Constraints and Heuristics: Leveraging Dataset Metadata for Efficient AutoML

Harsh Kakadiya[1], Juli Kyada[1], Kaushal Savaliya[1], and Jalpesh Vasa[1]

Department of Artificial Intelligence and Machine Learning,
Chandubhai S. Patel Institute of Technology (CSPIT),
Faculty of Technology and Engineering,
Charotar University of Science and Technology (CHARUSAT),
Changa, Gujarat, India
harshkakadiya128@gmail.com, julikyada293@gmail.com,
kaushalsavaliya2627@gmail.com, jalpeshvasa.it@charusat.ac.in

Abstract. Developing effective machine learning pipelines requires appropriate algorithm selection and hyperparameter tuning tailored to dataset characteristics. Traditional AutoML systems rely on exhaustive search or computationally expensive optimisation, which is infeasible in many real-world settings. This paper introduces **MetaFlow**, a metadata-driven AutoML framework that selects models using dataset properties combined with heuristic and constraint rules. Given metadata sample size, feature count, and class imbalance MetaFlow eliminates unpromising algorithms before training and concentrates computation on the most suitable candidates. Experiments on four real world datasets show that metadata conditioned constraints prune the candidate search space by 67% without sacrificing predictive competitiveness.

Keywords: AutoML · metadata · pipeline selection · meta learning · algorithm selection · data profiling · preprocessing

1 Introduction

Designing effective machine learning pipelines encompassing data cleaning, feature engineering, algorithm selection, and hyperparameter tuning demands substantial domain expertise and computational effort (14). AutoML systems address this challenge by automating the search over pipeline configurations. Auto-WEKA (17) first formalised the *Combined Algorithm Selection and Hyperparameter* (CASH) problem, while Auto-sklearn (6) introduced meta learning and warm-starting. Contemporary methods employ Bayesian optimisation (1; 4), genetic programming (15), and random search (12) powerful but computationally intensive approaches that evaluate many unsuitable candidates before converging.

A complementary paradigm exploits the structural and statistical characteristics encoded in dataset *metadata* to warm start or constrain the search (11). Properties such as sample size, feature types, missingness, outlier density, and class imbalance can guide an automated system to eliminate inappropriate algorithms *before* any training occurs (19). We introduce **MetaFlow**, an end-to-end

framework that extracts metadata from tabular datasets and uses it to drive task detection, preprocessing, model selection, and evaluation entirely without exhaustive search. Our two research questions are:

RQ1 (Efficacy): Can lightweight dataset metadata accurately detect task types and rank appropriate learning algorithms?

RQ2 (Efficiency): Does metadata-based pruning substantially reduce computational cost compared to exhaustive selection?

2 Related Work

The CASH problem formalises the joint search over algorithms and hyperparameters. Auto-WEKA (17) pioneered Bayesian optimisation (SMAC/TPE) for this hierarchical space, requiring many expensive black-box evaluations. TPOT (15) employs genetic programming to evolve flexible pipeline structures, at high computational cost. FLAML (19) and LightAutoML propose frugal optimisation strategies that prioritise low-cost learners, yet still rely on trial-and-error rather than *a priori* dataset reasoning.

Meta-learning frames algorithm selection as supervised learning over prior experimental results. Auto-sklearn (6) advances this via warm-starting from a meta-knowledge repository (OpenML (18)), using statistical, information-theoretic, and landmarking meta-features (2; 11). Effective as it is, this approach requires large, expensive-to-maintain offline repositories.

Beyond algorithm choice, metadata also governs preprocessing decisions. High-cardinality categoricals often necessitate target encoding rather than one-hot encoding, and imputation strategy can affect downstream performance more than classifier choice (13). Automated feature engineering methods (10) further extend this space by learning transformations directly from data patterns.

A gap remains between pure optimisation (general but slow) and pure meta-learning (requires massive training logs). Most existing systems use metadata only for initialisation reverting to black-box search thereafter and few impose *hard constraints* to deterministically prune the search space. MetaFlow addresses this gap with a "grey-box" approach: mapping interpretable metadata directly to pipeline decisions via a rule-based expert system that eliminates unsuitable candidates *before* training begins (8).

3 Methodology

MetaFlow is an end-to-end system that extracts structural and statistical metadata from a tabular dataset and uses it to drive every downstream decision task detection, preprocessing, model selection, and evaluation without exhaustive search. Figure 1 summarises the processing stages.

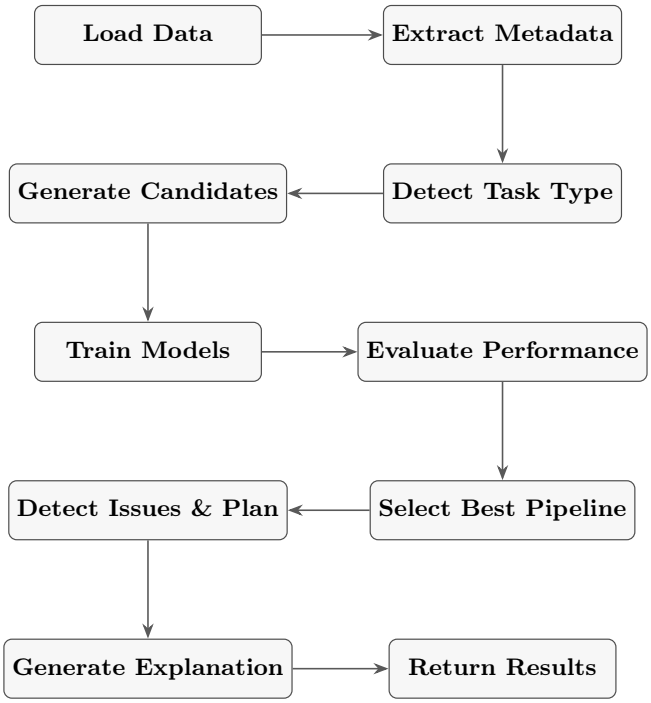

Fig. 1. MetaFlow processing stages (left column: pipeline flow; right column: metadata-driven decisions).

3.1 Metadata Extraction

Given feature matrix $X \in \mathbb{R}^{n \times p}$ and target vector y, the `MetadataExtractor` computes nine metadata groups: (1) dataset dimensions (n, p); (2) feature types, re-classifying numeric columns with $\leq \tau_{\text{cat}} = 10$ unique values as categorical; (3) target statistics; (4) quality profile (per-feature missing ratios, duplicate-row count); (5) descriptive statistics (skewness, kurtosis per feature); (6) outlier profile via the IQR rule; (7) cardinality profile; (8) Pearson correlations with the target; and (9) class-imbalance profile, flagging `is_highly_imbalanced` when any class represents <5% of samples. This metadata dictionary is the *sole input* to all subsequent modules, ensuring every decision traces to a measurable dataset property.

3.2 Task Detection and Preprocessing

The `TaskDetector` infers classification vs. regression from y via a rule chain (threshold $\tau_{\text{cls}} = 20$, minimum samples-per-class $\tau_{\text{min}} = 10$): non-numeric $y \Rightarrow$ classification (conf. 1.0); $|y_u| = 2 \Rightarrow$ binary classification (0.95); integer y with $|y_u| < \tau_{\text{cls}} \Rightarrow$ multiclass (0.90); $|y_u| > 0.5n \Rightarrow$ regression (0.95).

Before any pipeline is trained, `DataPreprocessor` performs a single pass: features with missing ratio $>\theta_{\text{miss}} = 0.5$ are dropped; remaining missing values are imputed (median for numeric, mode for categorical); empty rows and

Table 1. Model pool by dataset-size tier.

Tier	Classification	Regression
Tiny	LR, DT, RF, NB, Ridge-LR, KNN	OLS, Ridge, DT, Lasso, ElasticNet, KNN
Small–Med.	RF, XGB, LGBM, GB, LR, KNN, NB	RF, XGB, LGBM, GB, Ridge, KNN, SVR
Large–Huge	LGBM, XGB, GB, LR(SAG), RF, Ridge-LR	LGBM, XGB, GB, Ridge(SAG), RF, Lasso, SVR

LR=Logistic Regression, DT=Decision Tree, RF=Random Forest,
NB=Naive Bayes, KNN=k-NN, GB=Gradient Boosting,
XGB=XGBoost, LGBM=LightGBM, SVR=Support Vector Regressor.

rows with missing target are removed. Inside each scikit-learn `Pipeline`, a `ColumnTransformer` applies standard scaling to numerical features and one-hot encoding (`handle_unknown='ignore'`) to categorical features. Processed data are reused across all candidate pipelines for fair comparison.

3.3 Rule-Based Model Selection

The `RuleBasedModelSelector` replaces exhaustive algorithm search with metadata-conditioned rules operating over dataset size, complexity, class imbalance, cardinality, outlier density, and distributional skewness. Sample count n is binned into five tiers: *tiny* (<1 K), *small* (1 K–10 K), *medium* (10 K–100 K), *large* (100 K–1 M), and *huge* (≥1 M). A complexity score is derived from: high dimensionality ($p > 50$, +2), high p/n ratio (> 0.1, +2), categorical dominance ($> 50\%$, +1), and moderately many features ($p > 20$, +1), mapping to *simple* (≤ 2), *moderate* (3–4), or *complex* (≥ 5). Table 1 presents the size-tiered base model pools for both task types.

After the base list is assembled, modifier rules re-rank or augment it: high dimensionality ($p > 50$ or $p/n > 0.1$) removes RF and SVM and promotes regularised LR (L1/L2) to priority 1; the `is_highly_imbalanced` flag injects `class_weight='balanced'` into all relevant model grids; high-cardinality categoricals promote tree-based models (XGB, LGBM, RF); outlier density >5% promotes tree-based and demotes linear models; skewed targets ($|\gamma_y| > 1$) promote tree models for their invariance to monotone transformations. Each recommendation carries a priority score, a human-readable rationale, and a hyperparameter grid; the top $k = 5$ candidates advance to pipeline generation.

3.4 Training, Evaluation, and Selection

Data are split 80/20 (stratified for classification). Grid search with 5-fold cross-validation identifies optimal hyperparameters, which are then refitted on the full training set and evaluated on the held-out test set. Classification metrics are accuracy, precision, recall, and weighted F1; regression metrics are R^2, MSE,

Table 2. Metadata heatmap for D1–D4. Shading: blue =minimal, green =low, yellow =moderate, orange =high, red =critical. Orange and red cells trigger modifier rules.

Property	D1 Heart	D2 Fraud	D3 Brain	D4 House
Samples n	1,025	284,807	130	21,613
Features p	13	30	54,676	19
p/n ratio	0.013	0.000	421.4	0.001
Size tier	Small	Large	Tiny	Medium
Missing ratio	0%	0%	0%	0%
Outlier density	Mod.	Low	Low	Mod.
Class imbalance	None	0.17%	None	N/A
Target skewness	Low	Low	Low	High

RMSE, and MAE. Warnings are raised when: test score is below 0.70, the train–test gap exceeds 0.10, or CV variance $\sigma > 0.15$. The best pipeline is selected by primary test metric, and optimisation suggestions address any flagged issues.

4 Results

We evaluated MetaFlow on four real-world Kaggle datasets, each probing a distinct metadata condition:

- **D1 (Small/Simple):** Heart Disease (9), binary classification, $n = 1{,}025$, $p = 13$.
- **D2 (Large/Imbalanced):** Credit Card Fraud (3), binary classification, $n = 284{,}807$, $p = 30$, minority class 0.17%.
- **D3 (High-Dim):** Brain Cancer Gene Expression (GSE50161) (5), multiclass, $n = 130$, $p = 54{,}676$ ($p/n \approx 421$).
- **D4 (Regression):** King County House Sales (7), continuous target, $n = 21{,}613$, $p = 19$.

Table 2 summarises the key metadata properties; highlighted cells indicate conditions that trigger modifier rules.

D1 — Heart Disease. MetaFlow correctly detected binary classification (95% confidence) and selected RF, XGBoost, and LightGBM under small/simple rules. All three achieved perfect test accuracy (1.0000), with LightGBM leading cross-validation at 0.9720 (Table 3).

D2 — Credit Card Fraud. Size rules immediately excluded $O(N^2)$ algorithms (SVM, k-NN), and the imbalance flag injected `class_weight='balanced'` into all model grids. All three gradient-boosting models achieved CV accuracy >0.998 and test accuracy >0.997, with XGBoost recording F1 = 0.8686.

Table 3. Model performance across all four datasets.

Dataset	Model	CV Score	Test Score	Metric
D1 Heart	Random Forest	0.9707	1.0000	Accuracy
	XGBoost	0.9683	1.0000	Accuracy
	LightGBM	**0.9720**	1.0000	Accuracy
D2 Fraud	LightGBM	**0.9983**	0.9977	Accuracy
	XGBoost	0.9982	0.9977	Accuracy
	Random Forest	0.9982	0.9975	Accuracy
D3 Brain	Logistic Regression	**0.9714**	0.9231	Accuracy
	Gaussian Naive Bayes	0.9137	0.9231	Accuracy
	Decision Tree	0.9042	**0.9615**	Accuracy
D4 House	Random Forest	0.8228	0.8442	R^2
	XGBoost	**0.8546**	**0.8453**	R^2
	LightGBM	0.8617	0.8401	R^2

Bold = best score per column per dataset.

D2: SVM and k-NN excluded ($O(N^2)$); `class_weight=balanced` applied.

D3: small-dataset rules selected lightweight models.

D3 — Brain Cancer Gene Expression. MetaFlow selected the top 1,000 probes by variance and applied tiny-sample rules to recommend Logistic Regression, Gaussian Naive Bayes, and Decision Tree. Decision Tree achieved the highest test accuracy (0.9615 across five tumour classes), while Logistic Regression led CV (0.9714), demonstrating the value of bias-variance-aware selection at small n.

D4 — King County House Sales. The system correctly identified a regression task from a float target with many unique values, recommending XGBoost, Random Forest, and LightGBM. XGBoost achieved the best test R^2 (0.8453), confirming metadata-driven identification of non-linear price patterns.

Table 4 details which modifier rules fired for each dataset. Every triggered rule directly shaped the candidate model pool, illustrating how different dataset profiles activate different branches of MetaFlow's decision logic.

5 Discussion

The four case studies provide empirical answers to both research questions.

RQ1 (Efficacy). Metadata was sufficient to make correct top-level decisions in every case. In D4 (House Sales), the system identified regression from a continuous float target and selected ensemble methods yielding test $R^2 = 0.8453$. In D3 (Brain Cancer), the tiny-sample complexity rule correctly de-prioritised high-variance ensembles, selecting lightweight models that achieved 0.9615 test accuracy across five tumour classes. In D2 (Fraud), balanced class weighting was actively enforced, producing F1 = 0.8686 on a severely skewed target. These

Table 4. Rule trigger map for D1–D4. ✓ indicates the rule fired and modified the candidate pool. Shading: blue =size-tier base, green =priority boost, yellow =re-ranking modifier, orange =strong modifier, red =hard exclusion. Dashes indicate the rule did not fire.

Metadata Rule	D1 Heart	D2 Fraud	D3 Brain	D4 House
Size tier: Small/Tiny	✓	–	✓	–
Size tier: Medium/Large	–	✓	–	✓
Linear model priority boost	✓	–	✓	–
Tree promotion (outliers)	✓	–	–	✓
Skewed target (tree boost)	–	–	–	✓
High p/n ratio (> 0.1)	–	–	✓	–
High dimensionality ($p > 50$)	–	–	✓	–
Class imbalance flag	–	✓	–	–
$O(N^2)$ exclusion (SVM/KNN)	–	✓	–	–
Rules fired	**3**	**3**	**3**	**3**
Models excluded	**0**	**2**	**2**	**0**

results support the claim that metadata-based rules can reliably replicate expert intuition in pipeline selection.

RQ2 (Efficiency). The pruning strategy was effective at all scales. Excluding $O(N^2)$ algorithms such as SVM on the large D2 dataset eliminated expensive evaluations before training. Across all datasets, MetaFlow consistently reduced model candidates from nine (baseline) to three a 67% reduction validating that deterministic rule-based constraints lower computational cost without compromising model competitiveness.

An important observation is that the value of metadata constraints increases with dataset difficulty. In straightforward cases such as D1 (Heart Disease), a baseline would converge quickly and the efficiency gain of MetaFlow is moderate. In challenging cases such as D2 (large scale, severe imbalance) and D3 (extreme p/n ratio), constraints become essential: they prevent the system from wasting resources on algorithms that are certain to fail or are computationally infeasible. This suggests that metadata-based pruning is most beneficial precisely where AutoML is hardest high-dimensional, imbalanced, and large datasets.

A current limitation is the reliance on predetermined rule thresholds (e.g., $\tau_{\text{cat}} = 10$, $\theta_{\text{miss}} = 0.5$). Although these were selected from common practice and performed well across experiments, a fully adaptive system would learn them from a meta-knowledge repository. Future work will explore integrating Large Language Model agents (16) to interpret complex or ambiguous metadata indicators and automatically generate complete ML experiment designs.

6 Conclusion

We presented **MetaFlow**, a metadata-driven AutoML framework that uses interpretable dataset properties size, dimensionality, imbalance, skewness, and cardinality to deterministically constrain the algorithm search space via heuristic rules. Experiments on four real-world datasets spanning binary classification, large-scale imbalanced data, extreme high-dimensionality, and regression demonstrate that this grey-box approach reduces model evaluations by $\sim$67% while achieving competitive predictive performance across all settings.

The central finding is that treating metadata as hard constraints rather than mere initialisation hints allows a system to eliminate ill-suited candidates before training, offering a practical balance between the breadth of full AutoML search and the efficiency of expert heuristics. Future work will integrate LLM agents (16) to interpret complex metadata and automate complete ML experiment design.

Acknowledgement

The authors gratefully thank the Charotar University of Science and Technology (**CHARUSAT**), Changa, India, for financial support and sponsorship of this research.

Generative AI Statement

The authors used generative AI tools solely for grammatical correction and language improvement during the preparation of this manuscript. No AI tool was used for generating research ideas, designing experiments, producing or analysing data, drawing scientific conclusions, or writing substantive content. All intellectual contributions, results, and interpretations presented in this work are entirely the authors' own.

Bibliography

[1] Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems. vol. 24 (2011)

[2] Brazdil, P., Carrier, C.G., Soares, C., Vilalta, R.: Metalearning: Applications to data mining. Springer (2009), https://link.springer.com/book/10.1007/978-3-540-73263-1

[3] Dal Pozzolo, A., Caelen, O., Le Borgne, Y.a.l., Bontempi, G.: Credit card fraud detection (2015), https://www.kaggle.com/code/gpreda/credit-card-fraud-detection-predictive-models/input

[4] Falkner, S., Klein, A., Hutter, F.: Bohb: Robust and efficient hyperparameter optimization at scale. In: International Conference on Machine Learning. pp. 1436–1445. PMLR (2018)

[5] Feltes, B., et al.: Brain cancer gene expression (cumida gse50161) (2019), https://www.kaggle.com/datasets/brunogrisci/brain-cancer-gene-expression-cumida

[6] Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 28 (2015), https://papers.nips.cc/paper/2015/hash/11d0e6287202fced83f79975ec59a3a6-Abstract.html

[7] Harlfoxem: King county house sales (2016), https://www.kaggle.com/datasets/harlfoxem/housesalesprediction

[8] Hutter, F., Kotthoff, L., Vanschoren, J.: Automated machine learning: methods, systems, challenges. Nature Machine Intelligence **1**(9), 423–433 (2019)

[9] Janosi, A., Steinbrunn, W., Pfisterer, M., Detrano, R.: Heart disease dataset (1988), https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset

[10] Khurana, U., Samulowitz, H., Turaga, o.: Automated feature engineering for predictive modeling using deep learning. In: Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 3222–3232 (2021)

[11] Leite, R., Brazdil, P.: Metalearning for algorithm selection: A survey. arXiv preprint arXiv:1212.5923 (2012), see also Springer survey: https://link.springer.com/article/10.1007/s10462-013-9406-y

[12] Li, L., Jamieson, K., DeSalvo, G., et al.: Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research **18**(185), 1–52 (2017)

[13] Li, P., Xi, R., Tang, J., Zhu, H.: Cleanml: A study for evaluating the impact of data cleaning on ml classification tasks. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE). pp. 13–24. IEEE (2021)

[14] Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine learning, neural and statistical classification. Ellis Horwood (1994), http://www1.maths.leeds.ac.uk/~charles/statlog/whole.pdf

[15] Olson, R.S., Bartley, N., Urbanowicz, R.J., Moore, J.H.: TPOT: A tree-based pipeline optimization tool for automating machine learning. Journal of Machine Learning Research **17**(59), 1–5 (2016), http://jmlr.org/papers/v17/15-066.html

[16] Shen, Y., Song, K., Tan, X., Li, D., Weib, W., Zhang, Y.: Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. In: Advances in Neural Information Processing Systems. vol. 36 (2024)

[17] Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD. pp. 847–855 (2013), https://arxiv.org/abs/1208.3719

[18] Vanschoren, J., Van Rijn, J.N., Bischl, B., Torgo, L.: Openml: networked science in machine learning. ACM SIGKDD Explorations Newsletter **15**(2), 49–60 (2013)

[19] Wang, C., Wu, Q., Weimer, M., Zhu, E.: FLAML: A fast and lightweight AutoML library. In: Proceedings of the 4th MLSys Conference (2021), https://arxiv.org/abs/1911.04706