International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 106-128
https://doi.org/10.70917/ijcisim-2026-5558
Article

Hybrirank: A Dual-Axis Algorithm Ranking Framework via Dataset-Based and Preference-Guided Rank Fusion

Pavan Manikanta Raghava Kasturi[1], Dr. Vithya Ganesan[2], Dr. N. Kirubakaran[3]

[1,2]Department of CSE,Koneru Lakshmaiah Education Foundation ,Andhra Pradesh,India
[3]Department of Computer Science and Business Systems, Chennai Institute of Technology, Chennai,
[1]kasturikpmr@gmail.com, [2]vithyamtech@gmail.com, [3]drkiru70@gmail.com

Abstract: Algorithm selection remains a challenging task due to the absence of standardized tools that support informed decision-making for identifying suitable algorithms for AI and ML tasks. It plays a crucial role in designing and deploying intelligent systems. This paper presents Hybrirank, a hybrid framework that ranks algorithms by integrating two components, dataset-based benchmarking module that evaluates performance on varied domain-specific datasets using standard metrics, and a user preference modeling module that captures user-defined priorities through structured input. The system supports clustering, classification, and regression tasks, and allows users to generate rankings based on dataset performance, user preferences, or both. Each mode produces a context-aware ranking tailored to its input. In hybrid mode, HybrirankFramework leverages its core fusion mechanism to integrate data-driven performance and user priorities to generate a more selective and user-aligned recommendation. This ranking serves as a decision-support system, enabling users to select the most appropriate algorithm. Hybrirank offers a dual-layered evaluation process that enhances recommendation relevance, bridges the gap between empirical performance and practical needs, and it supports optimal selection guided by both data and priorities.
Keywords: Decision Support System, Multi-Criteria Decision Making, Hybrid Algorithm Ranking, Context-Aware System, Data-Driven Ranking, Preference-Based Ranking, Interactive System, Performance Benchmarking

1. Introduction:

In the current era of rapidly evolving AI, the development of effective models increasingly depends on selecting algorithms that align with the specific demands of the task and the characteristics of the data. The same algorithm may perform differently across datasets or application domains, making the algorithm selection process highly context-sensitive. In practice, algorithm choice is often guided by prior familiarity, experience, instinct or intuition, or trial-and-error, often overlooking key data properties or the specific needs of the task instead of being based on systematic evaluation. This can lead to suboptimal outcomes when the selection fails to consider the nature of the dataset or task-specific requirements.

To address these limitations, this study presents a hybrid framework that integrates empirical evaluation with user or task-specific intent. By aligning algorithm performance with task-specific preferences, the system generates context-aware rankings, enabling more accurate and practical algorithm selection across AI and ML applications by developers or practitioners working on real-world tasks. Under this unified framework, the system offers three operations namely Classification, Regression, and Clustering.

2. Literature Review:

Arangeofmethodologieshavebeenproposedtotackletheproblemofalgorithmselectionacrossdiversemachinelearningtasks.Thissectionreviews the foundational and recent works that inform each functional component of the Hybrirank Framework. The relevantworksare,

Mosqueira-Rey et al. [1] review human-in-the-loop machine learning systems, emphasizing lightweight interfaces that allow users to select tasks, configure preferences, and provide feedback with minimal complexity. They highlight key principles such as cognitive load reduction, preference-based input, and adaptive UI design. The LTSC interface in Hybrirank is built on these principles to enable simplified, goal-driven interaction, serving as an intuitive medium for generating context-aware algorithm rankings based on user priorities and performance data. Yang et al. [2] proposed a meta-learning method for selecting classification algorithms based on dataset features and parameter tuning, but itfocused only on classification. Hybrirank builds on this by extending to regression and clustering with real-time, task-specific metric evaluation. Roy et al. [3] developed CurFi for selecting regression models using R^2 from curve fitting; while effective for regression alone, it lacks generality. Hybrirank broadens this idea with multi-metric evaluation across all task types.

To expand algorithm diversity, Multi-Layer Perceptron (MLP) is included alongside classical models in both classification and regression. Recent advances like TabPFN v2 by Hollmann et al. [4] show that neural architectures, when effectively (pre)trained, can match or surpass tree-based methods on tabular tasks. Hutter et al. [5] also present MLPs as core to modern AutoML systems, reinforcing their value in model comparison. That's why MLPs are included in Hybrirank, to ensure the ranking framework captures strengths across both traditional and neural approaches, adapting to varied data characteristics.Identifying the optimal number of clusters remains a core challenge in unsupervised learning. Mahmud et al. [6] proposed an ensemble approach that estimates cluster counts by analyzing results across multiple random subsamples, whereas Sowan et al. [7] estimated the optimal k by aggregating internal validation indices within a single KMeans-based setup. OCND in Hybrirank addresses their limitations by integrating five clustering algorithms, diverse internal metrics, and inertia-based elbow detection. It applies consensus voting across models and metrics, supports a user-defined k range, filters unstable outputs, enables optional data scaling, and includes robust error handling with a default fallback. These features ensure a scalable, adaptive, and interpretable solution, offering greater reliability than earlier ensemble methods.

Xu and Tian [8] provided a detailed taxonomy of clustering techniques, and Rodriguez et al. [9] empirically compared their performance across varied data scenarios. While both works showed that clustering depends on data traits, they didn't address how to compute comparative scoring. Hybrirank advances this by computing clustering metrics for algorithm scoring in its benchmarking module. Garouani et al. [10] proposed a meta-feature-based selector for classification and regression using historical data, but it does not compute live metrics or support clustering. Hybrirank complements this by performing real-time benchmarking across classification, regression, and clustering, using dynamic, task-aware metrics within a unified ranking framework.

Tornede et al. [11] propose a meta-learning framework that uses rank aggregation (e.g., Borda count) across algorithm selectors. While effective at the meta level, it lacks per-metric transparency. Hybrirank builds on this by introducing interpretable per-metric ranking tables for granular decision-making and final scoring across ML tasks, a novel approach not addressed in prior work. Corsi and Urbano [12] proposed tie-aware extensions of Rank-Biased Overlap (RBOa/RBOb) to fairly evaluate ranked lists containing ties, whichthough not directly solving algorithm tie-breaking, their work inspired the need to address tie situations more deliberately in algorithm ranking. Building on this idea, Hybrirank introduces Avg-for-Tie: Tie Disambiguator (Avg_{tie}), a task-aware averaging mechanism. It resolves identical scores by computing a refined average of core evaluation metrics tailored to the problem type.Chatterjee et al. [13] fused metric-wise ranks into a single stable explanation to unify multiple XAI evaluation metrics. Bałchanowski and Boryczka [14] showed that aggregating rankings from multiple recommenders improves accuracy via consensus. UMRA builds on these by ranking algorithms per metric with consideration of metric direction, summing total ranks, resolving ties using normalized averages of direct and inverse metric values for fairness, and finally sorting to produce the Final rank table as an interpretable decision-support output.

The IROC method [15]by Hatefi, which derives calibrated weights from user-ranked criteria using refined rank-to-weight conversion, and the framework proposed by Zeeshan et al. [16], which uses weighted user preferences but relies on manual input and lacks backend automation, together provide foundational grounding for Hybrirank's Aspect Option Weighting (AOW).AOW builds on these by replacing manual weight input with expert-defined static scores for selectable options. Users select relevant aspects, and the system applies stable utility values (e.g., "High " = 1.550, "Low" = 1.100) through backend automation for consistent, interpretable preference mapping without user effort. Zhang et al. [17] propose SBCR, which applies deterministic inference-time boosts for calibration, inspiring CABS's use of boosting during result processing. Corsi and Urbano [12] present a rule-based tie-handling approach, offering the idea of differentiating between tied items. However, both lack user-defined aspect coverage, which CABS incorporates through tiered boosts based on the number of matched aspects in tie cases. Ahsen *et al.* [18] proposed SUMMA, an unsupervised method that weights classifier outputs based on confidence to produce ranked predictions. Fujita *et al.* [19] introduced HPA, which aggregates model-generated ranks by constructing a pseudo-answer and selecting top-aligned rankers via similarity. Advancing on these ideas, the CORE in Hybrirank fuses dataset-based and user preference rankings using weighted aggregation, penalties, and tie-breaks to generate context-aware model rankings. The references supporting other functional components in the above literature also collectively contribute to the formulation of CORE, the central engine of Hybrirank.

Chen et al. [20] introduced CMR, a hypernetwork-based re-ranking model enabling real-time adaptation to user preferences without retraining but requiring new model generation, leading to latency. Wang et al. [21] proposed a mutual-information-based re-ranking method using a precomputed utility matrix but lacked real-time responsiveness. Improving on both, Hybrirank's ZAPR enables zero-latency user interaction and instant re-ranking over precomputed results with changed preferences through lightweight computation, without model regeneration.Collectively, the reviewed literature informs the structure, motivation, and functional components that underpin the Hybrirank framework.

3. Methodology :

3.1 System Definition :

The dataset should exist as a single, complete file in either .csv or .xlsx format. For classification tasks, the user must provide an integer target variable representing class labels (not required in preference mode). For regression tasks, the user must provide a numeric target variable (integer or floating-point) for continuous prediction (also not required in preference mode). For preferences-only or hybrid (by both) modes, the user must specify evaluation requirements by selecting task-specific priorities. The responsibility of preprocessing the dataset lies with the user, depending on the intended use.

3.2 System Workflow :

The framework, built in Python and deployed via Google Colab, begins by prompting the user to select one operation type:

Clustering, Classification and Regression followed by one of three recommendation modes:

- Mode 1: Dataset-only

(Chosen when the practitioner prefers algorithm ranking based only on dataset performance metrics.)

- Mode 2: Preferences-only

(Chosen when the practitioner prefers algorithm ranking based only on task-specific preference inputs.)

- Mode 3: Hybrid (Dataset + Preferences)

(Chosen when the practitioner prefers algorithm ranking based on both dataset performance and task-specific preference inputs for more accurate selection.)

Based on the selected mode, the user or practitioner must provide the necessary inputs, such as a dataset file and/or evaluation preferences. The system then processes the input, evaluates multiple algorithms and generates a context-aware algorithm ranking reflecting the specified inputs. The flowchart (Fig. 1)depicts the complete workflow.

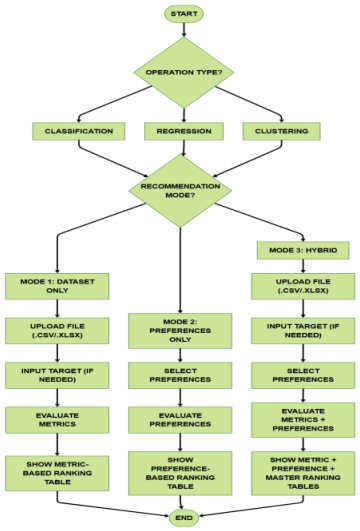

Fig. 1. Overview of Dual-Axis Selection Framework

3.3 Evaluation Models :

For a comprehensive assessment of algorithmic performance across classification, regression and clustering tasks, the Hybrirank framework utilizes a diverse set of established machine learning models, aslisted in (Table 1).

Table 1. Algorithms and Models Considered

Models used for Classification:	**Models used for Regression:**	**Models used for Clustering:**
1. Random Forest 2. XGBoost 3. Gradient Boosting 4. SVM (Support Vector Machine) 5. MLP (Multi-Layer Perceptron) 6. Naive Bayes 7. Logistic Regression 8. KNN (K-Nearest Neighbours) 9. Decision Tree 10. LDA (Linear Discriminant Analysis)	1. XGBoost 2. Random Forest 3. Gradient Boosting 4. Decision Tree 5. MLP (Multi-Layer Perceptron) 6. Linear Regression 7. SVR (Support Vector Regression) 8. KNN (K-Nearest Neighbours) 9. Ridge 10. Lasso 11. Elastic Net	1. KMeans 2. DBSCAN 3. HDBSCAN 4. Gaussian Mixture 5. Agglomerative Clustering 6. Spectral Clustering 7. OPTICS 8. Mean Shift 9. Birch 10. Affinity Propagation

3.4 Evaluation Metrics:

3.4.1 Dataset Mode Metrics :

These metrics(Table 2) assess algorithm performance under different task conditions using datasets.

Table 2. Task-Based Metrics for Dataset Mode

Classification	Regression	Clustering
▪ Accuracy ▪ F1-Score ▪ Precision ▪ Recall ▪ ROC AUC ▪ Average for Tie *(*Avg for Tie*)* ▪ Speed (ms) ▪ Memory Usage (MB)	▪ R^2 (Coefficient of Determination) ▪ MAE (Mean Absolute Error) ▪ RMSE (Root Mean Squared Error) ▪ MAPE (Mean Absolute Percentage Error) ▪ Average for Tie (Avg for Tie) ▪ Speed (ms) ▪ Memory Usage (MB)	▪ Silhouette Score ▪ Davies-Bouldin Index ▪ Calinski-Harabasz Index ▪ Average for Tie (Avg for Tie) ▪ Speed (ms) ▪ Memory Usage (MB)

3.4.2 Preference Mode Metrics:

These questions(Table 3) will be asked across varied task contexts to capture user preferences, , with options listed under each question but omitted here for brevity.

Table 3. Task-Based Metrics in Preferences Mode

Classification	Regression	Clustering
● Sensitive to outliers? ● Sensitive to noise? ● Solves overfitting? ● Speed? ● Data preprocessing required? ● Sensitive to missing values? ● Size of dataset? ● Works with high-dimensional data? ● Interpretability? ● Flexibility (Non-linear)? ● Computational Cost (Training)? ● Ease of Implementation? ● Memory Usage? ● Feature Importance Output? ● Scalability? ● Handling Categorical Features? ● Robustness to Feature Scaling? ● Parameter Tuning Effort? ● Ability to Handle Imbalanced Data? ● Risk of Underfitting? ● Performance on Mixed Data Types? ● Need for Feature Engineering? ● Model Stability?	● Assumes linear relationship? ● Sensitive to outliers? ● Sensitive to noise? ● Solves overfitting? ● Assumes normal distribution? ● Speed? ● Data preprocessing required? ● Sensitive to missing values? ● Handles categorical features? ● Works with high-dimensional data? ● Interpretability? ● Flexibility (Non-linear)? ● Computational Cost (Training)? ● Ease of Implementation? ● Memory Usage? ● Feature Importance Output? ● Scalability? ● Robustness to Feature Scaling? ● Parameter Tuning Effort? ● Ability to Handle Multicollinearity? ● Performance on Mixed Data Types? ● Need for Feature Engineering? ● Model Stability?	● Shape of Clusters Discoverable? ● Number of Clusters Required? ● Handles Varying Cluster Sizes? ● Handles Varying Cluster Densities? ● Sensitivity to Outliers? ● Scalability (Large Datasets)? ● Interpretability of Clusters? ● Handles Overlapping Clusters? ● Deterministic? ● Handles High Dimensionality? ● Discovers Clusters with Bridges? ● Ease of Implementation? ● Parameter Tuning Effort? ● Handles Non-Globular Clusters? ● Robustness to Initialization? ● Provides Cluster Probabilities? ● Suitable for Mixed Data Types? ● Sensitivity to Feature Scaling? ● Computational Cost (Prediction)? ● Memory Usage (Model)? ● Tendency to Find Equal-Sized Clusters? ● Handles Manifold Structures?

		• Computational Cost (Training)?

3.5 Hybrirank: A Dual-Axis Algorithm Ranking Framework :

A Dual-Axis Hybrirank Framework invokes internal components (I) LTSC for guided task and mode selection, (II) MLP extension to include neural models, (III) OCND for estimating the ideal number of clusters, (IV) PMART to display algorithm scores per metric, (V) AFT to resolve ties using averaged metric values, (VI)UMRA to combine metric-based ranks fairly, (VII) AOW for expert-weighted scoring of selected options, (VIII) CABS for boosting tied algorithms based on user aspect coverage, (IX) CORE to integrate benchmarking and preferences into a unified rank, and (X)ZAPR for real-time re-ranking based on updated user inputs. A visual overview of these integrated components is illustrated in (Fig. 2).

Fig. 2. Modules of the Dual-Axis Hybrirank Framework

(I) Lightweight Task Selection & Configuration (LTSC) Interface :

Beyond its core workflow pipeline, Hybrirank introduces the Lightweight Task Selection & Configuration (LTSC) Interface act as a minimal yet powerful interaction layer designed for ease of use. Unlike traditional systems that require extensive setup or GUI overhead, LTSC provides a guided, menu-driven approach for task execution. Users can quickly initiate a comparative analysis by selecting the ML task type (Clustering, Classification, Regression), Choosing their preferred ranking strategy (Dataset-only, Preference-only, or Hybrid), Supplying the dataset and specifying the target variable.

This interface eliminates unnecessary friction in experimentation, especially in fast-paced research or deployment settings. Its design ensures that even users with minimal technical overhead can efficiently navigate and configure ranking scenarios. By embedding core functionality into a single, intuitive interface, LTSC enhances accessibility while enabling rapid prototyping and informed decision-making.The (Fig. 3)presents the LTSC Interface.

```
Enter the type of comparison you want to perform (1=clustering / 2=classification / 3=regression): 1

Select recommendation mode:
1. by dataset only
2. by preferences only
3. by both
Enter choice (1 / 2 / 3): 3
Enter the path to your CSV file: /content/yolo object detection.csv
Performing clustering task...
```

Fig. 3. Selection and Configuration UI of the LTSC Module

(II) Neural Extension: MLP

To extend beyond conventional algorithms, the Multi-Layer Perceptron (MLP) is added as a neural model option in both classification and regression. Its strength in modeling nonlinear relationships complements existing methods like Random Forest, XGBoost, SVM, and Linear Regression, offering a more adaptable selection space in Hybrirank.

(III) Optimal Cluster Number Detection (OCND) :

In clustering workflows, selecting the appropriate number of clusters (k) is a critical yet often non-trivial step. Hybrirank addresses this challenge through its Optimal Cluster Number Detection (OCND) module, a robust, automated system designed to estimate the ideal number of clusters in a dataset using an ensemble-based, metric-driven strategy. The method performs an exhaustive evaluation of k values, ranging from 2 to user-defined maximum , across five diverse clustering algorithms: KMeans, Agglomerative Clustering, Birch, Spectral Clustering, and Gaussian Mixture Models. This multi-model strategy improves reliability by factoring in different algorithmic behaviours and structural assumptions.

For each algorithm, OCND computes a suite of internal validation metrics by Silhouette Score (measuring cohesion and separation),Davies-Bouldin Index (evaluating inter-cluster similarity) and Calinski-Harabasz Index (assessing cluster dispersion).

In addition, for KMeans, OCND calculates inertia values and applies the KneeLocator algorithm to identify the elbow point , a well-established heuristic for selecting optimal k. Each model then identifies its best-performing k based on the peak values of its respective metrics. These independent selections are aggregated through a consensus mechanism, where the most frequently suggested k across all models and metrics is selected as the final optimal cluster count.

To ensure result quality and robustness, OCND incorporates several practical enhancements:

- Unstable or invalid clustering's (e.g., producing a single cluster) are automatically filtered out, preventing their influence on the final decision.
- Optional data scaling is supported to improve clustering behaviour on varied data distributions.
- Resilient error handling ensures that failures in individual models do not interrupt the overall detection process.
- Users may define the search range for k (Predefined = 10), making OCND scalable across small and large datasets.
- In the absence of consensus or when all algorithms fail, OCND falls back to a failsafe default of k = 2.

By combining multiple algorithms, diverse metric signals, and robust logic handling, OCND offers an automated, adaptive, and interpretable mechanism for cluster count estimation. It distinguishes itself from conventional single-metric or model-dependent strategies, making it a cornerstone feature within Hybrirank's clustering pipeline.

(IV) Per-Metric Algorithm Ranking Tables (PMART) :

Hybrirank provides per-metric algorithm rankings in Dataset-only and Hybrid modes, allowing detailed performance analysis across classification, regression, and clustering tasks. These metric-level rankings directly contribute to the final algorithm score, creating a transparent, data-driven ranking pipeline.This granularity supports informed decision-making based on specific evaluation priorities, as shown in Figs. 4(a), 4(b), and 4(c).

Classification Results (Ranked by Precision):

Rank	Algorithm	Precision
1	Gradient Boosting	0.898
2	MLP	0.895
3	XGBoost	0.892
4	Random Forest	0.872
5	Decision Tree	0.868
6	Naive Bayes	0.814
7	KNN	0.808
8	LDA	0.803
9	Logistic Regression	0.76
10	SVM	0.663

Regression Results (Ranked by RMSE):

Rank	Algorithm	RMSE
1	Gradient Boosting	0.287
2	MLP	0.29
3	XGBoost	0.302
4	Random Forest	0.315
5	Decision Tree	0.38
6	Ridge	0.392
7	Linear Regression	0.392
8	KNN	0.416
9	Elastic Net	0.446
10	Lasso	0.446
11	SVR	0.476

Clustering Results (Ranked by Calinski-Harabasz Index):

Rank	Algorithm	Calinski-Harabasz Index
1	KMeans	3524.72
2	Spectral	3454.56
3	Agglomerative	2907.48
4	Birch	2816.28
5	Mean Shift	1149.32
6	Affinity Propagation	268.546
7	HDBSCAN	29.001
8	Gaussian Mixture	14.513
9	OPTICS	12.304
10	DBSCAN	0

Fig. 4(a). Classification Ranks (Precision)

Fig. 4(b). Regression Ranks (RMSE)

Fig. 4(c). Clustering Ranks (CH Index)

(V) Avg-for-Tie: Tie Disambiguator(AFT) :

To enhance ranking precision in scenarios where multiple algorithms yield identical or near-identical composite scores, Hybrirank integrates a task-aware Avg_{tie} differentiation mechanism, as defined in Eqs. (1), (2), and (3).This module computes a refined average score using core evaluation metrics tailored to the task type :

Classification:

$$Avg_{tie} = \frac{Accuracy + F1\ Score + Precision + Recall + ROC\ AUC}{5} \qquad (1)$$

Regression:

$$Avg_{tie} = \frac{R^2 + \frac{1}{MAE} + \frac{1}{RMSE} + \frac{1}{MAPE}}{4} \qquad (2)$$

Clustering:

$$Avg_{tie} = \frac{Silhouette + Calinski\ Harabasz + \frac{1}{Davies\ Bouldin + 1e^{-8}}}{3} \qquad (3)$$

This scoring acts as a fine-grained discriminator to resolve ranking collisions and ensures fair prioritization in the final algorithm list, strengthening Hybrirank's decision-support capabilities.

(VI) Unified Metric-wise Rank Aggregation (UMRA) :

To identify the best-performing algorithm across multiple evaluation criteria, Hybrirank uses a systematic rank aggregation process applied in Mode 1 (Dataset-based) and Mode 3 (Hybrid). This method ensures fairness, interpretability, and robustness in final algorithm ranking.

Step 1: Individual Metric Ranking

Let there be n algorithms and a set of k evaluation metrics $\{M_1, M_2, ..., M_k\}$. For each metric M_j, the algorithms are sorted based on their performance scores as defined in Eq. (4).

- If higher is better (e.g., Accuracy, F1-Score, Precision, Recall, ROC AUC, R^2 , Silhouette Score, Calinski-Harabasz Index):
  Algorithms are ranked in descending order of metric value.
- If lower is better (e.g., MAE, RMSE, MAPE ,Davies-Bouldin Index, Speed (ms), Memory Usage (MB)):
  Algorithms are ranked in ascending order of metric value.

$$R_{ij} = Rank\ of\ algorithm\ A_i\ on\ metric\ M_j \qquad (4)$$

Step 2: Total Rank Aggregation

For each algorithm A_i , the total rank is computed by summing its ranks across all metrics using Eq. (5).

$$TotalRank_i = \sum_{j=1}^{k} R_{ij} \qquad (5)$$

This represents the overall performance considering all metrics equally.

Individual rank tables, such as Fig. 5(a) and Fig. 5(b), collectively contribute to the final consolidated ranking presented in Fig. 5(c).

Clustering Results (Ranked by Silhouette Score):

Rank	Algorithm	Silhouette Score
1	KMeans	0.507
2	Mean Shift	0.506
3	Spectral	0.501
4	Agglomerative	0.462
5	Birch	0.457
6	Affinity Propagation	0.204
7	HDBSCAN	-0.074
8	OPTICS	-0.217
9	Gaussian Mixture	-0.5
10	DBSCAN	-1

Clustering Results (Ranked by Davies-Bouldin Index):

Rank	Algorithm	Davies-Bouldin Index
1	Affinity Propagation	0.256
2	Mean Shift	0.507
3	Spectral	0.524
4	KMeans	0.525
5	Agglomerative	0.53
6	Birch	0.533
7	OPTICS	4.661
8	HDBSCAN	8.715
9	Gaussian Mixture	12.158
10	DBSCAN	inf

Fig. 5(a). Clustering Ranks (Silhouette) **Fig. 5(b).** Clustering Ranks (DB Index)

Algorithm Ranking (Based on Total Ranks):

Rank	Algorithm	Total Rank
1	KMeans	6
2	Spectral	8
3	Mean Shift	9
4	Agglomerative	12
5	Affinity Propagation	13
6	Birch	15
7	HDBSCAN	22
8	OPTICS	24
9	Gaussian Mixture	26
10	DBSCAN	30

Fig. 5 (c). Clustering Ranks (Final Rank's)

Step 3: Tie-Breaking via Normalized Performance Score

If two or more algorithms have the same TotalRank, Hybrirank breaks ties using an average of transformed metric values.

For example, using Eq. (2):

- For gain metrics (e.g., Accuracy, R^2): use the value directly.
- For loss metrics (e.g., MAE, RMSE, MAPE): use the inverse to reward lower values.

$$Avg_{\text{tie}} = (R^2 + 1/MAE + 1/RMSE + 1/MAPE)/4 \qquad (2)$$

This provides a **continuous, performance-sensitive score** for fine-grained comparison in case of tied ranks.

Step 4: Final Sorting

Algorithms are finally sorted by:

- Ascending TotalRank
- If tied, descending Avg_{tie}

(VII) Aspect Option Weighting (AOW) :

To ensure preference-based ranking remains fine-grained and reflective of real-world decision contexts, Hybrirank introduces Aspect Option Weighting (AOW) a unified scoring backbone applied consistently across Clustering, Classification, and Regression tasks to ensure preference-based ranking. Each aspect in the user preference interface (e.g., Scalability, Interpretability, Speed) contains multiple selectable options (e.g., High, Medium, Low). These options are pre-assigned calibrated numerical weights, typically ranging between 1.000 and 1.600. The values are task-specific, representing the practical utility or computational advantage of a behaviour in that operation domain.

For example:

- In classification, options like "Solves overfitting: Yes" may be assigned 1.505 to emphasize robustness.
- In regression, "Assumes linear relationship: No" may be weighted 1.300 to favour flexibility.
- In clustering, "Handles overlapping clusters: Yes" may carry a weight of 1.520 to reflect practical necessity.

By integrating AOW into the preference-based ranking engine, Hybrirank ensures that:

- Identical preference profiles produce distinct scores, avoiding tie collisions.

- Important behavioural traits receive amplified influence in the final ranking.
- Consistency is maintained across all comparison modes, enabling a fair and intelligent merging process in hybrid evaluations.

This mechanism acts as a fine-resolution lens that captures user expectations and translates them into quantifiable guidance, elevating the quality and trustworthiness of the generated rankings.

(VIII) Coverage-Aware Boosting Strategy (CABS) :

To resolve tie situations in preference-based scoring, Hybrirank introduces CABS (Coverage-Aware Boosting Strategy). When multiple algorithms obtain the same total preference score, CABS breaks ties by assigning small deterministic boosts based on how many defined aspects each algorithm satisfies, as specified in Eq. (6).

$$Final\ score = \underbrace{Raw\ score}_{Sum\ of\ matched\ aspect\ scores} + \underbrace{Boost_{coverage-tier}}_{Tie\ control} \quad (6)$$

Boosts are tiered (e.g., 0.1, 0.08, …) and assigned in descending order to algorithms covering more aspects.

Example: Suppose the following algorithms each have the same raw score of 6.75 (Table 4) :

Table 4. CABS: Mathematical Description

Algorithm	Matching Aspects	Raw Score	Boost	Final Score
Algo A	6 aspects	1.550 + 1.600 + 1.400 + 1.350 + 0.500 + 0.350 = 6.75	+0.10	6.85
Algo B	5 aspects	1.550 + 1.505 + 1.495 + 1.200 + 1.000 = 6.75	+0.08	6.83
Algo C	4 aspects	1.550 + 1.500 + 1.500 + 1.200 = 6.75	+0.06	6.81

Here, although all three had the same original score, CABS ranks them fairly by rewarding wider alignment with user preferences. This leads to stable, intuitive rankings even in tie scenarios.

(IX) Combined Objective Ranking Engine (CORE) Hybrirank:

At the foundation of Hybrirank lies a mathematically structured fusion engine designed to integrate two essential aspects of model assessment: objective performance data and subjective user-defined priorities. This integration results in the generation of a Master Ranking Table that supports informed, balanced, and context-aware selection.

Inputs to the Ranking Engine

The ranking engine operates on two distinct rank sources:

- **Dataset-Based Ranks**: These are derived from benchmark evaluation metrics such as RMSE, MAE, and R^2, providing a quantitative view of each model's empirical performance.
- **Preference-Based Ranks**: These reflect user-defined priorities, such as latency, resource usage, or other task-specific considerations, gathered through a structured input interface.

Ranking Computation Methodology

Let R_d denote the dataset-based rank and R_p denote the preference-based rank. The final score, S (Eq. 10) is computed using a three-part weighted formulation in Eqs. (7), (8), and (9) :

1. **Weighted Rank Aggregation**

$$Combined\ Score = w_{ds} \cdot R_d + w_{pf} \cdot R_p \qquad (7)$$

where w_{ds} and w_{pf} are the configurable weights that determine the influence of empirical versus user-defined factors.

2. **Disagreement Penalty**

$$Penalty = \left|R_d - R_p\right| \cdot w_{pen} \qquad (8)$$

A penalty is applied when there is a discrepancy between the two ranks, encouraging alignment and discouraging instability. This penalty is scaled by w_{pen} , a tuneable weight that controls how strongly rank disagreements impact the final score.

3. **Tiebreak Adjustment**

$$Tiebreak = R_d \cdot w_{tb} \qquad (9)$$

A minimal adjustment ensures deterministic ordering in case of identical scores without introducing bias.

4. **Final Score and Ranking**

$$S = Combined\ Score + Penalty + Tiebreak \qquad (10)$$

Entries are sorted in ascending order of S, with lower scores indicating higher overall suitability based on both performance data and user needs.

Predefined Weights in Hybrirank (CORE) :

- w_{ds} = **0.555 :** Weight assigned to the dataset-based rank.
- w_{pf} = **0.445 :** Weight assigned to the preference-based rank.
- w_{pen} = **0.01 :** Penalty coefficient for rank disagreement between the two sources.
- w_{tb}= **0.001 :** Tiebreak adjustment factor applied to the dataset rank to resolve ties.

(X) Zero-latency Algorithm Preference Re-ranker (ZAPR) :

Hybrirank features a real-time, ultra-responsive ranking experience powered by ZAPR (Zero-latency Algorithm Preference Re-ranker). After a one-time initial computation, approximately 5 seconds for preference-only mode (Mode 2) or under 1 minute for hybrid mode (Mode 3), users can freely modify their preferences and receive instantaneous updates to the ranking tables.

This architecture eliminates the need for repeated full evaluations, enabling:

- Instant preference-based re-ranking with no latency or reload.
- Real-time experimentation, where users can test different priority configurations seamlessly.
- Immediate generation of master rankings in hybrid mode or preference rankings in preference mode after editing algorithm preferences.

By decoupling computation from interaction, ZAPR empowers practitioners and researchers to make informed algorithm selection decisions with lightning-fast feedback, supporting practical, iterative exploration without system lag.

4. Results:

Theresultssectionisorganizedbyoperation type: classification, regression and clustering. Within each category, three ranking tables are presented as the results of dataset-based benchmarking, user preference modeling, and the merged fusion output. The dataset-based tablesrankalgorithmsbyperformanceacrossoperation-relevantmetricswithPMART,withtiesresolvedviaAFTandrankingsharmonizedthroughUMRA.Thepreference-basedtablesrankalgorithmsbyuser-selectedprioritiesandcoverage,scoredbyAOWandprioritizedbyCABS,whichalsoresolvesties.Themergedfusiontablesintegratedatasetandpreference rankingsviaCORE,producingconsolidated,context-awarerankings,whileZAPRinstantly re-rankspreferenceandfusiontableswhenuserpreferenceschange.Collectively,Hybrirank provides aplug-and-play,fullyautomated,interpretable,multi-task framework foralgorithmrankingthatinstantlyadaptstonewdataandpreferenceswhileensuringcontext-alignedrankings.

Forclarity,threeexternalvisualizations,oneforeachtable(presented as figures for formatting purposes),arepreparedpercategory.Thesevisualizationsarenotembeddedwithin Hybriranktokeeptheframeworksimpleandtopreservethecompletenumericaldetailsavailableinthetables,whilestillprovidinganaccessibleexternalreferenceforinterpretation.

4.1 Classification :

This section presents the ranking outcomes for classification algorithms using three evaluation modes: Fig. 6(a), Fig. 6(b), and Fig. 6(c). For visual interpretation, three external plots are provided: Fig. 7(a), Fig. 7(b), and Fig. 7(c).

Classification Results (Dataset Based):

Rank	Algorithm	Accuracy	F1-Score	Precision	Recall	ROC AUC	Avg for Tie	Speed (ms)	Memory Usage (MB)
1	Gradient Boosting	0.901	0.899	0.898	0.535	0.964	0.83946	2393.92	50.79
2	MLP	0.899	0.896	0.895	0.537	0.958	0.8371	391.12	193.77
3	XGBoost	0.894	0.892	0.892	0.527	0.956	0.83214	1865.05	130.85
4	Random Forest	0.875	0.873	0.872	0.581	0.936	0.82728	577.31	70.97
5	Decision Tree	0.865	0.866	0.868	0.494	0.699	0.75857	15.69	126.87
6	Naive Bayes	0.766	0.787	0.814	0.582	0.836	0.75704	7.13	108.28
7	LDA	0.832	0.8	0.803	0.508	0.886	0.76586	15.78	170.1
8	KNN	0.844	0.823	0.808	0.346	0.673	0.69865	443.96	184.67
9	Logistic Regression	0.818	0.742	0.76	0.257	0.685	0.65245	18225.8	77.3
10	SVM	0.815	0.731	0.663	0.25	0.585	0.60892	1036.29	101.39

Fig. 6(a). Dataset benchmarking ranking table for classification.

Classification Results (Preferences Based):

Rank	Algorithm	Raw Score	Features Matched	Boost	Final Score + Boost
1	MLP	14.36	11	0	14.36
2	SVM	12.95	10	0	12.95
3	KNN	12.885	10	0	12.885
4	Naive Bayes	11.79	9	0.1	11.89
5	LDA	11.79	9	0.08	11.87
6	Logistic Regression	11.18	9	0	11.18
7	Random Forest	11.145	8	0	11.145
8	XGBoost	10.85	8	0	10.85
9	Decision Tree	9.875	7	0	9.875
10	Gradient Boosting	8.63	6	0	8.63

Fig. 6(b). User preference ranking table for classification**.**

Master Rank Table (Dataset + Preferences) Based :

Master Rank	Algorithm	Rank by Dataset	Rank by Preferences	Final Score
1	MLP	2	1	1.567
2	Gradient Boosting	1	10	5.096
3	Naive Bayes	6	4	5.136
4	XGBoost	3	8	5.278
5	Random Forest	4	7	5.369
6	KNN	8	3	5.833
7	LDA	7	5	6.137
8	SVM	10	2	6.53
9	Decision Tree	5	9	6.825
10	Logistic Regression	9	6	7.704

Fig. 6(c). Fused ranking table for classification.

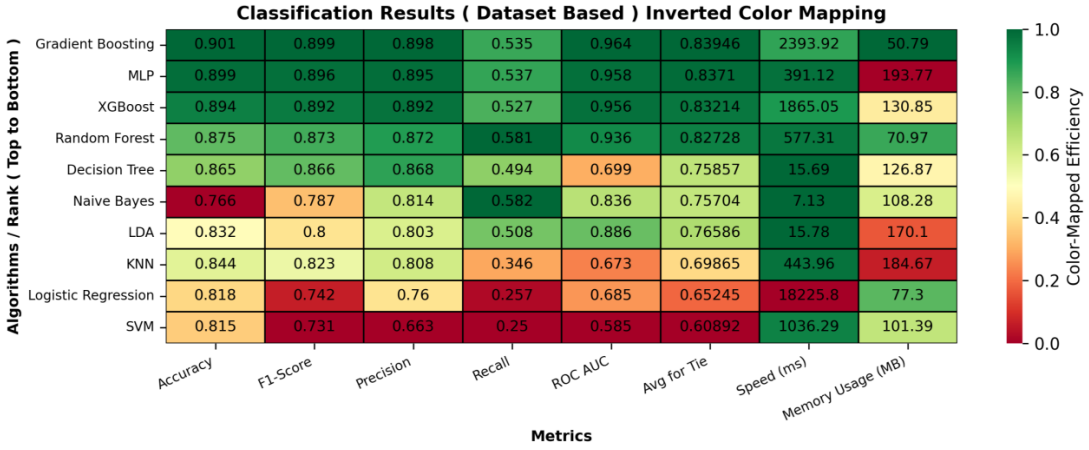

Fig. 7(a). Heatmap of dataset benchmarking for classification.

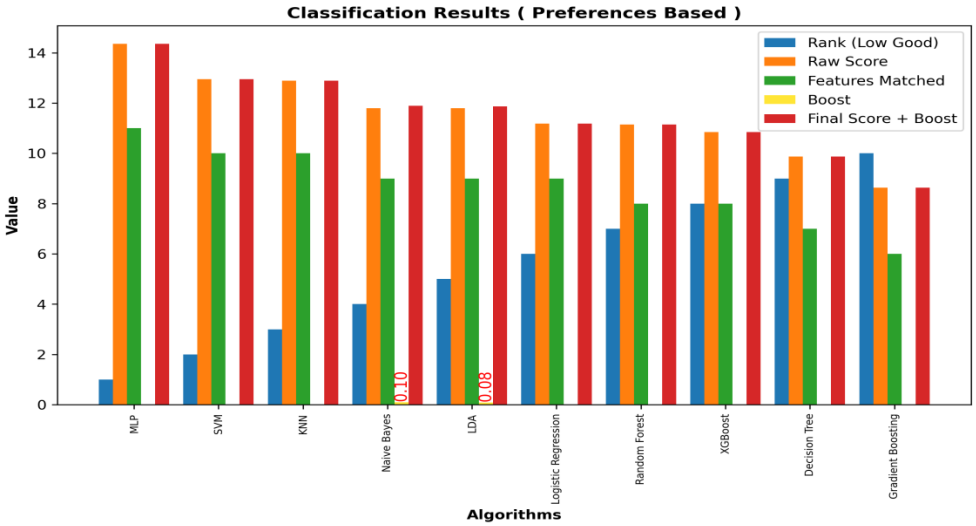

Fig. 7(b). Bar chart of user preference ranking for classification.

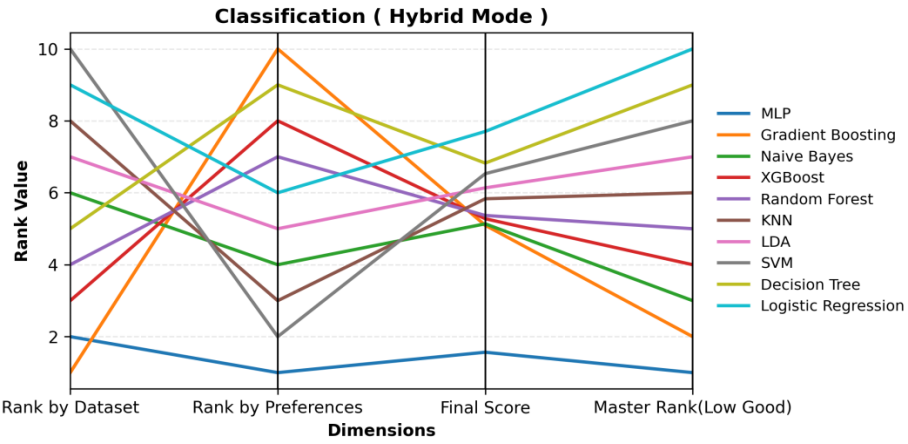

Fig. 7(c). Parallel coordinates of fused ranking for classification.

4.2 Regression :

This section presents the ranking outcomes for regression algorithms using three evaluation modes: Fig. 8(a), Fig. 8(b), and Fig. 8(c). For visual interpretation, three external plots are provided: Fig. 9(a), Fig. 9(b), and Fig. 9(c).

Regression Results (Dataset Based):

Rank	Algorithm	R²	MAE	RMSE	MAPE	Avg for Tie	Speed (ms)	Memory Usage (MB)
1	Random Forest	0.537	0.14	0.315	0	3.95944	682.77	128.86
2	MLP	0.608	0.152	0.29	0.139	4.45043	852.13	118.46
3	XGBoost	0.575	0.151	0.302	0.141	4.39711	108.82	55.16
4	Gradient Boosting	0.617	0.159	0.287	0.196	3.87346	429.83	151.57
5	Decision Tree	0.328	0.141	0.38	0.045	8.12698	14.97	56.47
6	Ridge	0.284	0.26	0.392	0.727	2.01219	5.35	61.29
7	KNN	0.195	0.21	0.416	0.476	2.36702	48.43	114.45
8	Linear Regression	0.284	0.26	0.392	0.727	2.01217	5.76	79.15
9	SVR	-0.052	0.27	0.476	0.183	2.80534	699.25	62.07
10	Elastic Net	0.076	0.306	0.446	1	1.64763	4.5	157.68
11	Lasso	0.076	0.306	0.446	1	1.64756	5.02	182.06

Fig. 8(a). Dataset benchmarking ranking table for regression.

Regression Results (Preferences Based):

Rank	Algorithm	Raw Score	Features Matched	Boost	Final Score + (boost)
1	Gradient Boosting	20.6	14	0	20.6
2	XGBoost	19.35	13	0	19.35
3	Decision Tree	17.6	12	0.1	17.7
4	Random Forest	17.6	12	0.08	17.68
5	Ridge	13	9	0.1	13.1
6	Lasso	13	9	0.08	13.08
7	KNN	12.75	9	0	12.75
8	Linear Regression	12.65	9	0	12.65
9	SVR	11.35	8	0	11.35
10	Elastic Net	11.3	8	0	11.3
11	MLP	9.8	7	0	9.8

Fig. 8(b). User preference ranking table for regression.

Master Rank Table (Dataset + Preferences) Based :

Master Rank	Algorithm	Rank by Dataset	Rank by Preferences	Final Score
1	Random Forest	1	4	2.366
2	XGBoost	3	2	2.568
3	Gradient Boosting	4	1	2.699
4	Decision Tree	5	3	4.135
5	Ridge	6	5	5.571
6	MLP	2	11	6.097
7	KNN	7	7	7.007
8	Linear Regression	8	8	8.008
9	Lasso	11	6	8.836
10	SVR	9	9	9.009
11	Elastic Net	10	10	10.01

Fig. 8(c). Fused ranking table for regression.

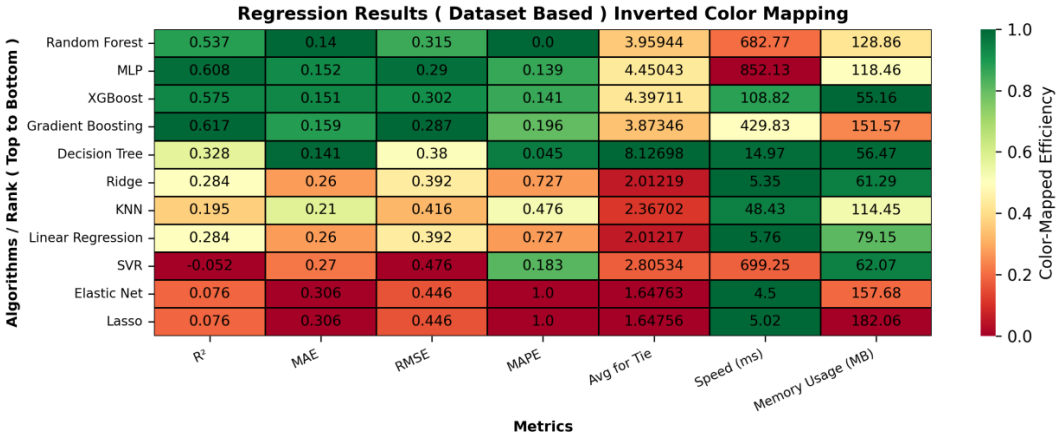

Fig. 9(a). Heatmap of dataset benchmarking for regression.

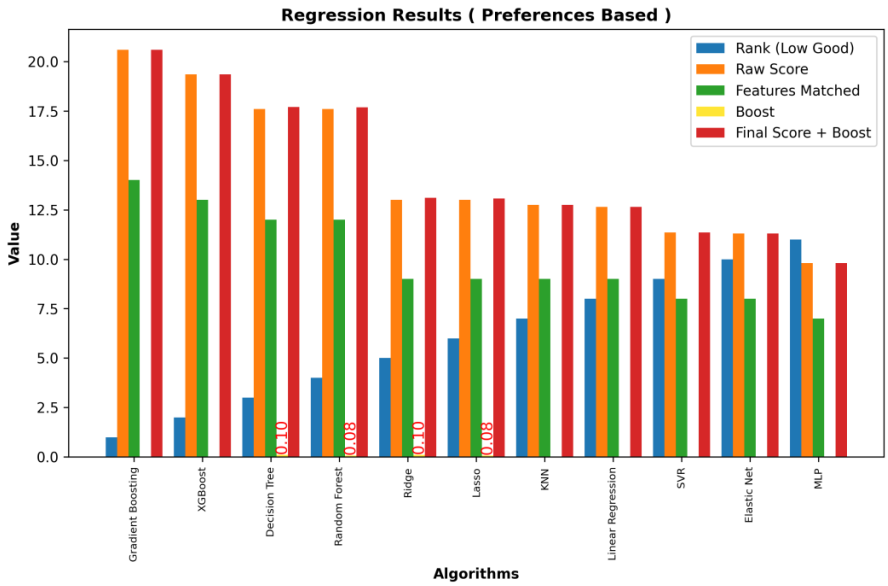

Fig. 9(b). Bar chart of user preference ranking for regression.

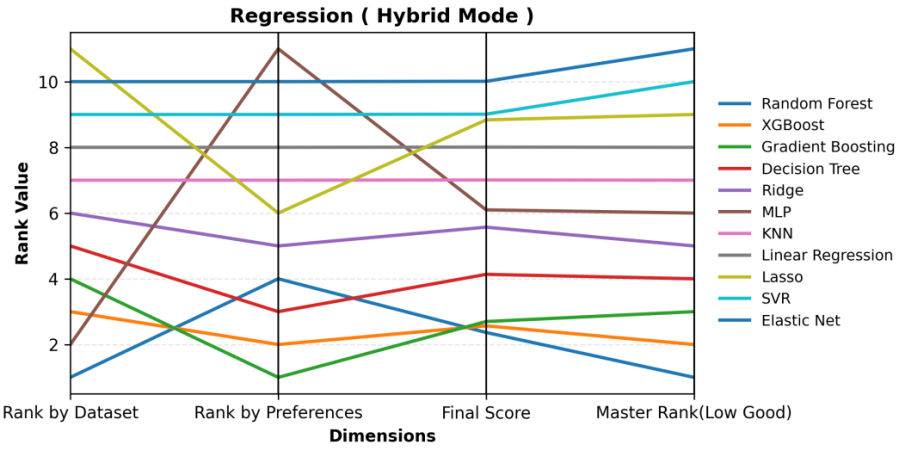

Fig. 9(c). Parallel coordinates of fused ranking for regression.

4.3 Clustering :

This section presents the ranking outcomes for clustering algorithms using three evaluation modes: Fig. 10(a), Fig. 10(b), and Fig. 10(c). For visual interpretation, three external plots are provided: Fig. 11(a), Fig. 11(b), and Fig. 11(c).

Clustering Results (Dataset Based):

Rank	Algorithm	Silhouette Score	Davies-Bouldin Index	Calinski-Harabasz Index	Avg for Tie	Speed (ms)	Memory Usage (MB)
1	KMeans	0.507	0.525	3524.72	1175.71	15.73	116.28
2	Spectral	0.501	0.524	3454.56	1152.32	124.82	72.22
3	Mean Shift	0.506	0.507	1149.32	383.932	1100.68	67.26
4	Agglomerative	0.462	0.53	2907.48	969.942	16.86	109.96
5	Affinity Propagation	0.204	0.256	268.546	90.8874	383.32	173.29
6	Birch	0.457	0.533	2816.28	939.538	79.37	179.09
7	HDBSCAN	-0.074	8.715	29.001	9.68056	17.36	101.25
8	OPTICS	-0.217	4.661	12.304	4.10056	606.66	65.89
9	Gaussian Mixture	-0.5	12.158	14.513	4.69826	408.82	66.39
10	DBSCAN	-1	inf	0	-0.33333	13.3	157.18

Fig. 10(a). Dataset benchmarking ranking table for clustering.

Clustering Results (Preferences Based):

Rank	Algorithm	Raw Score	Features Matched	Boost	Final Score + Boost
1	Agglomerative	17.265	13	0	17.265
2	KMeans	12.575	11	0	12.575
3	OPTICS	11.88	9	0	11.88
4	Spectral	11.36	9	0	11.36
5	DBSCAN	10.88	8	0	10.88
6	Mean Shift	9.565	7	0.1	9.665
7	HDBSCAN	9.565	7	0.08	9.645
8	Birch	7.515	6	0	7.515
9	Affinity Propagation	6.845	5	0	6.845
10	Gaussian Mixture	4.03	3	0	4.03

Fig. 10(b). User preference ranking table for clustering.

Master Rank Table (Dataset + Preferences) Based :

Master Rank	Algorithm	Rank by Dataset	Rank by Preferences	Final Score
1	KMeans	1	2	1.456
2	Agglomerative	4	1	2.699
3	Spectral	2	4	2.912
4	Mean Shift	3	6	4.368
5	OPTICS	8	3	5.833
6	Affinity Propagation	5	9	6.825
7	Birch	6	8	6.916
8	HDBSCAN	7	7	7.007
9	DBSCAN	10	5	7.835
10	Gaussian Mixture	9	10	9.464

Fig. 10(c). Fused ranking table for clustering.

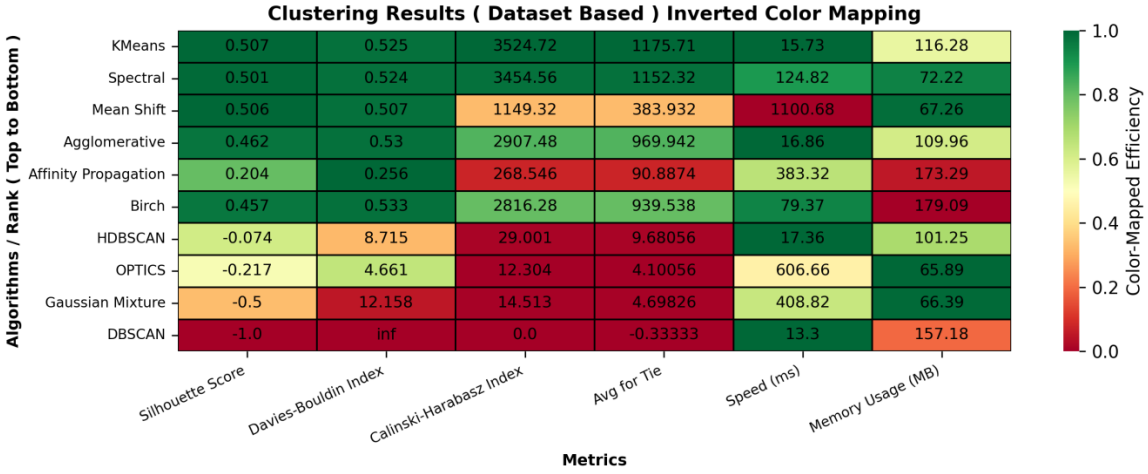

Fig. 11(a). Heatmap of dataset benchmarking for clustering.

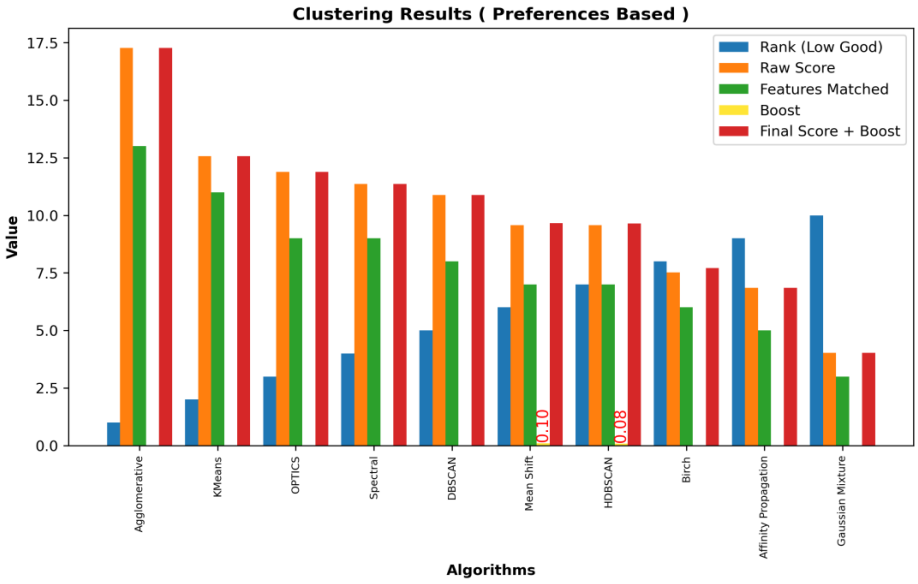

Fig. 11(b). Bar chart of user preference ranking for clustering.

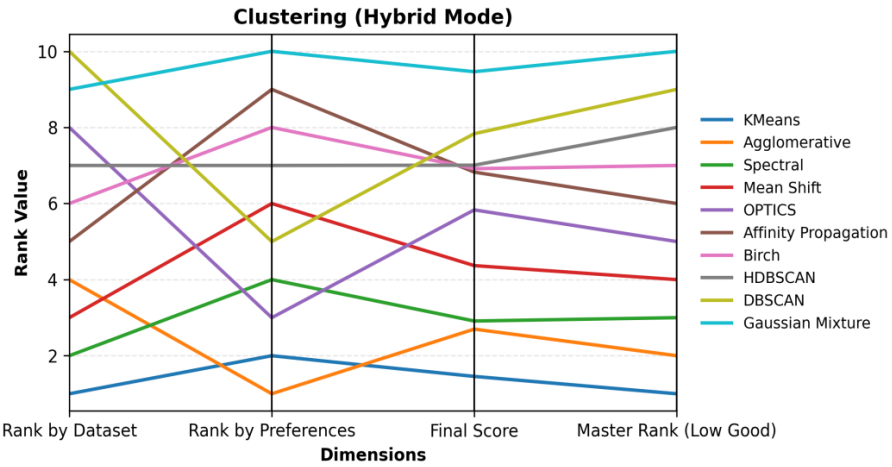

Fig. 11(c). Parallel coordinates of fused ranking for clustering.

5.Comparison : (Table 5) provides a feature-based comparison outlining how the proposed Hybrirank framework advances beyond existing reference approaches.

Table 5. Comparison of Existing Methods and Hybrirank

Feature	What Reference Provides	What Hybrirank Does or Improves
Lightweight Task Selection & Configuration (LTSC)	Mosqueira-Rey et al. [1] describe human-in-the-loop ML systems using adaptive, lightweight UIs for task selection, preference setup, and feedback—focusing on low cognitive effort.	Hybrirank implements this via its LTSC terminal interface, which guides users through mode selection, operation type, target variable input, and integrates both preference-based and performance-based systems to generate context-aware algorithm rankings seamlesslywith further reduced cognition effort.
Multi-Task Algorithm Support	Yang et al. [2] propose a meta-learning method to select classification algorithms based on dataset features and optimal parameters, but focus only on classification. Roy et al. [3] developed CurFi for selecting regression models using R^2 via curve fitting, but it is limited to regression tasks only.	Hybrirank extends these ideas by supporting classification, regression, and clusteringwithin a single system, using task-specific, real-time metric evaluation and a unified benchmarking process across all ML task types.
Neural Extension: MLP	Hollmann et al. [4] show that well-pretrained neural models like TabPFN can match or outperform tree-based methods on tabular data. Hutter et al. [5] present MLPs as key components in modern	Hybrirank incorporates MLPs alongside classical models in both classification and regression to ensure algorithm diversity and enable ranking across both traditional and neural paradigms,

	AutoML systems, highlighting their practical relevance.	serving as an add-on to conventional systems.
Optimal Cluster Number Detection (OCND)	Mahmud et al. [6] estimate cluster counts using ensemble analysis over multiple random subsamples. Sowan et al. [7] estimate the optimal number of clusters by aggregating internal validation indices within a single KMeans-based setup, but both are limited in algorithm variety and the number of validation metrics considered.	Hybrirank's OCND advances this by integrating five clustering algorithms, diverse internal metrics, and inertia-based elbow detection. It uses consensus voting,search range fork, filters unstable results, optionally scales data, and applies fallback logic yielding a robust, interpretable solution for estimating optimal cluster number (k).
Clustering Algorithm Scoring	Xu and Tian [8] provide a detailed taxonomy of clustering techniques, while Rodriguez et al. [9] empirically compare their performance across varied data scenarios. However, both focus on analysis and benchmarking, without offering a method for computing algorithm scores.	Hybrirank advances this by computing internal clustering metrics (e.g., Silhouette, Davies-Bouldin) to generate quantitative scores for each algorithm in its benchmarking module, enabling objective comparison and ranking.
Unified Real-Time Benchmarking	Garouani et al. [10] proposed a meta-feature-based selector for classification and regression using historical data, but it does not compute live metrics or support clustering.	Hybrirank complements this by performing real-time benchmarking across classification, regression, and clustering using task-aware dynamic metrics within a unified, adaptive ranking framework.
Per-Metric Algorithm Ranking Tables (PMART)	Tornede et al. [11] propose a meta-learning framework using rank aggregation (e.g., Borda count) across algorithm selectors, but lack per-metric transparency.	Hybrirank extends this by introducing interpretable per-metric ranking tables PMART for granular algorithm evaluation and final scoring across ML tasks, a novel decision-support feature not addressed in prior work.
Avg-for-Tie: Tie Disambiguator (AFT)	Corsi and Urbano [12] propose tie-aware variants of Rank-Biased Overlap (RBOa/RBOb) to fairly evaluate ranked lists with ties.	Hybrirank draws on this idea by introducing AFT (Avg_{tie}), a task-aware averaging strategy that resolves tie scores by computing refined averages over core evaluation metrics, tailored to the specific ML task type.
Unified Metric-wise Rank Aggregation (UMRA)	Chatterjee et al. [13] unify metric-wise ranks into a stable explanation output for XAI evaluation, but lack direction handling and fair tie resolution. Bałchanowski and Boryczka [14] show that rank aggregation improves recommender accuracy but focus on consensus over interpretability.	Hybrirank's UMRA extends these by ranking algorithms per metric with direction-aware handling, summing total ranks, and resolving ties using normalized averages of both direct and inverse metric values, producing a fair, interpretable final rank table for decision support.
Aspect Option Weighting (AOW)	Hatefi [15] proposes IROC to derive calibrated weights from	Hybrirank builds on this by using expert-defined static scores (AOW)

	user-ranked criteria using refined rank-to-weight conversion, but requires manual ranking input. Zeeshan et al. [16] introduced a ranking framework based on weighted user preferences for algorithm selection, but itrelies on manual weight input and lacks backend automation.	for selectable options. Users select relevant aspects, and the system applies preset utility values (e.g., "Fast" = 1.550) to automatically score algorithms through backend matching, enabling consistent, effort-free preference mapping.
Coverage-Aware Boosting Strategy (CABS)	Zhang et al. [17] propose SBCR, which enhances ranking calibration by applying deterministic inference-time boosts. Corsi and Urbano [12] introduce a rule-based method for handling ties in ranked outputs by distinguishing between equally scored item, but neither considers user-defined aspect coverage.	Hybrirank's CABS builds on these by introducing a tiered boosting mechanism applied during tie cases. The boost strength is based on how many required aspects an algorithm satisfies, enabling nuanced, coverage-sensitive ranking calibration tailored to user priorities.
Context-Oriented Rank Engine (CORE)	Ahsen et al. [18] propose SUMMA, an unsupervised method that combines classifier outputs using confidence-weighted aggregation. Fujita et al. [19] introduce HPA, which constructs a pseudo-answer and selects the most aligned ranking models for ensemble ranking, but both lack integration of diverse ranking sources.	Hybrirank's CORE builds on these by fusing dataset- and preference-based ranks using weighted aggregation, penalties, and tie-breaks to produce context-aware model rankings.Other referenced modules contribute to the design of CORE as Hybrirank's central decision-making engine.
Zero-latency Algorithm Preference Re-ranker (ZAPR)	Chen et al. [20] propose CMR, a hypernetwork-based re-ranking model that adapts to user preferences but requires model regeneration, introducing latency. Wang et al. [21] present a preference-driven method using precomputed utility scores but lack real-time responsiveness.	Hybrirank's ZAPR improves on both by supporting instant re-ranking through lightweight re-computation over precomputed results when preferences change achieving zero-latency interaction without retraining or new model generation.

6. Conclusion:

Hybrirank is a unified framework for algorithm selection across classification, regression, and clustering, combining dataset benchmarking with user-driven preference modeling. To assess its performance across different tasks, Hybrirank's rankings are evaluated using external correlation metrics, comparing them against both dataset-based and preference-based ranks.

Across the three distinct algorithm ranking scenarios (classification, regression, and clustering) considered from the above results section, the Master Rank (Hybrirank) delivers a clear efficiency boost over both Dataset and Preferences Ranks. Spearman rank correlation (ρ) with dissimilarity defined as $1 - \rho$, and Kendall rank correlation (τ) with dissimilarity derived from Kendall distance, are used to validate the ranking efficiency. Against the Dataset Rank, Hybrirank shows efficiency gains, averaging 80.92% with Spearman (range: 78.72%–

84.03% from above three results, varies with dataset) and 66.06% with Kendall (range: 62.50%–70.97% from the above three results), a stable improvement since the dataset remains fixed. Against the Preferences Rank, Hybrirank shows efficiency gains, averaging 52.69% with Spearman (range: 36.97%–61.54% from the three results, varies with preferences) and 33.94% with Kendall (range: 29.03%–37.50% from the three results), though the gains are more volatile due to varying task-specific preferences. The variability in preferences affects not only the Preferences Rank but also the Master Rank, since it incorporates them with about half the weight. This is not a drawback but reflects task-specific flexibility: with 23 to 24 questions each offering 3 to 4 options in preference mode, numerous option configurations are possible. This diversity gives preferences strong influence, and alignment with algorithm qualities can approach complete efficacy. Spearman and Kendall analyses show that Hybrirank significantly improves algorithm ranking quality, achieving strong alignment with both Dataset and Preferences, as shown in (Figure 12).

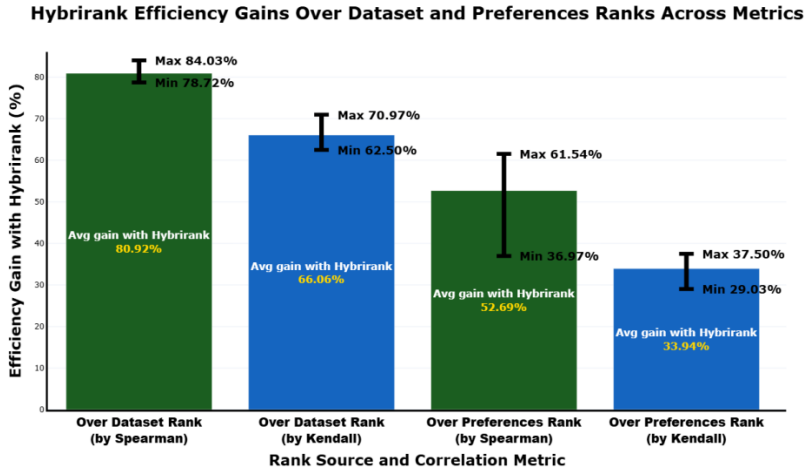

Fig. 12. Hybrirank Efficiency Gains over Dataset and Prefernces Ranks Across Metrics

The Master Rank (Hybrirank) combines performance and preferences to deliver more task-relevant and data-oriented accurate algorithm rankings. This does not imply that Dataset Rank or Preferences Rank is ineffective, and each remains valuable when only one type of input is available. Future work will explore integration with deep learning pipelines, expand support to time-series and multi-modal data inputs, and incorporate interactive feedback for dynamic preference learning. These enhancements will position Hybrirank as a core decision-support engine in next-generation ML ecosystems.

7. Acknowledgments

7.1 Conflict of Interest and Funding Disclosure

Conflict of Interest: The authors declare that there are no financial or personal relationships that could have influenced the research, analysis, or conclusions presented in this study.

Funding: This research received no specific funding from any public, commercial, or non-profit funding agency.

7.2 Author Contributions

Pavan Manikanta Raghava Kasturi1: Conceptualization, methodology, algorithm design, implementation, experimentation, data analysis, and manuscript preparation.

Dr. Vithya Ganesan 2: Supervision, research guidance, critical review, and manuscript revision.

Both authors reviewed and approved the final version of the manuscript.

7.3 Data Availability:

The custom dataset developed for this study is available from the corresponding author upon reasonable request.

REFERENCES

1. Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J., & Fernández-Leal, Á. Human-in-the-loop machine learning: A state of the art. *Artificial Intelligence Review.* (2023). *56* : 3005–3054. https://doi.org/10.1007/s10462-022-10246-w
2. Yang, Z., Li, H., Ali, S., & Ao, Y. Choosing classification algorithms and its optimum parameters based on data set characteristics. *Journal of Computers.* (2017). *28(5)* : 26–38. https://doi.org/10.3966/199115992017102805003
3. Roy, A., Al Zubayer, T., Tabassum, N., Islam, M. N., & Sattar, M. A. CurFi: An automated tool to find the best regression analysis model using curve fitting.*Engineering Reports.* (2022). *4(12)* : e12522. https://doi.org/10.1002/eng2.12522
4. Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., Schirrmeister, R. T., & Hutter, F. Accurate predictions on small data with a tabular foundation model. *Nature.* (2025).*637* :319–326. https://doi.org/10.1038/s41586-024-08328-6
5. Hutter, F., Kotthoff, L., &Vanschoren, J. (Eds.). Automated machine learning: Methods, systems, challenges. *Springer*. (2019). 140-141. https://doi.org/10.1007/978-3-030-05318-5
6. Mahmud, M. S., Huang, J. Z., Ruby, R., & Wu, K. An ensemble method for estimating the number of clusters in a big data set using multiple random samples. *Journal of Big Data.* (2023). *10(1)* : Article 40. https://doi.org/10.1186/s40537-023-00709-4
7. Sowan, B., Hong, T.-P., Al-Qerem, A., Alauthman, M., & Matar, N. Ensembling validation indices to estimate the optimal number of clusters. *Applied Intelligence*. (2022). *53* : 9933–9957. https://doi.org/10.1007/s10489-022-03939-w
8. Xu, D., & Tian, Y. A comprehensive survey of clustering algorithms. *Annals of Data Science.* (2015). *2(2)* : 165–193. https://doi.org/10.1007/s40745-015-0040-1
9. Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. F., & Rodrigues, F. A. Clustering algorithms: A comparative approach. *PLOS ONE.* (2019). *14(1)* : e0210236. https://doi.org/10.1371/journal.pone.0210236
10. Garouani, M., Ahmad, A., Bouneffa, M., Hamlich, M., Bourguin, G., & Lewandowski, A. Using meta-learning for automated algorithms selection and configuration: An experimental framework for industrial big data. *Journal of Big Data*. (2022). *9* : Article 57. https://doi.org/10.1186/s40537-022-00612-4
11. Tornede, A., Gehring, L., Tornede, T., &Hüllermeier, E. Algorithm selection on a meta level. *Machine Learning*. (2022). *112* : 1253–1286. https://doi.org/10.1007/s10994-022-06161-4
12. Corsi, M., & Urbano, J. The treatment of ties in rank-biased overlap. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24)*. (2024). 251–260. https://doi.org/10.1145/3626772.3657700
13. Chatterjee, S., Colombo, E. R., & Raimundo, M. M. Multi-criteria rank-based aggregation for explainable AI.*International Joint Conference on Neural Networks (IJCNN)*. (2025). https://doi.org/10.48550/arXiv.2505.24612
14. Bałchanowski, M., & Boryczka, U. A comparative study of rank aggregation methods in recommendation systems. *Entropy.* (2023). *25(1)* : 132. https://doi.org/10.3390/e25010132
15. Hatefi, M. A. An improved rank order centroid method (IROC) for criteria weight estimation: An application in the engine/vehicle selection problem. *Informatica*. (2023). *34(2)* : 249–270. https://doi.org/10.15388/23-INFOR507
16. Zeeshan, Z., Engrannie, Q. A., Bhatti, U. A., Memon, W. H., Ali, S., Nawaz, S. A., Nizamani, M. M., Mehmood, A., Bhatti, M. A., &Shoukat, M. U. Feature-based multi-criteria recommendation system using a weighted approach with ranking correlation. *Intelligent Decision Technologies.* (2021). *25(4)*. https://doi.org/10.3233/IDA-205388
17. Zhang, S., Liu, H., Bao, W., Yu, E., & Song, Y. A self-boosted framework for calibrated ranking. *Proceedings of the 30th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '24)*. (2024). 6226–6235. https://doi.org/10.1145/3637528.3671570
18. Ahsen, M. E., Vogel, R. M., &Stolovitzky, G. A. Unsupervised Evaluation and Weighted Aggregation of Ranked Classification Predictions. *Journal of Machine Learning Research*. (2019). *20* : 1-40 http://jmlr.org/papers/v20/18-094.html
19. Fujita, S., Kobayashi, H., Okumura, M. Unsupervised Ensemble of Ranking Models for News Comments Using Pseudo Answers. *Advances in Information Retrieval (ECIR 2020).* Lecture Notes in Computer Science(), (2020). vol *12036* : 133–140. *Springer*. https://doi.org/10.1007/978-3-030-45442-5_17
20. Chen, S., Wang, Y., Wen, Z., Li, Z., Zhang, C., Zhang, X., Lin, Q., Zhu, C., & Xu, J. Controllable multi-objective re-ranking with policy hypernetworks. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*. (2023). 3855–3864. https://doi.org/10.1145/3580305.3599796
21. Wang, H., Li, M., Miao, D., Wang, S., Tang, G., Liu, L., Xu, S., & Hu, J. A preference-oriented diversity model based on mutual information in re-ranking for e-commerce search. *Proceedings of the 47th International ACM SIGIR*

Conference on Research and Development in Information Retrieval (SIGIR '24), (2024). 2895–2899. https://doi.org/10.1145/3626772.3661359