159) with more than 15,000 records were stratified sampled to 15,000 instances while preserving class balance. After preprocessing, the *problexity* Python library was applied to each dataset's feature matrix and target label, generating a score for each measure.

The *problexity* module automatically inverts scores for measures where lower values originally indicated higher complexity. In general, values close to 0 represent lower complexity, while values close to 1 represent higher complexity. This normalization ensures consistent directional logic across all measures. The module also provides an overall dataset difficulty score, calculated as the arithmetic mean of the 22 individual measures.

The complexity measures are defined in six categories by [1], and subsequently rephrased by [12] and [13]. The tables that follow define each measure and interpret the complexity level it represents. For selected measures where definitions may not be immediately intuitive, formulas are included to provide clarity and replicability.

Category 1. Feature-Based Measures

This group evaluates how effectively individual features, or combinations of features, can distinguish between classes in the dataset. These measures capture the separability of the data based on its raw attributes, without relying on model-specific behavior. A high number of discriminative features tends to simplify the dataset. **Figure C-1**, on the following page, provides a description of the five measures in this category.

[10] For datasets exceeding 15,000 records, a stratified random sample was drawn to prevent memory overflow during complexity calculations. This limitation reflects current inefficiencies in the problexity implementation, which does not efficiently leverage available multicore resources (28 cores in our environment).

Figure C-1. Feature-Based Measures

Acronym	Name and Definition	Interpretation	Min	Max
F1	**Maximum Fisher's Discriminant Ratio.** Calculates how close the classes are, for each feature. Quantifies the maximum linear separability achieved between any pair of classes when projected onto the single most informative feature (or feature combination). Roughly, it indicates how easy it is to draw a straight line that separates the classes using the best single feature available. Calculates for each feature the ratio of two components: a) Squared difference in the means of two classes, and b) The sum of the variances of the data in each class This maximizes the distance between classes while minimizing the spread within each class. F1 is the maximum of these feature ratios (how separable the dataset is along its single best feature axis). Lorena (and Python's *problexity*) invert this score.	**Higher Complexity.** No single feature separates (discriminates) classes well. Class distributions overlap heavily in every feature space.	≈0	1
F1v	**Directional Vector Maximum Fisher's Discriminant Ratio.** It complements F1, looking for the hyperplane, generated by a vector, that best separates the classes instead of using the separate features. Also similar to F1, score is inverted: F1v = 1 / (1 + Directional Fisher's Discriminant Ratio).	**Higher Complexity.** Dataset is so heavily mixed that a linear boundary cannot achieve good separation, even in the most advantageous direction.	≈0	1
F2	**Volume of Overlapping Region.** Estimates the degree of overlap between the class distributions of each feature (dimension). This is done by calculating the normalized overlap for each feature individually (based on a feature's min and max values within an individual class) and then taking the product of the normalized values across all features. This final product represents the overall volume of the feature space where classes heavily intermingle, directly measuring the ambiguity of the raw features. Note from [2]: The value is 0 if there is at least one feature in which value ranges of two classes do not overlap.	**Higher Complexity.** Greater overlap suggests less clear class boundaries, indicating that the classes (e.g., 0 and 1) are thoroughly intermingled.	0	1
F3	**Maximum Individual Feature Efficiency.** Estimates the individual efficiency of each feature in separating the classes. Specifically, it quantifies the proportion of samples that fall into the overlapping (ambiguous) region for each feature, and determines the feature with the	**Higher Complexity.** Nearly all, or all, feature values of separate classes overlap; even the best single feature struggles to separate the classes.	0	1

Acronym	Name and Definition	Interpretation	Min	Max
	maximum value among all the features. Here Lorena et al. [1] (and *problexity*) takes the complement of this measure (i.e., the minimum value) so that higher values are obtained for more complex problems. The problem can be considered simpler if there is at least one feature which shows low ambiguity between the classes.			
F4	**Collective Feature Efficiency.** Based on an iterative use of F3 over the dataset, each time choosing the next most efficient feature and setting aside the non-overlapped points of that feature, until there are no more points or features. The final F4 score is determined by the proportion of data points that were never unambiguously discriminated throughout this entire sequential process.	**Higher Complexity.** A large proportion of data points (e.g., dataset records) remain ambiguous and hard to classify; suggests that the features do not work effectively together to clearly separate the classes	0	1

Category 2. Linearity Measures

These measures try to quantify to what extent the classes are linearly separable, that is, if it is possible to separate the classes by a hyperplane. [1] The assumption is that linearly separable datasets are less complex. Higher values indicate greater complexity, suggesting that linear decision boundaries may be insufficient and that non-linear models could be required for effective classification. **Figure C-2** provides a description of the three measures included in this category.

Figure C-2. Linearity Measures

Acronym	Definition	Interpretation	Min	Max
L1	**Sum of the Error Distance by Linear Programming.** Assesses if the data are linearly separable by computing the sum of the distances of incorrectly classified examples to a linear boundary used in their classification. If the value of L1 is zero, then the problem is perfectly linearly separable and can be considered simpler than a problem for which a non-linear boundary is required. [1]	**Higher Complexity.** Many data points are far from being correctly separated by a linear boundary. This suggests the classes are significantly non-linearly separable, signifying a high complexity problem in terms of linearity.	0	≈1
L2	**Error Rate of the Linear Classifier.** The actual error rate (1 - accuracy) of a simple linear classifier (problexity uses a Linear SVM) trained on the full dataset. L2 has similar issues with L1 in that it does not differentiate between problems that are barely linearly separable (i.e., with a narrow margin) from those with classes that are very far apart. [1]	**Higher Complexity.** A high error rate suggests the presence of non-linear patterns or substantial overlap (i.e., dataset cannot be effectively separated by a linear model).	0	1

Acronym	Definition	Interpretation	Min	Max
L3	**Non-Linearity of a Linear Classifier.** Creates a new dataset, randomly interpolating pairs of training examples of the same class. Then, a linear classifier is trained on the original data and has its error rate measured in the new data points. This index is sensitive to how the data from a class are distributed in the border regions and also on how much the convex hulls which delimit the classes overlap. [1]	**Higher Complexity.** Boundaries are non-linear or highly ambiguous, confusing the linear classifier. Classes are internally "messy" or intertwined, making them highly non-linear.	0	1

Category 3. Neighborhood Measures

These measures try to capture the shape of the decision boundary and characterize the class overlap by analyzing local neighborhoods of the data points. Some of them also capture the internal structure of the classes. All of them workover a distance matrix storing the distances between all pairs of points in the dataset. [1]

These measures evaluate the complexity of class boundaries by analyzing the proximity and distribution of neighboring data points. These measures determine distances between neighborhoods, overlaps, and decision boundaries. They are particularly relevant for assessing the behavior of instance-based classifiers such as k-nearest neighbors (k-NN), which rely on local structure to make predictions. **Figure C-3** provides a description of the six measures included in this category.

Figure C-3. Neighborhood Measures

Acronym	Definition	Interpretation	Min	Max
N1	**Fraction of Borderline Points.** The fraction of data points over all data points whose nearest neighbor belongs to a *different* class in a Minimum Spanning Tree (MST). The MST is examined to identify edges that connect two examples belonging to opposite classes ($y_i \neq y_j$). N1is the fraction of data points (vertices) that connect to at least one of these opposite-class edges. These points are considered borderline because they are the closest neighbors that belong to different classes, indicating a tight boundary, an overlap, or noisy data.	**Higher Complexity.** Indicates high class overlap and diffuse, interwoven boundaries. The decision boundary would need to be highly complex or convoluted to separate these classes effectively. Could indicate a case where erroneous class labels are introduced during data preparation. [1]	0	1
N2	**Ratio of Intra/Extra Class Nearest Neighbor (NN) Distance.** The ratio of the average distance between a point and its nearest neighbor of the *same* class (intra) to the average distance to its nearest neighbor of a *different* class (extra). Measures if points from the same class are closer to each other (or not) than they are to points from a different class.	**Higher Complexity.** Indicates classes are locally mixed and heavily overlap, leading to high ambiguity and noisy boundaries	0	≈1

Acronym	Definition	Interpretation	Min	Max
N3	**Error Rate of NN Classifier.** The error rate (1 - accuracy) of a 1-NN classifier, often calculated using leave-one-out cross-validation method.* Note from [2]: The proximity of points in opposite classes obviously affects the error rate of a NN classifier. * Each data point is classified using all the other points as the training set, with its own record left out, so the error rate reflects how well the classifier generalizes at the most granular level.	**Higher Complexity.** Indicates that the class boundaries are highly non-linear and difficult for a simple distance-based approach, resulting in a higher error rate.	0	1
N4	**Non-linearity of NN Classifier.** Measures the error rate of the 1-NN classifier on a set of synthetic points interpolated between same-class pairs. Synthetic points are calculated at the midpoint between every pair of points that belong to the same class. These points should ideally also be classified the same as the two end points. Similar to L3 but using 1-NN instead of a linear model.	**Higher Complexity.** Indicates that the local class boundaries are excessively jagged or ambiguous. Many samples (records) intrude into the “wrong” class’s territory.	0	1
T1	**Fraction of Hyperspheres Covering Data.** Measures how fragmented and tangled the class regions are. It estimates the difficulty of separating the classes by calculating how many distinct, single-class "islands" must be individually identified to cover all the data points. It uses a geometric process: a) Hypersphere growth - expanding the radius of each sphere around every data point until it reaches a datapoint of another class, b) Sphere elimination - eliminating small spheres contained in larger spheres, and c) T1 calculation: ratio of the number of remaining, non-overlapping spheres to total number of data points.	**Higher Complexity.** Many small spheres remain. Data points of one class are deeply surrounded by the other class. The decision boundary needed to separate these small, intermixed regions is highly complex, nonlinear, and difficult for algorithms to model accurately.	0	1

Acronym	Definition	Interpretation	Min	Max
LSC	**Local Set Average Cardinality.** Measures the average local density of same-class neighbors around each data point. It measures how isolated an instance is from its own class relative to the proximity of the opposite class. The measure uses a distance-based mechanism for every data point (x). the average of the set of points whose distance is smaller than the distance of the other class. (*n* = number of records or instances) Lorena et al. inverts the LSC result, so the case of low actual LSC (data points are scattered close to the boundary) gets a high score. $LSC = \frac{\sum \lvert L_{x_i} \rvert}{n^2}$ Where $\lvert L_{xi} \rvert$ is the cardinality of local set for instance x_i, and *n* is the total number of instances in the dataset.	**Higher Complexity.** Indicates local neighborhoods are highly mixed and overlapping, with many points from other classes. Data points are extremely close to the decision boundary.	0	$1 - \frac{1}{n}$

Category 4. Network Measures

This category uses graph theory to characterize the topological structure of the dataset by modeling data points as nodes and local neighborhood relationships as connections. These measures quantify properties like local connectivity, cliquishness, and sparsity. Analyzing this graph structure reveals how fragmented or cohesive the underlying class boundaries are, offering insight into how tightly grouped or dispersed the data are.

Crucially, Lorena et al. [1] invert the scores for all three measures from their standard definitions. This normalization ensures that a higher score consistently indicates higher complexity, specifically due to a lack of local structure, greater dispersion, or increased fragmentation within the dataset's neighborhoods. **Figure C-4**, on the following page, provides a description of the three measures included in this category.

Figure C-4. Network Measures

Acronym	Definition	Interpretation	Min	Max
Density	**Average Density of the Network.** Quantifies how fully connected the dataset's neighborhood graph is. It is derived by comparing the actual number of existing links (edges) between data points (nodes) within a defined local neighborhood to the maximum possible number of links that could exist. It measures how tightly clustered and connected points of the same class are in the local neighborhood. The Lorena et al. Density measure uses an inverted and normalized calculation to ensure that lower density in the graph structure correctly maps to higher complexity.	**Higher Complexity.** Fewer edges observed for datasets of low density (examples are far apart in the input space) and/or for which examples of opposite classes are near each other, imply higher complexity.	0	1
ClsCoef	**Average Clustering Coefficient.** Quantifies the cliquishness (or tendency to form closed triangles) within the local neighborhoods of the data graph. It assesses the degree of local connectivity, which is then inverted to quantify complexity. It is derived as the mean of the ratio between the actual number of closed triangles (cliques) centered on a point and the maximum possible number of triangles that could exist with its neighbors. This calculation assesses the local cohesiveness or clumpiness of the dataset's graph structure.	**Higher Complexity.** Data points in a neighborhood are not tightly connected to one another. There are few "closed triangles" or cliques, indicating higher complexity.	0	1
Hubs	**Mean Hub Score.** This is a measure of the influence of each node of the graph. [1] It assesses the concentration of influence within the dataset's graph structure by quantifying how often a few points ("hubs") are chosen as nearest neighbors by others. A Hub is a data point that is a nearest neighbor to a significantly larger number of other points than the average. This measure captures the structural homogeneity of the dataset. With the Lorena et al. inversion, a low score for this concentration suggests a stable, centralized structure (low complexity), while a high score suggests a dispersed, fragile structure (high complexity).	**Higher Score Complexity.** The structure is less centralized, more dispersed, and offers less robust local guidance to model.	0	1

Category 5. Dimensionality Measures

This group captures the potential for data sparsity arising from high dimensionality relative to the number of samples. Higher values indicate greater complexity, suggesting that models may encounter challenges in localized or underpopulated regions of the feature space. **Figure C-5**, on the following page, provides a description of the three measures included in this category.

Figure C-5. Dimensionality Measures

Acronym	Definition	Interpretation	Min	Max
T2	**Average Number of Features per Point.** Defined as the ratio of the number of features to the number of data points in the dataset. Reflects data sparsity: if there are many predictive attributes and few data points, they will probably be sparsely distributed in the input space. (*m* = number of features)	**Higher Complexity.** More features per point will increase sparsity, indicating higher complexity.	≈0	*m*
T3	**Average Number of PCA Dimensions per Point.** Measures the dataset's intrinsic dimensionality relative to its sample size. It is calculated as the ratio of the number of essential Principal Component Analysis (PCA)[a] dimensions (PCA-D) needed to explain 95% of the data's variance to the total number of records in the dataset. This measure represents the true, necessary dimensionality of the data after removing redundancy. (*m* = number of features)	**Higher Complexity.** Even when reduced by PCA, the ratio of required dimensions to records remains high, indicating sparsity.	≈0	*m*
T4	**Ratio of the PCA Dimension to the Original Dimension.** The ratio of the intrinsic dimensionality (from PCA) to the original number of features. This quantifies how many of the original features are genuinely needed to explain the data's variance (the intrinsic dimensionality) versus the total number of features available.	**Higher Complexity.** If nearly all the original features are needed to describe the data's variability, complexity is greater. These relationships between the input variables are complex, redundant, or highly scattered.	0	1

[a] PCA reduces dimensionality by projecting data onto orthogonal directions that capture maximal variance in a dataset, and offers some insight into a dataset's intrinsic complexity

Category 6. Class Imbalance Measures

These measures quantify disparities in the number of samples in each class, which can lead to model bias toward the majority class. The two measures help characterize the extent to which class imbalance may affect classification performance. According to Lorena et al., when the differences are severe, most of the ML classification techniques tend to favor the majority class and present generalization problems. [1]

Lorena et al. [1] utilize generalized and normalized formulas for C1 and C2, rather than their raw statistical definitions. This methodology ensures both measures are correctly oriented, meaning that a higher score consistently indicates a higher level of imbalance complexity for the algorithm. Thus, both measures provide a standardized, comparable assessment of the penalty imposed on classifiers by skewed class distributions. **Figure C-6**, on the following page, provides a description of the two measures included in this category.

Figure C-6. Class Imbalance Measures

Measure	Definition	Interpretation	Min	Max
C1	**Entropy of Class Proportions.** Derived from Shannon Entropy, then normalized and inverted to quantify the Imbalance difficulty of the class distribution. When classes are perfectly balanced (i.e., 50% v. 50%), C1 score is lowest. When one class dominates completely (e.g., 99% v. 1%), C1 score is highest, meaning there is no uncertainty about the class label. A model can simply guess the majority class.	**Higher Complexity.** The less balanced the classes, the higher the complexity.	0	1
C2	**Imbalance Ratio.** Measures the degree of disparity in the number of samples per class, using a generalized formula designed to work consistently for both binary and multiclass classification problems. The value ranges from 0 for perfectly balanced datasets to nearly 1 for extreme imbalance, imposing a significant difficulty penalty. The formula simplifies for a binary classification task to: $C2 = 1 - \frac{2n_1n_2}{n_1^2n_2^2}$ Here, n1 and n2 represent the number of records belonging to the two respective classes.	**Higher Complexity.** Indicates a highly imbalanced dataset.	0	1

Appendix B – Dataset Characteristics and Sources

Exhibit C-1, on the following page, catalogs the 159 datasets used in the benchmarking study. For each entry, the exhibit lists the name, record count, feature count, and source. Also shown are three analytical indicators: the overall complexity score, mean accuracy (across all 15 models), and the dominant complexity category for each dataset. These indicators show how a dataset's structural properties contribute to classification difficulty, referencing the complexity analysis presented in the main report.

The Dominant Category column identifies which of the six complexity categories (as defined by Lorena et al. [1]) best characterizes each dataset. This category is determined by calculating the mean standardized score for each group of complexity measures and then selecting the group with the maximum mean, which is reported alongside the category name (in parentheses). For example, a dataset is classified as "Feature-based" if the average of its five feature-based measures (F1, F1v, F2, F3, F4) was the highest category mean. That mean value is shown in parentheses.

Selected statistics from the dataset catalog include:

- Most records (566,602): covertype
- Least records (32): parity5
- Most features (2,569): ember2024_50000
- Least features (3): prnn_synth, make_blobs_dataset, make_circles_dataset, and make_moons_dataset
- Highest complexity score (0.683): parity5
- Lowest complexity score (0.181): kddcup99
- Highest mean accuracy (1.000): TCGA_GBM_LGG_Mutations_all
- Lowest mean accuracy (0.180): parity5
- Highest maximum category score (1.000): parity 5, Feature-based category
- Lowest maximum category score (0.386): ember2024_50000, Neighborhood category.
- Number of occurrences in Dominant Category:

Feature-based	50
Linearity	0
Neighborhood	2
Network	90
Dimensionality	0
Class Imbalance	17
Total	**159**

Exhibit C-1 Dataset Catalog and Characteristics

Dataset	Records	Features	Source	Complexity Score	Mean Accuracy	Dominant Category (Mean Lorena Score)
adult_income	45,222	15	UCI	0.408	0.838	Feature-based (0.649)
ailerons	13,750	41	OpenML	0.320	0.852	Feature-based (0.509)
air_quality_and_pollution_assessment	5,000	10	Kaggle	0.295	0.965	Class Imbalance (0.656)
albert	58,252	32	OpenML	0.518	0.638	Network (0.862)
Amazon_access	32,769	10	OpenML	0.499	0.941	Feature-based (0.823)
analcatdata_aids	50	5	OpenML	0.412	0.511	Feature-based (0.578)
analcatdata_asbestos	83	4	OpenML	0.487	0.749	Feature-based (0.851)
analcatdata_bankruptcy	50	7	OpenML	0.286	0.841	Network (0.684)
analcatdata_boxing1	120	4	OpenML	0.484	0.733	Feature-based (0.808)
analcatdata_boxing2	132	4	OpenML	0.501	0.651	Feature-based (0.898)
analcatdata_creditscore	100	7	OpenML	0.305	0.923	Network (0.549)
analcatdata_cyyoung8092	97	11	OpenML	0.340	0.786	Network (0.685)
analcatdata_cyyoung9302	92	11	OpenML	0.339	0.831	Network (0.623)
analcatdata_fraud	42	12	OpenML	0.466	0.694	Network (0.770)
analcatdata_japansolvent	52	10	OpenML	0.267	0.809	Network (0.533)
analcatdata_lawsuit	264	5	OpenML	0.285	0.974	Class Imbalance (0.737)
appendicitis	106	8	OpenML	0.326	0.870	Network (0.571)
australian	690	15	OpenML	0.403	0.860	Network (0.836)
backache	180	32	OpenML	0.449	0.794	Network (0.778)
Bank Customer Churn Prediction	10,000	12	Kaggle	0.504	0.840	Network (0.823)
bank_marketing_data	45,211	13	UCI	0.422	0.896	Feature-based (0.628)
banknote_authentication	1,372	5	UCI	0.242	0.982	Network (0.646)
BEED	8,000	17	UCI	0.317	0.858	Feature-based (0.602)
biomed	209	9	OpenML	0.325	0.902	Network (0.808)
bioresponse	3,434	420	OpenML	0.339	0.749	Feature-based (0.539)
blood_transfusion_classification	748	5	UCI	0.446	0.775	Feature-based (0.731)
BNG_breast-w	39,366	10	OpenML	0.295	0.980	Network (0.559)
BNG_cmc	55,296	10	OpenML	0.489	0.780	Feature-based (0.832)
breast_cancer_dataset	569	31	Kaggle	0.209	0.962	Network (0.526)
breast_cancer_prediction	569	31	UCI	0.208	0.962	Network (0.526)
california_environment_conditions	128,009	18	OpenML	0.372	0.938	Network (0.714)
california_housing_classification	20,640	9	OpenML	0.358	0.856	Feature-based (0.608)
cardiovascular_disease_classification	70,000	13	OpenML	0.514	0.701	Feature-based (0.876)
churn	5,000	21	OpenML	0.478	0.910	Network (0.850)
clean1	476	169	OpenML	0.258	0.976	Network (0.857)
cleve	303	14	OpenML	0.437	0.797	Network (0.847)

Dataset	Records	Features	Source	Complex-ity Score	Mean Accuracy	Dominant Category (Mean Lorena Score)
cmc	1,473	10	OpenML	0.502	0.688	Feature-based (0.843)
codrna	488,565	9	OpenML	0.347	0.948	Feature-based (0.638)
company_bankruptcy_prediction	6,819	96	Kaggle	0.360	0.945	Class Imbalance (0.864)
compass	16,644	18	OpenML	0.456	0.719	Network (0.744)
comprehensive _diabetes_clinical	100,000	16	Kaggle	0.297	0.951	Network (0.692)
contraceptive_method_choice	1,473	10	UCI	0.501	0.688	Feature-based (0.843)
corral	160	7	OpenML	0.403	0.952	Feature-based (0.747)
covertype	566,602	11	OpenML	0.476	0.784	Network (0.772)
credit	16,714	11	OpenML	0.421	0.721	Feature-based (0.722)
credit_approval	690	16	UCI	0.393	0.857	Feature-based (0.613)
credit_g	1,000	21	OpenML	0.524	0.745	Feature-based (0.859)
creditcard	284,807	30	OpenML	0.275	0.996	Class Imbalance (0.768)
default_of_credit_card_clients	30,000	24	UCI	0.444	0.805	Feature-based (0.675)
diabetes_health_indicators	253,680	22	UCI	0.493	0.854	Feature-based (0.819)
diabetes130US	71,090	8	OpenML	0.505	0.596	Feature-based (0.770)
diabetic_retinopathy_debrecen	1,151	20	UCI	0.419	0.687	Network (0.660)
digits	1,797	65	Other	0.320	0.983	Network (0.900)
e_commerce_shipping_data	10,999	12	Kaggle	0.437	0.670	Network (0.804)
egg_eye_state	14,980	15	OpenML	0.374	0.753	Feature-based (0.731)
electricity	38,474	8	OpenML	0.415	0.795	Network (0.692)
elevators	16,599	19	OpenML	0.382	0.865	Network (0.661)
ember2024_50000	50,000	2,569	Other	0.267	0.900	Neighborhood (0.386)
fico_heloc	9,871	24	OpenML	0.449	0.724	Network (0.763)
fitness_class_2212	1,467	7	Kaggle	0.408	0.737	Feature-based (0.612)
gallstone	319	39	UCI	0.412	0.753	Network (0.855)
gina_agnostic	3,468	971	OpenML	0.403	0.887	Network (0.983)
haberman	306	4	OpenML	0.506	0.728	Feature-based (0.834)
heart_disease_dataset_comprehensive	1,190	12	OpenML	0.379	0.881	Network (0.724)
heart-statlog	270	14	OpenML	0.417	0.819	Network (0.817)
heloc	10,000	23	OpenML	0.447	0.709	Network (0.698)
higgs_sample20000	19,999	29	OpenML	0.518	0.664	Network (0.895)
hill_valley	1,212	101	OpenML	0.367	0.608	Neighborhood (0.576)
house_16h	13,488	17	OpenML	0.384	0.846	Feature-based (0.620)
HTRU2	17,898	9	UCI	0.265	0.976	Class Imbalance (0.452)
in_vehicle_coupon_recommendation	12,684	26	UCI	0.569	0.583	Feature-based (0.983)
insurance	23,548	11	OpenML	0.579	0.746	Feature-based (0.982)
Invistico_Airline	129,880	22	Kaggle	0.408	0.896	Network (0.830)
ionosphere	351	35	OpenML	0.311	0.915	Network (0.693)

Dataset	Records	Features	Source	Complex-ity Score	Mean Accuracy	Dominant Category (Mean Lorena Score)
iranian_churn	3,150	14	UCI	0.329	0.918	Network (0.653)
iris	150	5	OpenML	0.282	0.893	Network (0.578)
irish	500	6	OpenML	0.472	0.997	Feature-based (0.785)
is_this_a_good_customer	1,723	14	OpenML	0.496	0.834	Feature-based (0.755)
jannis	57,580	55	OpenML	0.475	0.749	Network (0.927)
jasmine	2,984	145	OpenML	0.425	0.777	Network (0.878)
jm1	10,885	22	OpenML	0.383	0.807	Feature-based (0.695)
kc1	2,109	22	OpenML	0.348	0.851	Feature-based (0.603)
KDD	5,032	46	OpenML	0.432	0.754	Network (0.692)
kdd_ipums_la_97_small	5,188	21	OpenML	0.289	0.849	Network (0.610)
kddcup99	25,000	42	OpenML	0.181	0.998	Class Imbalance (0.930)
liver-disorders	345	7	OpenML	0.468	0.677	Network (0.705)
lupus	87	4	OpenML	0.394	0.736	Network (0.614)
madeline	3,140	260	OpenML	0.511	0.670	Network (0.995)
MagicTelescope	19,020	12	OpenML	0.207	0.983	Network (0.678)
make_blobs_dataset	20,000	3	Other	0.355	0.845	Feature-based (0.588)
make_circles_dataset	20,000	3	Other	0.413	0.864	Feature-based (0.763)
make_classification	20,000	31	Other	0.466	0.897	Network (0.951)
make_moons_dataset	20,000	3	Other	0.289	0.930	Network (0.618)
mammography	11,183	7	OpenML	0.361	0.982	Class Imbalance (0.897)
marketing_campaign	2,240	29	Kaggle	0.438	0.877	Network (0.758)
medical_appointment_no_shows_convert_dates	110,527	20	OpenML	0.478	0.782	Network (0.773)
Medical-Appointment	61,214	19	OpenML	0.557	0.789	Feature-based (0.882)
miniboone	72,998	51	OpenML	0.293	0.873	Feature-based (0.582)
multiple_sclerosis_disease_cols_removed	273	17	Kaggle	0.436	0.794	Network (0.883)
mux6	128	7	OpenML	0.484	0.828	Network (0.978)
nasa_nearest_object_valued	338,199	6	Kaggle	0.319	0.880	Network (0.507)
naticusdroid_android_permissions	29,332	87	UCI	0.273	0.928	Network (0.461)
nhanes_age_predictions_subset	2,278	8	UCI	0.430	0.837	Feature-based (0.681)
numerai28.6	96,320	22	OpenML	0.587	0.518	Feature-based (0.997)
occupancy_detection_combined_date	20,560	10	UCI	0.246	0.991	Network (0.751)
online_shoppers	12,330	18	OpenML	0.382	0.876	Feature-based (0.582)
ozone-level-8hr	2,534	73	OpenML	0.380	0.921	Network (0.764)
page-blocks	5,473	11	OpenML	0.305	0.958	Class Imbalance (0.650)
parity5	32	6	OpenML	0.683	0.180	Feature-based (1.000)
pc1	1,109	22	OpenML	0.314	0.927	Class Imbalance (0.744)
pc3	1,563	38	OpenML	0.335	0.865	Class Imbalance (0.650)

Dataset	Records	Features	Source	Complexity Score	Mean Accuracy	Dominant Category (Mean Lorena Score)
pc4	1,458	38	OpenML	0.331	0.905	Class Imbalance (0.596)
philippine	5,832	309	OpenML	0.435	0.729	Network (0.849)
PhiUSIIL Phishing URL - Website_sample_25000	25,000	52	UCI	0.219	0.999	Network (0.714)
phoneme	5,404	6	OpenML	0.419	0.841	Network (0.730)
pima_indians_diabetes	768	9	UCI	0.452	0.757	Network (0.732)
pol	10,082	27	OpenML	0.308	0.915	Network (0.610)
pollen	3,848	6	OpenML	0.549	0.496	Feature-based (0.902)
postoperative-patient-data	88	9	OpenML	0.547	0.699	Network (0.816)
predict_students_dropout_and_academic_success	4,424	37	UCI	0.403	0.830	Network (0.811)
prnn_crabs	200	8	OpenML	0.287	0.947	Network (0.579)
prnn_synth	250	3	OpenML	0.321	0.850	Network (0.613)
pumpkin_seeds	2,500	13	OpenML	0.301	0.880	Network (0.615)
qsar	1,055	41	OpenML	0.353	0.840	Network (0.672)
qsar-biodeg	1,054	42	OpenML	0.352	0.847	Network (0.672)
Raisin	900	8	UCI	0.289	0.859	Network (0.503)
rice_cammeo_and_osmancik	3,810	8	UCI	0.255	0.925	Network (0.581)
ringnorm	7,400	21	OpenML	0.457	0.893	Network (0.864)
rl	4,970	13	OpenML	0.498	0.670	Feature-based (0.798)
road_safety	111,762	33	OpenML	0.465	0.729	Network (0.823)
sa-heart	462	10	OpenML	0.480	0.703	Network (0.795)
sandtander_customer_satisfaction	25,000	201	OpenML	0.509	0.904	Network (0.994)
satellite	5,100	37	OpenML	0.287	0.989	Class Imbalance (0.930)
segment	2,310	20	OpenML	0.258	0.984	Network (0.656)
sepsis_survival_minimal_clinical_records	110,204	4	UCI	0.505	0.926	Feature-based (0.878)
shipping	10,999	10	OpenML	0.430	0.660	Network (0.746)
shrutime	10,000	11	OpenML	0.494	0.840	Network (0.783)
sick	3,103	23	OpenML	0.355	0.892	Class Imbalance (0.745)
smoking_drinking_dataset_Ver01	100,000	24	Kaggle	0.436	0.722	Feature-based (0.633)
sonar	208	61	OpenML	0.357	0.811	Network (0.857)
spambase	4,601	58	OpenML	0.301	0.918	Feature-based (0.486)
spect	267	23	OpenML	0.433	0.795	Network (0.803)
spectf	349	45	OpenML	0.367	0.857	Network (0.844)
stroke_prediction_dataset	4,909	11	Kaggle	0.396	0.915	Class Imbalance (0.829)
student_depression_dataset	27,901	17	OpenML	0.424	0.813	Network (0.739)
svmguide3	1,243	23	OpenML	0.406	0.828	Feature-based (0.593)
sylvine	5,124	21	OpenML	0.396	0.918	Network (0.922)
taiwanese_bankruptcy_prediction	6,819	96	UCI	0.360	0.945	Class Imbalance (0.864)

Dataset	Records	Features	Source	Complex-ity Score	Mean Accuracy	Dominant Category (Mean Lorena Score)
TCGA_GBM_LGG_Mutations_all	862	26	UCI	0.193	1.000	Network (0.552)
telecom_customer_churn	7,043	20	Kaggle	0.465	0.745	Feature-based (0.832)
twonorm	7,400	21	OpenML	0.397	0.967	Network (0.932)
vehicleNorm	25,000	101	OpenML	0.393	0.847	Network (0.712)
visualizing_soil	8,641	5	OpenML	0.228	0.977	Network (0.649)
vulnonevul	5,692	17	OpenML	0.329	0.983	Class Imbalance (0.953)
waterQuality1	7,996	21	Kaggle	0.502	0.928	Network (0.919)
waveform-5000	5,000	41	OpenML	0.427	0.869	Network (0.912)
wilt_seed	2,000	6	OpenML	0.312	0.965	Class Imbalance (0.792)
wine_quality	6,497	12	UCI	0.482	0.642	Feature-based (0.766)
yeast_ml8	2,417	117	OpenML	0.424	0.973	Network (0.984)

Appendix C - Computational Infrastructure

The performance evaluation workflow was implemented within Visual Studio Code, using Jupyter notebooks on a local Windows PC. Standard data science libraries were employed, including *pandas* for data manipulation and I/O operations (e.g., reading and writing Parquet and Excel files), and *scikit-learn* for model instantiation, training, hyperparameter tuning (via *GridSearchCV*), and cross-validation. Model training and testing followed a 5-fold stratified K-fold procedure to ensure balanced representation of class labels across folds.

Additional libraries used included:

- **joblib** for saving model objects and results.
- **importlib** for dynamically importing and instantiating model classes from metadata, enabling flexible, programmatic creation of model instances without hardcoding imports. Metadata was stored in a Python dictionary containing class names, module paths, default parameters, and hyperparameter grids.
- **IPython.display** for inline notebook outputs.
- **matplotlib**, **seaborn**, and **plotly.express** for charts and results visualizations.

File and directory paths were centrally managed using a combination of a *config.py* module and *pathlib*, improving modularity and extensibility for new datasets. JSON and Excel files store model configurations and benchmarking results for downstream use.

The primary throughput metric was predictions per second. Measurements were taken on a Windows 11 PC equipped with an Intel Core i7-14700F processor (20 cores, 28 logical processors, 2.10 GHz base speed), 32 GB RAM, and a 1 TB SSD. The evaluation pipeline ran on Python 3.13.3 using *scikit-learn* 1.7.0. Models capable of multithreaded execution were configured to utilize all available CPU cores. Throughput measurements were taken under controlled conditions, with background applications closed and power-saving features disabled to minimize interference.

Appendix D - Pearson Correlations Between Complexity Measures and Accuracy by Model Family

Figure D.1 presents Pearson correlation coefficients between each of the 22 complexity measures and model accuracy, calculated separately for each of the eight model families. Each column represents one model family, showing how strongly that family's performance correlates with each complexity dimension.

The "Overall" column provides the correlation across all 2,384 model-dataset pairs for reference, enabling comparison of family-specific sensitivities against the benchmark-wide pattern. The “Range” column shows how much each measure impacts each family differently. Notably, the 14 largest variations in family responses are driven solely by the probabilistic family (Naive Bayes). Excluding this model reduces maximum range from 0.437 to 0.192 (L3: Tree-based – Linear = 0.192), indicating that correlation strengths are relatively consistent across the remaining seven model families.

Figure D. Complexity-Accuracy Correlations by Model Family

Measure	Tree-based	Proba-bilistic	Neural	Linear	Kernel/ SVM	Instance-based	Ensemble / Bagging	Discrim-inant	Overall	Range
N1	(0.837)	**(0.494)**	(0.874)	(0.777)	(0.874)	(0.891)	(0.855)	(0.783)	**(0.777)**	0.397
N3	**(0.865)**	(0.466)	**(0.881)**	(0.734)	**(0.883)**	**(0.897)**	**(0.871)**	(0.758)	(0.771)	0.431
N4	(0.777)	(0.447)	(0.789)	(0.756)	(0.825)	(0.822)	(0.796)	(0.769)	(0.733)	0.378
L2	(0.670)	(0.424)	(0.770)	(**0.860**)	(0.786)	(0.720)	(0.711)	**(0.787)**	(0.709)	**0.437**
L3	(0.643)	(0.425)	(0.750)	(0.835)	(0.768)	(0.694)	(0.687)	(0.761)	(0.687)	0.410
F1v	(0.596)	(0.447)	(0.694)	(0.755)	(0.710)	(0.639)	(0.640)	(0.696)	(0.637)	0.308
T1	(0.646)	(0.336)	(0.658)	(0.580)	(0.646)	(0.694)	(0.605)	(0.563)	(0.567)	0.358
F1	(0.434)	(0.457)	(0.475)	(0.553)	(0.488)	(0.453)	(0.432)	(0.511)	(0.461)	0.121
L1	(0.460)	(0.282)	(0.470)	(0.484)	(0.503)	(0.462)	(0.485)	(0.451)	(0.445)	0.222
N2	(0.559)	(0.264)	(0.501)	(0.369)	(0.507)	(0.541)	(0.476)	(0.407)	(0.427)	0.295
F3	(0.423)	(0.345)	(0.418)	(0.435)	(0.423)	(0.402)	(0.382)	(0.409)	(0.389)	0.090
F4	(0.381)	(0.308)	(0.430)	(0.431)	(0.446)	(0.388)	(0.388)	(0.403)	(0.385)	0.138
F2	(0.288)	(0.175)	(0.319)	(0.351)	(0.336)	(0.332)	(0.347)	(0.312)	(0.311)	0.176
LSC	(0.288)	(0.281)	(0.309)	(0.317)	(0.312)	(0.309)	(0.272)	(0.318)	(0.287)	0.046
Density	(0.254)	(0.056)	(0.254)	(0.291)	(0.240)	(0.273)	(0.230)	(0.211)	(0.224)	0.235
T3	(0.257)	(0.180)	(0.170)	(0.175)	(0.185)	(0.228)	(0.220)	(0.176)	(0.193)	0.087
ClsCoef	(0.247)	(0.030)	(0.190)	(0.231)	(0.195)	(0.249)	(0.188)	(0.156)	(0.181)	0.219
T4	(0.151)	0.026	(0.208)	(0.230)	(0.192)	(0.192)	(0.192)	(0.157)	(0.169)	0.256
Hubs	(0.094)	0.009	(0.158)	(0.189)	(0.109)	(0.135)	(0.078)	(0.079)	(0.101)	**0.199**
T2	(0.142)	(0.122)	(0.059)	(0.058)	(0.054)	(0.130)	(0.094)	(0.060)	(0.083)	0.088
C2	0.362	0.100	0.378	0.430	0.378	0.402	0.345	0.414	0.348	0.330
C1	0.378	0.116	0.386	0.440	0.386	0.409	0.356	0.424	0.358	0.324

References

[1] A. C. Lorena, L. P. Garcia, J. Lehmann, M. C. Souto and T. K. Ho, "How complex is your classification problem? A survey on measuring classification complexity," *ACM Computing Surveys CSUR,* vol. 52, no. 5, pp. 1-34, 2018.

[2] T. K. Ho and M. Basu, "Complexity Measures of Supervised Classification Problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, no. 3, pp. 289-300, 2002.

[3] J. D. Pascual-Triana, D. Charte, M. A. Arroyo and A. Fernandez, "Revisiting Data Complexity Metrics Based on Morphology for Overalap and Imbalance: Snapshot, New Overlap Number of Balls Metrics and Singular Problems Prospect," *Knowl Inf Syst,* pp. 1-29, 2021.

[4] G. Zabergja, A. Kadra, C. M. Frey and J. Grabocka, "Tabular Data: Is Deep Learning All You Need," *TBD,* vol. TBD, p. TBD, TBD.

[5] V. Borisov, T. Leemann, K. SeBler, J. Haug, M. Pawelczyk and G. Kasneci, "Deep Neural Networks and Tabular Data: A Survey," *arXiv preprint arXiv,* vol. 2110.01889, 2021.

[6] S. Uddin and H. Lu, "Confirming the statistically significant superiority of tree-based machine learning algorithms over their counterparts for tabular data," *PLoS ONE,* vol. 19, no. 4 e0301541, 2024.

[7] R. L. Theriault and R. E. Ellisa, "Data Complexity Measures Can Predict Classification Accuracy," pp. 1-34, 2025.

[8] D. C. McElfresh, S. Khandagale, J. Valverde, V. Prasad C., G. Ramakrishnan, M. Goldblum and C. White, "When Do Neural Nets Outperform Boosted Trees on Tabular Data?," *NeurIPS,* pp. 76336-76369, 2023.

[9] H.-J. Ye, S.-Y. Liu, H.-R. Cai, Q.-L. Shou and D.-C. Zhan, "A Closer Look at Deep Learning Methods on Tabular Datasets," 2025.

[10] E. B. Mansilla and T. K. Ho, "On Classifier Domains of Competence," in *International Conference on Pattern Recognition*, 2004.

[11] M. A. Ali, J. Liu, S. Moore and O. Nibouche, "Assessing the Effect of Data Complexity and Instance Overlap Issues on Imbalanced Learning," in *2024 the 7th International Conference on Big Data and Education*, Oxford, 2024.

[12] J. Komorniczak and P. Ksieniewicz, problexity - an open source Python library for binary classification problem complexity assessment, 2022.

[13] J. Eberlein, D. Rodriguez and R. Harrison, "The effect of data complexity on classifier performance," *Empirical Software Engineering,* vol. 30, no. 16, pp. 1-23, 2024.

[14] G. Armano and E. Tamponi, "Experimenting multiresolution analysis for identifying regions of different classification complexity," *Pattern Analysis and Applications,* vol. 19, no. 1, pp. 129-137, 2016.

[15] J. D. Pacual-Triana, D. Charte, M. A. Arroyo, A. Fernandez and F. Herrera, "Revisiting Data Complexity Metrics Based on Morphology for Overlap and Imbalance: Snapshot, New Overap Number of Balls Metrics and Singular Problems Prospect," *TBD,* vol. TBD, no. TBD, p. TBD, 2020.

[16] H. O'Brian Quinn, M. Sedky, J. Francis and M. Streeton, "Literature Review of Explainable Data Analysis," *Electronics,* vol. 13, no. 19, p. 31, 2024.