International Journal of Innovative Research in Engineering
Volume 7, Issue 6 (November - December 2026), PP: 161-176.
https://www.doi.org/10.59256/ijire.20260706019
www.theijire.com
ISSN No: 2582-8746

A Comparative Analysis of Data Preprocessing Methods in Business Analytics

Divya B[1], Maheboob Pasha[2], Koulik De Sarkar[3]
[1, 2, 3]Department of Master of Computer Applications, Acharya Institute of Technology, Bengaluru, Karnataka, India.

How to cite this paper:
Divya B[1], Maheboob Pasha[2], Koulik De Sarkar[3], "A Comparative Analysis of Data Preprocessing Methods in Business Analytics", IJIRE-V7I6-161-176.

Copyright © 2026 by author(s) and Fifth Dimension Research Publication. This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/

Abstract: *Business analytics models for churn, credit, marketing, and sales depend on how raw data are prepared, yet real business data are incomplete, skewed, measured on different scales, and full of categorical fields. Earlier studies usually test one preprocessing step at a time, so the joint effect of several steps on business tasks is still unclear. This paper presents a leakage-free full-factorial benchmark that tests five preprocessing stages together: missing-value imputation (deletion, mean/mode, K-Nearest Neighbours), outlier treatment (none, interquartile-range clipping), categorical encoding (One-Hot, Label, Target), feature scaling (none, Min-Max, Z-score), and dimensionality reduction (none, Principal Component Analysis (PCA), mutual information). All 162 supervised and 72 clustering pipelines are fitted inside 5-fold cross-validation on Telco Customer Churn, UCI Bank Marketing, and Online Retail II, and are scored with Logistic/Linear Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), XGBoost, and K-Means in 12,510 model fits. Differences are tested with the Friedman test and Holm-corrected Wilcoxon signed-rank tests. Z-score scaling raised the F1-score by 0.105 on average and added 0.532 to the SVM F1-score on Telco, while tree models were unaffected unless PCA was used. PCA lowered F1 by 0.085, Label encoding lowered it by 0.019 compared with One-Hot encoding, and deletion was worse than imputation. KNN imputation cut imputation error by 16% to 33% but gave no significant F1 gain and took 3.7 times longer. IQR clipping lowered the regression error on skewed customer spend for every model. Preprocessing should therefore be chosen by model family and task, and the paper gives evidence-based pipeline rules for business analytics.*

Key Words: *Data Preprocessing, Business Analytics, Missing Value Imputation, Feature Scaling, Categorical Encoding, Dimensionality Reduction, Machine Learning.*

I.INTRODUCTION

A. Background

Modern businesses use analytics and machine learning (ML) to guide decisions such as dynamic pricing, demand forecasting, credit risk assessment, fraud detection, and customer retention. The accuracy and stability of these models depend strongly on the quality of the input data [1]. Raw data often contains inconsistencies, missing values, and noise [1]. This holds across business tasks, from market basket analysis of e-commerce transactions [2] to phishing detection with logistic regression, support vector machines, and decision trees [3].

B. Problem Statement

Data preprocessing turns raw data into a clean, numeric form that models can use [1]. It has four main parts: data cleaning, integration, transformation, and reduction [1], [4]. Real business data is rarely ready for modeling. It often has missing values (for example, customer details that were never recorded), extreme outliers (for example, very large bulk orders or data entry errors), features on very different scales, and categorical fields with many levels [5], [6], [7], such as applicant attributes in loan approval data [8]. Unsuitable methods, or fitting a transformation on the whole dataset before splitting it, can distort the features and leak validation information into the model [9].

C. Motivation

Analysts often choose preprocessing steps by habit or by library defaults. A poor choice can waste computing time, hide a good model, or leak validation data into training [9]. Earlier benchmarks show that the best imputer, scaler, or encoder changes with the model and the data [10], [11], [12], so a rule taken from one step or one domain may not transfer to business data, where all five problems appear together. Business teams also need to know whether a costly step, such as KNN imputation, is worth its extra run time. These practical questions motivate a joint, leakage-free comparison on real business datasets.

D. Research Gap

Earlier studies look at one step, one model family, or one domain (Section II). Few of them test cleaning, transformation, and reduction together on business data, and none of the reviewed studies compares all five stages across classification, regression, and clustering with a leakage-free design. Section II-C gives the detailed gap analysis.

E. Objectives and Purpose

The purpose of this paper is to build a standard experiment that compares data preprocessing pipelines in business analytics. ML models are used only as a measuring tool. Each pipeline is judged by how it changes model accuracy, cluster quality, and run time. The objectives are:

1. To describe the core preprocessing stages used in business analytics.
2. To test all combinations of five stages (162 pipelines for supervised tasks and 72 for clustering) under identical cross-validation splits.
3. To measure their effect on six supervised and unsupervised algorithms.
4. To give evidence-based guidelines for choosing a pipeline.

F. Significance

The study separates the main effect of each stage from the effect of combining stages. It covers telecommunications, banking, and retail data. The results should help analysts reduce trial and error, avoid data leakage, and decide when costly steps such as KNN imputation or PCA are worth their extra run time.

G. Contributions

The main contributions of this paper are as follows:

- A cross-validation protocol in which every preprocessing step, including listwise deletion and target encoding, is fitted on the training fold only, so no validation information leaks into training (Algorithm 1).
- A full-factorial benchmark of five preprocessing stages (162 supervised and 72 clustering pipelines, 12,510 model fits) on three real business datasets from telecommunications, banking, and retail.
- An ablation-style analysis that isolates the effect of each stage choice while the other four stages are held fixed, together with five tested hypotheses on model-family behaviour (H1–H5).
- Statistical evidence from Friedman and Holm-corrected Wilcoxon tests, together with a run-time (overhead) analysis of each stage.
- Practical pipeline guidelines for common business analytics scenarios (Table IX).

H. Paper Organization

Section II reviews related work and states the research questions and hypotheses. Section III describes the proposed methodology, equations, algorithm, and baseline estimators. Section IV describes the datasets and data splitting. Section V presents the results, the ablation analysis, and the discussion. Section VI concludes the paper, and the references follow.

II. LITERATURE REVIEW

A. Preprocessing Stages and Missing Value Imputation

Preprocessing is usually grouped into data cleaning, integration, transformation, and reduction [1]. Reviews describe the main methods for each group [4]. Missing values can be handled by deleting records or by imputing them [5]. Values can go missing in three ways: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR) [6]. Yu et al. [13] compared one-hot coding of missingness with imputation in credit classification and found that the better choice depended on the share of missing values. Federated and continual learning settings add a further problem, because data arrive over time at several sites and statistics cannot be pooled [14].

Missing-value handling is the most studied step. Hasan et al. [6] reviewed 191 articles published from 2010 to August 2021 and traced which imputation methods, datasets, and evaluation metrics were used; as a review, it does not report one effect size. The benchmark of Jäger et al. [10] gives measured results. They compared six imputation methods (mean/mode, KNN, random forest, a discriminative neural network, a variational autoencoder, and GAIN) on 69 datasets, with 1% to 50% of values missing under MCAR, MAR, and MNAR patterns. KNN, random forest, and the neural network most often ranked among the top three for imputation accuracy, while mean/mode did well mainly for categorical columns at high MNAR rates. When the imputation model was trained on complete data, imputing the missing test values improved the downstream score by 10% to 20% for classification and by about 15% for regression in more than 75% of the experiments. When it was trained on incomplete data, the gains were below 1% in almost all settings. Cost also differed: the average training time, including hyperparameter search, was 0.005 s for mean/mode and 41 s for KNN.

B. Outliers, Feature Scaling, and Categorical Encoding

Business data often has heavy tails and extreme values. These may be errors or real events such as very large orders [7]. A simple and common rule is the IQR rule: a value more than 1.5 times the IQR beyond the first or third quartile is flagged as extreme. Clipping these values (winsorizing) keeps all rows and limits their pull on linear and distance-based models.

Scaling puts features on comparable ranges, and its effect depends strongly on the model. De Amorim et al. [11] tested five scalers on 20 classifiers and 82 imbalanced binary datasets. The mean gap between the best and the worst scaler was close to or above 0.15 for the linear SVM, RBF SVM, Gaussian process, perceptron, and MLP, and close to 0.26 for the Gaussian process. For the linear SVM, the F1 gap exceeded 0.5 on 32% of the datasets. A poor scaler could also be worse than none: the linear SVM had a mean F1 of 0.44 without scaling, 0.38 with Min-Max, 0.36 with MaxAbs, and 0.52 with the Quantile Transformer. Decision trees and tree ensembles (Random Forest, XGBoost, and AdaBoost) were almost unaffected, and the Friedman test rejected equal performance in 28 of 40 tests. Ahsan et al. [15] tested six scalers with eleven classifiers

on one heart disease dataset. Performance was similar with or without scaling for all algorithms except Logistic Regression, KNN, and SVM. KNN accuracy rose from 75% without scaling to 87% with MaxAbs or the Quantile Transformer, whereas the SVM fell from 99% without scaling to 63% with Normalisation and 92% with standardisation. Random Forest and Extra Trees stayed at 98% under every scaler, and no single scaler was best for all models.

Margin-based classifiers are a common case where scaling matters: SVM and ResNet50 models have been compared for abnormal activity recognition in surveillance videos [16], and tree ensembles are widely used for feature extraction from raw EEG signals [17] because they need little rescaling.

Categorical features need numeric codes. Common choices are one-hot, ordinal, and binary coding [18], and target encoding replaces a category with a smoothed average of the target [12]. In applied work, URL and domain attributes are encoded before they are passed to logistic regression, SVM, and decision tree classifiers for phishing detection [3].

For encoding, Pargent et al. [12] compared seven encoding techniques with five algorithms (lasso, random forest, gradient boosting, KNN, and SVM) on 24 regression and classification datasets, one of them with a single feature of more than 30,000 levels. Regularised target encoding (a mixed-model encoder with 5-fold cross-validation) ranked first for every algorithm except the SVM, and it was never convincingly beaten by target-agnostic methods such as one-hot or integer encoding. The advantage was clearest on the nine datasets with many levels, where dropping the high-cardinality features also hurt performance. On the 13 datasets with the fewest levels, traditional encoders were competitive, and on the two datasets with the most levels, target encoding without regularisation overfit badly. Run time was not consistently longer than for one-hot encoding, except with gradient boosting, where it rose substantially.

Study	Preprocessing focus	Data and models	Key finding	Limitation
Hasan et al. (2021) [6]	Missing value imputation	Review of 191 articles	Summarizes imputation methods, datasets, and metrics	Imputation only
Chuang et al. (2026) [9]	Encoding and resampling (with imputation)	2 medical datasets; 6 classifiers	One-hot encoding with ROSE gave the most consistent gains	Medical domain only
Jäger et al. (2021) [10]	Missing value imputation	69 datasets; downstream ML models	Judged imputation by error and by downstream model quality	Imputation only
De Amorim et al. (2023) [11]	Feature scaling	Classification datasets	Scaler choice matters; a poor choice can be worse than none	Scaling only
Ahsan et al. (2021) [15]	Feature scaling	Heart disease data; 11 algorithms, 6 scalers	Results change with the scaler	One dataset; scaling only
Pargent et al. (2022) [12]	Categorical encoding	Regression and classification data; 5 algorithms	Regularized target encoding worked best	Encoding only
Alshdaifat et al. (2021) [19]	Normalization and missing values	18 datasets; classifiers	Significant differences that vary by algorithm	Numeric features; two steps
Johnson et al. (2023) [20]	Feature selection, encoding, imputation	Synthetic data and Lending Club; XGBoost	No single best encoder; missing-indicator imputation did best	One model family
Yu et al. (2022) [13]	One-hot encoding vs imputation for missing data	Credit classification data	Compares two ways of handling missing data	Missing data and credit data only

Table I: Comparison of Recent Studies on Data Preprocessing (2020–2026)

C. Dimensionality Reduction, Research Gap, and Research Questions

Dimensionality reduction follows two routes: feature selection keeps some of the original features, and feature extraction builds new ones [21]. Filter methods such as mutual information rank features by how strongly they relate to the target. Principal Component Analysis (PCA) is a common linear feature-extraction method [22]. Nonlinear optimisation has also been used to extract compact patterns from large IoT data streams before mining [23], and random forest importance has been used to select informative EEG features [17].

Studies that combine several steps are fewer. Alshdaifat et al. [19] crossed three normalisation methods (Min-Max, Z-score, and decimal scaling) with three missing-value strategies (deletion, mean, and KNN imputation) for SVM and neural-network classifiers on 18 datasets. With 10% of values removed, the differences between the nine combinations were significant for both classifiers (neural network $p = 0.001$; SVM $p < 0.001$). Z-score was the best scaler, decimal scaling the worst, and deletion with decimal scaling the worst combination. At 20% missing values, deleting incomplete rows still gave results comparable to mean and KNN imputation. Deletion had the shortest model-building time and KNN imputation the longest. The effect was stronger for the SVM than for the neural network.

Johnson et al. [20], in a preprint limited to XGBoost, compared feature-selection, encoding, and imputation methods on three synthetic datasets (250,000 rows each) and Lending Club loan data (2.26 million rows). No encoder won everywhere: frequency encoding was best on Lending Club but worst on all three synthetic datasets, where one-hot, Helmert, and binary encoding gave near-perfect AUC. For imputation, a missing-value indicator gave the best and most stable scores, mean, median, and decile imputation behaved alike, and decision-tree imputation was the most variable. Chuang et al. [9] crossed one-hot, frequency, and target encoding with SMOTE and ROSE resampling for six classifiers on 412 patients with end-stage renal disease, and then repeated the pipeline on the BRFSS 2015 survey. One-hot encoding with ROSE and LightGBM scored best (AUC 0.940, accuracy 0.932). Under ROSE, frequency and target encoding reached AUC 0.913 and 0.900, and the three encoders did not differ significantly ($p \geq .67$). The resampling choice made the larger difference: ROSE pipelines beat SMOTE in most comparisons ($p \leq .002$), with an average AUC of 0.83 against 0.59.

Research gap. As Table I shows, most studies examine one stage [6], [10], [11], [15], [12], [13]. The multi-step studies are limited to few steps, one model family, or one domain [9], [19], [20]. In real workflows the steps run one after another, and each step changes the data that the next step sees. This study fills the gap with a five-stage factorial benchmark on classification, regression, and clustering tasks.

Research questions.

1. **RQ1 (Stage interaction):** How do combinations of imputation, outlier treatment, encoding, scaling, and reduction affect model performance on business classification, regression, and clustering tasks?
2. **RQ2 (Model family):** Do linear, distance/kernel-based, and tree-based models react differently to scaling and outlier clipping?
3. **RQ3 (Accuracy and time):** Which imputation and encoding methods give the best balance between predictive quality (F1-Score, AUC-ROC, RMSE, imputation error) and run time?
4. **RQ4 (Significance):** Are the differences between pipelines statistically significant under the Friedman and Wilcoxon signed-rank tests?

Hypotheses. Based on the studies above, five hypotheses are tested:

H1: Tree-based models (Decision Tree, Random Forest, XGBoost) are insensitive to feature scaling [11], [15].

H2: Distance-, kernel-, and linear models (SVM, Logistic Regression, K-Means) benefit from feature scaling [11], [15].

H3: KNN imputation gives a lower imputation error and better downstream scores than mean/mode imputation, at a higher run time [10], [19].

H4: IQR clipping improves linear and distance-based models on skewed business data [7].

H5: Target encoding gives a more compact feature matrix and matches or beats One-Hot encoding for tree-based models [12], [20].

III. PROPOSED METHODOLOGY

A. Overview and Workflow

The study is a controlled experiment. Each dataset goes through five stages in a fixed order: (1) missing value imputation, (2) outlier treatment, (3) categorical encoding, (4) feature scaling, and (5) dimensionality reduction. Each stage has two or three methods, and one choice per stage forms one pipeline. For the supervised tasks (classification and regression) there are $3 \times 2 \times 3 \times 3 \times 3 = 162$ pipelines per dataset. K-Means has no target, so Target encoding and Mutual Information selection cannot be used with it. K-Means therefore uses $3 \times 2 \times 2 \times 3 \times 2 = 72$ pipelines.

Fig. 1 shows the workflow. Every fitted value (means, quantiles, encoder tables, PCA axes) is learned from the training fold only and then applied to the validation fold. This stops information from the validation data from leaking into the model.

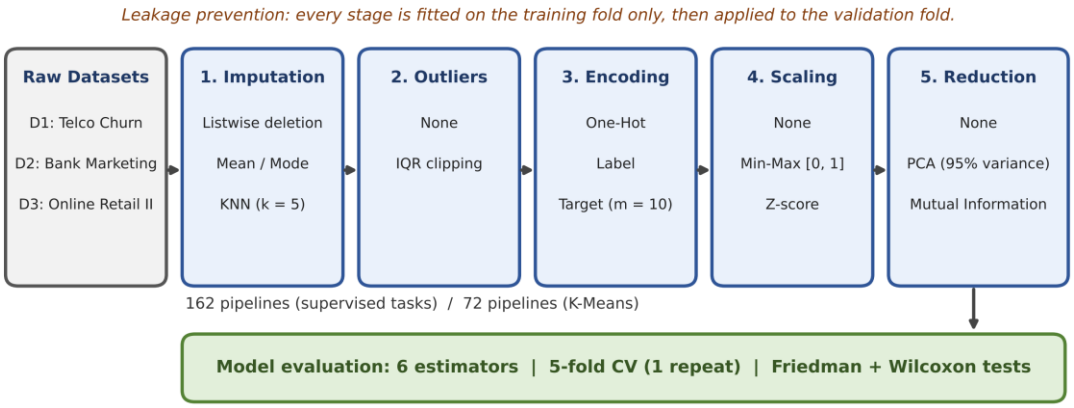

Fig. 1. Proposed five-stage factorial workflow. Every stage is fitted on the training fold only.

B. Notation and Equations

Let $X \in \mathbb{R}^{N \times d}$ be a dataset with N rows and d features, and let y be the target when there is one. All statistics below are computed on the training fold only.

Stage 1: Imputation. KNN imputation fills a missing value with the average of the same feature in the $k = 5$ nearest rows, as in Eq. (1). Distance is computed on standardized features (z), so a feature with large numbers (for example, TotalCharges) does not dominate the search. This also means that Stage 4 scaling does not change the imputed values.

$$\hat{x}_{ij} = \frac{1}{k} \sum_{l \in \mathcal{N}_k(i)} x_{lj}\,, \qquad d(a,b) = \sqrt{\sum_{j \in O_{ab}} \left(z_{aj} - z_{bj}\right)^2} \tag{1}$$

Here $\mathcal{N}_k(i)$ is the set of the k nearest rows to row i, and O_{ab} is the set of features observed in both rows a and b. To measure imputation quality directly, 10% of the numeric values are hidden completely at random (MCAR) and then imputed. Let $\mathcal{M}$ be the set of the N_{miss} hidden cells. The imputation error is given in Eq. (2).

$$\mathrm{RMSE}_{imp} = \sqrt{\frac{1}{N_{miss}} \sum_{(i,j) \in \mathcal{M}} \left(\frac{\hat{x}_{ij} - x_{ij}}{\sigma_j}\right)^2} \tag{2}$$

Here σ_j is the training standard deviation of feature j, so features on different scales count equally. Listwise deletion creates no imputed values, so RMSE_{imp} is not defined for it.

Stage 2: Outlier treatment. IQR clipping limits each numeric value to the range given in Eq. (3).

$$x'_{ij} = \min\left(\max\left(x_{ij},\ Q1_j - 1.5\,IQR_j\right),\ Q3_j + 1.5\,IQR_j\right), \qquad IQR_j = Q3_j - Q1_j \tag{3}$$

Stage 3: Categorical encoding. One-Hot encoding creates one 0/1 column per category. Label encoding gives each category an integer code [18]. Target encoding replaces a category c with a smoothed average of the target, as in Eq. (4) [12].

$$e_c = \lambda(n_c)\,\bar{y}_c + \left(1 - \lambda(n_c)\right)\bar{y}_{global}, \qquad \lambda(n_c) = \frac{n_c}{n_c + m} \tag{4}$$

Here n_c is the number of training rows in category c, $\bar{y}_c$ is the target mean of that category, $\bar{y}_{global}$ is the overall training target mean, and $m = 10$ is the smoothing weight. Rare categories are pulled toward the overall mean. To avoid leakage, the code for each training row is computed from the other inner folds (5-fold out-of-fold coding). Validation rows use the whole training fold. Categories not seen in training are coded as all zeros (One-Hot), -1 (Label), or $\bar{y}_{global}$ (Target).

Stage 4: Feature scaling. Min-Max scaling maps each numeric feature to $[0,1]$, and Z-score scaling gives it zero mean and unit variance, as in Eqs. (5) and (6).

$$x'_{ij} = \frac{x_{ij} - \min_i x_{ij}}{\max_i x_{ij} - \min_i x_{ij}} \tag{5}$$

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}, \qquad \mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}\,, \qquad \sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_{ij} - \mu_j\right)^2} \tag{6}$$

Stage 5: Dimensionality reduction. PCA [22] keeps the smallest number of components p that explain at least 95% of the variance, as in Eq. (7). Mutual Information selection [21] keeps the top 50% of features ranked by their mutual information with the target, as in Eq. (8).

$$\frac{\sum_{r=1}^{p} \lambda_r}{\sum_{r=1}^{d} \lambda_r} \geq 0.95 \tag{7}$$

$$I\left(X_j; Y\right) = \sum_x \sum_y p\,(x,y) \log \frac{p(x,y)}{p(x)\,p(y)} \tag{8}$$

Here λ_r is the r-th largest eigenvalue of the covariance matrix of the features.

Evaluation metrics. Classification uses Accuracy, Precision, Recall, F1-Score, and AUC-ROC. F1-Score is given in Eq. (9). Regression uses MAE, RMSE, and R^2, as in Eqs. (10) to (12). Clustering uses the Silhouette Score, as in Eq. (13) [24].

$$F1 = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \qquad \mathrm{Precision} = \frac{TP}{TP + FP}, \qquad \mathrm{Recall} = \frac{TP}{TP + FN} \tag{9}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{10}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{11}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{12}$$

$$s(i) = \frac{b(i) - a(i)}{\max\big(a(i),\, b(i)\big)} \tag{13}$$

Here $a(i)$ is the mean distance from point i to the other points in its own cluster, and $b(i)$ is the mean distance to the nearest other cluster. The Silhouette Score is the mean of $s(i)$. Because scaling and PCA change distances, all clusterings are scored in one common space: the Z-scored numeric features without clipping or reduction. This makes the scores comparable across pipelines. Total run time is given in Eq. (14).

$$t_{exec} = t_{preprocess} + t_{train} \tag{14}$$

Statistical test. Pipelines are ranked from 1 (best) to k inside each block. One block is one validation split of one dataset–model pair, and N_s is the number of blocks. The Friedman statistic [25] is given in Eq. (15).

$$\chi_F^2 = \frac{12\,N_s}{k(k+1)}\sum_{j=1}^{k} R_j^2 - 3N_s(k+1) \tag{15}$$

Here R_j is the mean rank of pipeline j over all blocks. A small p-value means that at least one pipeline differs from the others. Pairwise Wilcoxon signed-rank tests with Holm correction then compare specific stage choices, such as KNN versus mean imputation or Target versus One-Hot encoding. The significance level is $\alpha = 0.05$. To make blocks comparable across tasks with different units, the scores in each block (F1-Score for classification, negative RMSE for regression) are standardized across pipelines. A stage choice is compared with its reference by the mean difference over matched pipelines, that is, pipelines that differ only in that stage. The Wilcoxon test is applied to these block-level differences.

C. Step-by-Step Algorithm

Step	Action
1. Input	Dataset $D = (X, y)$; pipeline grid Π (162 pipelines for supervised tasks, 72 for K-Means); estimator set M; $K = 5$ folds; $R = 1$ repeat; random seed 42.
2. Prepare data	Remove ID columns and fix blank values (Section IV-D). Hide 10% of the numeric values of D completely at random (MCAR) and keep the true values for Eq. (2).
3. Split	Create $R \times K = 5$ pairs of training and validation sets ($D_{train}^{(f)}$, $D_{val}^{(f)}$). Use stratified splits for classification.
4. Fit	For each split f and pipeline π, fit Stages 1 to 5 on $D_{train}^{(f)}$ only: imputer, IQR limits, encoder tables, scaling statistics, and the PCA or Mutual Information selection.
5. Transform	Apply the fitted stages to $D_{train}^{(f)}$ and $D_{val}^{(f)}$. For listwise deletion, drop incomplete rows from the training set only, and fill validation rows with the training mean/mode so that all pipelines score the same validation rows.
6. Train and score	Train each estimator with fixed hyperparameters on the transformed training set. Record the validation metrics and t_{exec}. For K-Means, fit on the training set, assign validation rows to the nearest centroid, and compute the Silhouette Score in the common space.
7. Imputation test	After Stage 1, compute RMSE_{imp} on the hidden validation cells (Eq. (2)).
8. Test	Rank the pipelines in each block. Compute the Friedman statistic and Wilcoxon signed-rank tests with Holm correction.
9. Output	Report mean ± standard deviation of every metric for every pipeline.

Algorithm 1: Leakage-Free Full Factorial Preprocessing and Cross-Validation

D. Experimental Parameters

Stage	Methods compared	Settings
1. Missing values	Listwise deletion; Mean/Mode; KNN	Deletion drops incomplete training rows only. Mean/Mode uses the mean for numeric and the mode for categorical features. KNN uses $k = 5$ and standardized numeric features; categorical gaps use the mode.
2. Outliers	None; IQR clipping	Numeric features only. Limits are $Q1 - 1.5\,IQR$ and $Q3 + 1.5\,IQR$ from the training fold, as in Eq. (3).
3. Encoding	One-Hot; Label; Target	Target encoding uses $m = 10$ and 5-fold out-of-fold coding, as in Eq. (4). It is not used for K-Means.
4. Scaling	None; Min-Max; Z-score	Numeric features only; statistics come from the training fold, as in Eqs. (5) and (6).
5. Reduction	None; PCA; Mutual Information	PCA keeps 95% of the variance. Mutual Information keeps the top 50% of features. Mutual Information is not used for K-Means.

TableII: Preprocessing Stages, Methods, and Settings

All models use fixed hyperparameters, so any difference in results comes from preprocessing and not from tuning. Tree-based models are included because they perform strongly on tabular data [26]. The experiments use Python 3.12, scikit-learn 1.8.0 [27], and XGBoost 3.4.1 [28] on a single-core virtual machine with 4 GB RAM. Because of this limited computing power, the models are smaller than in a typical study (see Table III and the size paragraph below). K-Means [29] uses $k = 4$ clusters, fixed in advance for all pipelines and not tuned.

Task	Estimators	Fixed hyperparameters	Metrics
Binary classification (D1, D2)	Logistic Regression (LR); Decision Tree (DT); Random Forest (RF); Support Vector Machine (SVM); XGBoost (XGB)	LR: L2 penalty, lbfgs, max_iter = 1000. DT: max_depth = 10, gini. RF: 60 trees, max_features = sqrt. SVM: C = 1, RBF kernel, gamma = scale, trained on a random sample of 2,500 training rows. XGB: 100 trees, learning rate 0.1, max_depth = 6.	Accuracy, Precision, Recall, F1-Score, AUC-ROC, run time
Regression (D3)	Linear Regression; Decision Tree; Random Forest; Support Vector Regression (SVR); XGBoost	Linear: ordinary least squares. DT: max_depth = 10. RF: 60 trees, max_features = sqrt. SVR: C = 1, epsilon = 0.1, RBF kernel, trained on a random sample of 2,500 training rows. XGB: 100 trees, learning rate 0.1, max_depth = 6. The target is log(1 + total spend).	MAE, RMSE, R^2 (log scale), run time
Clustering (D3)	K-Means	k = 4, init = k-means++, n_init = 10, max_iter = 300, seed 42	Silhouette Score, run time
All tasks		Seed 42 for all random steps	RMSE_{imp} (on the hidden cells)

Table III: Estimators, Fixed Hyperparameters, and Metrics

Size of the experiment. Each supervised dataset has 162 pipelines × 5 validation splits × 5 estimators = 4,050 model fits. D3 adds 72 K-Means pipelines × 5 splits = 360 fits. The full study therefore has 4,050 + 4,050 + 4,050 + 360 = 12,510 model fits. To fit the available computing power, the study uses one repeat of 5-fold cross-validation (5 validation splits), smaller forests, and subsamples for the SVM and for D2 (Section IV).

E. Baseline Estimators and Hyperparameter Choice

Six estimators are used as measuring tools. They cover the linear, margin-based, tree-based, and clustering families that analysts use most often on tabular business data.

Logistic and Linear Regression (LR). Linear models that weight each feature; they are fast, easy to explain, and sensitive to feature scale and outliers. Logistic regression is a standard baseline in business classification tasks such as phishing detection [3].

Decision Tree (DT). A single tree that splits on feature thresholds; it handles mixed feature types and is not affected by monotone scaling, but it can overfit [3].

Random Forest (RF). An ensemble of trees grown on bootstrap samples with random feature subsets; it reduces the variance of a single tree and is widely used for tabular prediction and feature selection [26], [17]. Ensemble voting has likewise stabilised predictions in clinical image classification [30].

Support Vector Machine (SVM) and SVR. A margin-based model with an RBF kernel; distances between rows drive the kernel, so it depends strongly on scaling [11], [16].

XGBoost (XGB). A gradient-boosted tree ensemble that adds trees to correct earlier errors; it is a strong tabular baseline [26], [28], and gradient boosting has also performed well in energy forecasting [31].

K-Means. A centroid-based clustering method that assigns each row to the nearest of k centroids [29]; it is distance-based and therefore scale-sensitive. Clustering of customer records is a common step in retail analytics, together with association rule mining [2]. Nonlinear clustering has also been scaled to large datasets with hybrid cloud–quantum solvers [32].

Hyperparameter choice. Hyperparameters are fixed so that every difference comes from preprocessing. Most values are library defaults: C = 1 and gamma = scale for the SVM, 100 trees and a maximum depth of 6 for XGBoost, k = 5 neighbours for KNN imputation, and k-means++ initialisation with 10 restarts for K-Means. The XGBoost learning rate of 0.1 is a common choice for 100 boosting rounds. The Decision Tree depth is limited to 10 to curb overfitting, and Random Forest uses 60 trees instead of the default 100, and the SVM and SVR are trained on 2,500 rows, so that all 12,510 fits run on the single-core machine. PCA keeps 95% of the variance and Mutual Information keeps the top 50% of features, which are common settings in the literature [21], [22]. With the target-encoding smoothing weight m = 10, a category with 10 training rows gets equal weight from its own mean and the global mean, so rare categories are pulled toward the global mean [12]. K-Means uses k = 4 clusters, a typical number of customer segments in recency-frequency-monetary analysis.

IV. DATASET

A. Data Description

Three public business datasets from different sectors are used.

D1: Telco Customer Churn [33]. It has 7,043 customer records and 21 columns about telecommunications services (for example, tenure, contract type, payment method, monthly charges, and total charges). The target is binary customer churn (26.5% churners). Besides the ID and the target, it has 3 numeric and 16 categorical predictors. The TotalCharges column has a few blank values.

D2: UCI Bank Marketing [34]. It has 41,188 telemarketing records and 20 input columns from a Portuguese retail bank. Because of limited computing power, the experiments use a stratified random sample of 6,000 records with the same class balance. They include client details (age, job, marital status, education), campaign history, and economic indicators (Euribor 3-month rate, consumer confidence index). The target is whether the client subscribed to a term deposit (11.3% positive). Several categorical columns contain ‘unknown’ values.

D3: Online Retail II [35]. It has 1,067,371 transaction lines from a UK online retailer between 2009 and 2011. After removing lines without a customer ID, cancelled invoices, and lines with non-positive quantity or price, the lines are combined into one row per customer (5,878 customers). The features are Recency, Frequency, number of different products, average items per invoice, days between first and last purchase, Monetary value, country group, and preferred weekday. It supports regression (total spend) and customer segmentation with K-Means.

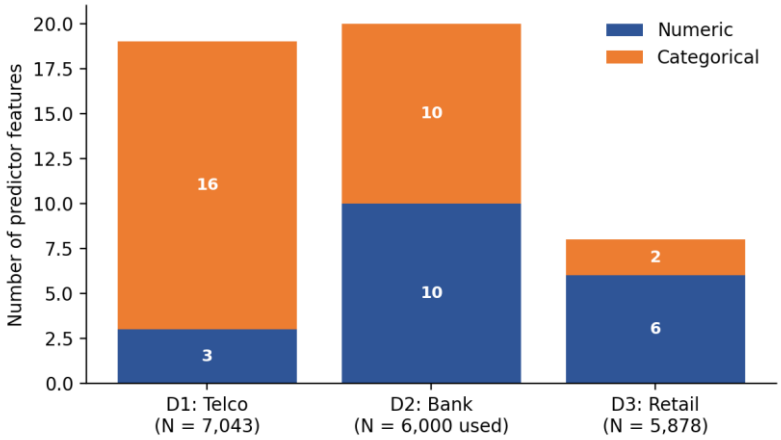

Fig. 2. Number of numeric and categorical predictor features in the datasets (IDs and targets excluded). D3 counts are for K-Means; the regression task excludes Monetary value.

B. Data Sources

D1 is distributed through the IBM sample data sets and Kaggle [33]. D2 [34] and D3 [35] come from the University of California, Irvine (UCI) Machine Learning Repository.

C. Selection Justification

The three datasets cover three business sectors and three analytics tasks. Each one brings a typical data problem. D1 has mixed feature types and a few missing values. D2 has many categorical features, ‘unknown’ values, and imbalanced

classes. D3 has skewed spend values and large wholesale orders that act as outliers. Together they test every stage of the pipeline.

D. Cleaning, Missing-Value Injection, and Splitting

Before the five stages, ID columns are removed (customerID in D1; Invoice and Customer ID after aggregation in D3), and cancelled transactions in D3 are removed. In D1, blank TotalCharges strings are changed to missing values. In D2, 'unknown' entries are changed to missing values so that Stage 1 can be tested. The D2 column duration (length of the last call) is dropped, because it is only known after the call and would leak the target. For the D3 regression task, the target is log(1 + Monetary value), because spend is very skewed, and Monetary value is not used as an input; regression errors are therefore on the log scale. The K-Means task uses all numeric features, including Monetary value.

As described in Section III-B, 10% of the numeric values of every dataset are hidden completely at random (MCAR), in addition to the natural missing values. The models are trained and tested on these data, so Stage 1 changes the downstream results, and the imputation error is measured on the hidden cells. Each dataset is split with 5-fold cross-validation (one repeat, 5 splits, 80% training and 20% validation per split) [36]. Classification splits are stratified by class.

Dataset	Sector	Task	Size and features	Main data problems
D1: Telco Customer Churn [33]	Telecommunications	Binary classification (churn)	7,043 rows; 3 numeric and 16 categorical predictors	Mixed feature types; blank TotalCharges; 26.5% churners; tenure and charges on different scales
D2: UCI Bank Marketing [34]	Banking	Binary classification (term deposit)	41,188 rows (6,000 used); 10 numeric and 10 categorical inputs	Many categories (for example, 12 job types); 11.3% positive class; many 'unknown' values
D3: Online Retail II [35]	Retail	Regression (spend) and K-Means clustering	1,067,371 lines combined into 5,878 customers; 6 numeric and 2 categorical features	Very skewed spend; large wholesale orders; lines without customer ID (about 23%)

Table IV: Structure and Data Quality Problems of the Datasets

V. RESULTS AND DISCUSSION

This section first reports representative pipelines (Section V-A). It then presents the marginal effect of each stage as an ablation study and tests hypotheses H1–H5 (Section V-B), compares the findings with earlier studies (Section V-C), reports the statistical tests (Section V-D), and answers the research questions (Section V-E). Run time is reported throughout as the overhead of each stage.

A. Experimental Setting and Representative Results

All results in this section come from running Algorithm 1 end to end on the three real datasets: 162 pipelines for each supervised task (D1 and D2 classification, D3 regression) and 72 pipelines for K-Means on D3. This gave 12,150 supervised model fits and 360 K-Means fits (12,510 in total).

Because of limited computing power (a single-core machine), the protocol was reduced as described in Section III-D: 5-fold cross-validation with one repeat (5 validation splits), Random Forest with 60 trees, XGBoost with 100 trees, SVM and SVR trained on 2,500 training rows, and a 6,000-row sample of D2. The 10% MCAR gaps are part of the data on which the models are trained and tested, so Stage 1 changes the downstream results. RMSE_{imp} comes from the same run, on the hidden validation cells (413 per split on average in D1, 1,065 in D2, and 570 in D3). The Mutual Information ranking uses a random sample of up to 2,500 training rows. For each pipeline, values are the mean ± standard deviation over the 5 validation splits, and the seed is 42. D3 regression errors are on the log scale, because the target is log(1 + total spend).

Pipeline (S1 + S2 + S3 + S4 + S5)	Dataset and estimator	RMSE_{imp}	Primary metric	Secondary metric	Run time t_{exec} (s)
P1 (Baseline): Delete + None + Label + None + None	D1; LR	N/A	F1: 0.559 ± 0.036	AUC: 0.828	0.34
P1 (Baseline): Delete + None + Label + None + None	D1; SVM	N/A	F1: 0.000 ± 0.000	AUC: 0.747	0.16
P2 (Simple): Mean/Mode + None + One-Hot + None + None	D1; RF	0.998	F1: 0.539 ± 0.037	AUC: 0.818	0.35
P3 (Scaled): KNN + None + One-Hot + Z-score + None	D1; RF	0.757	F1: 0.543 ± 0.041	AUC: 0.816	0.79
P3 (Scaled): KNN + None + One-Hot + Z-score + None	D1; SVM	0.757	F1: 0.565 ± 0.032	AUC: 0.799	0.65

Pipeline (S1 + S2 + S3 + S4 + S5)	Dataset and estimator	$RMSE_{imp}$	Primary metric	Secondary metric	Run time t_{exec} (s)
P4 (Target-coded): KNN + IQR + Target + Z-score + None	D2; XGB	0.644	F1: 0.359 ± 0.051	AUC: 0.760	0.84
P4 (Target-coded): KNN + IQR + Target + Z-score + None	D2; LR	0.644	F1: 0.321 ± 0.021	AUC: 0.764	0.77
P5 (PCA): Mean/Mode + IQR + One-Hot + Z-score + PCA	D2; SVM	0.962	F1: 0.142 ± 0.052	AUC: 0.670	0.11
P6 (Mutual Information): KNN + IQR + Target + Min-Max + MI	D2; RF	0.644	F1: 0.341 ± 0.046	AUC: 0.764	1.16
P7 (Unclipped): Mean/Mode + None + One-Hot + None + None	D3; Linear	0.772	RMSE: 0.908 ± 0.090	R^2: 0.566	0.01
P7 (Unclipped): Mean/Mode + None + One-Hot + None + None	D3; SVR	0.772	RMSE: 0.714 ± 0.043	R^2: 0.735	0.14
P8 (Robust): KNN + IQR + Target + Z-score + None	D3; XGB	0.648	RMSE: 0.557 ± 0.026	R^2: 0.838	0.62
P7 (Unclipped): Mean/Mode + None + One-Hot + None + None	D3; K-Means	N/A	Silhouette: 0.686 ± 0.122	N/A	0.03
P9 (Robust, K-Means): KNN + IQR + One-Hot + Z-score + None	D3; K-Means	N/A	Silhouette: 0.180 ± 0.014	N/A	0.56

Table V: Results for Representative Pipelines (Mean ± Sd Over 5 Validation Splits)

Notes: Target encoding needs a target, so the K-Means pipeline P9 uses One-Hot in place of Target. $RMSE_{imp}$ is the imputation error of Eq. (2). It is the same for every pipeline that shares Stage 1, it is not defined for deletion, and it was not recorded for K-Means. D3 RMSE is on the log scale.

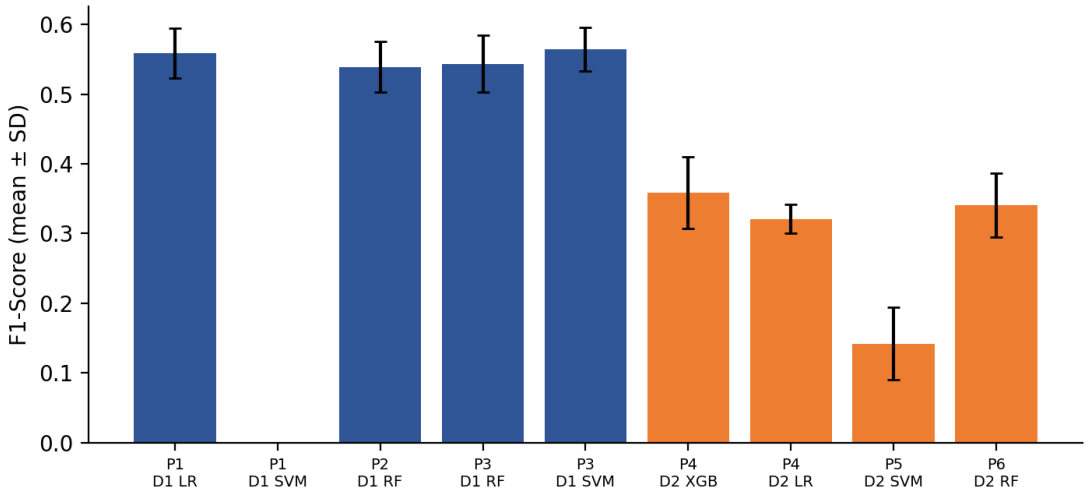

Fig. 3. Mean F1-Score (± SD over 5 splits) of the classification rows of pipelines P1 to P6 in Table V.

Table V and Fig. 3 show three clear patterns. First, an SVM without scaling (P1, D1) predicted the majority class for every validation row, so its F1-Score is 0.000. With KNN imputation and Z-score scaling (P3) the same model reached 0.565 ± 0.032, the highest F1-Score among the D1 rows in Table V. Second, on D2 the F1-Scores are lower for every pipeline (0.142 to 0.359 in Table V) because only 11.3% of the rows are positive and the models use a fixed 0.5 threshold. AUC-ROC (0.670 to 0.764) is the more informative score there. Third, on the D3 regression task the gradient-boosting model in P8 reached an RMSE of 0.557 ± 0.026 (R^2 = 0.838), compared with 0.908 ± 0.090 (R^2 = 0.566) for Linear Regression in P7. For K-Means, the pipeline with clipping and Z-score scaling (P9) reached a Silhouette of 0.180 ± 0.014, while the unclipped, unscaled pipeline (P7) reached 0.686 ± 0.122. This difference is partly caused by how Silhouette is scored (Section V-B).

Averaged over all 162 pipelines, the five estimators rank as follows. On D1 by F1-Score: XGB 0.517, RF 0.505, LR 0.504, DT 0.485, SVM 0.366; by AUC-ROC: LR 0.811, XGB 0.804, RF 0.786, SVM 0.751, DT 0.718. On D2 by F1-Score: XGB 0.306, DT 0.300, RF 0.299, LR 0.261, SVM 0.138; by AUC-ROC: LR 0.762, XGB 0.752, RF 0.739, SVM 0.656, DT 0.634. On D3 regression by R^2: XGB 0.779, RF 0.763, SVR 0.715, DT 0.703, Linear Regression 0.599. The mean Silhouette of K-Means over its 72 pipelines is 0.169. Average run time per fit ranges from 0.21 s (Linear Regression on D3) to 0.68 s (Random Forest on D1).

B. Ablation Study: Effect of Each Stage and Test of the Hypotheses

Because the design is full-factorial, each stage choice can be removed or replaced while the other four stages are held fixed. Tables VI to VIII therefore form an ablation study of the five stages. Table VI gives the marginal effect of each

stage choice: the mean score over every pipeline that contains that choice, so the other four stages are averaged out. For F1-Score, AUC, and Silhouette higher is better; for RMSE lower is better. Run time and feature counts are averaged over the supervised tasks.

Stage: choice	D1 F1	D2 F1	D1 AUC	D2 AUC	D3 RMSE	D3 Silhouette	t_{exec} (s)	Features
1. Missing values: Delete	0.465	0.245	0.767	0.696	0.759	0.181	0.17	14.1
1. Missing values: Mean/Mode	0.479	0.267	0.777	0.714	0.700	0.158	0.21	14.3
1. Missing values: KNN	0.481	0.270	0.779	0.716	0.693	0.168	0.78	14.2
2. Outliers: None	0.475	0.273	0.774	0.713	0.777	0.302	0.39	14.2
2. Outliers: IQR	0.475	0.248	0.774	0.705	0.658	0.036	0.39	14.2
3. Encoding: One-Hot	0.476	0.274	0.775	0.713	0.713	0.209	0.45	23.8
3. Encoding: Label	0.473	0.239	0.773	0.709	0.725	0.129	0.36	9.7
3. Encoding: Target	0.477	0.269	0.775	0.704	0.714	N/A	0.36	9.1
4. Scaling: None	0.352	0.247	0.730	0.703	0.767	0.375	0.38	12.2
4. Scaling: Min-Max	0.538	0.261	0.796	0.712	0.728	−0.034	0.39	15.2
4. Scaling: Z-score	0.535	0.274	0.797	0.712	0.657	0.167	0.39	15.2
5. Reduction: None	0.512	0.277	0.803	0.717	0.657	0.171	0.33	23.1
5. Reduction: PCA	0.398	0.222	0.722	0.696	0.836	0.167	0.37	7.6
5. Reduction: MI	0.515	0.283	0.798	0.714	0.659	N/A	0.47	11.9

Table VI: Marginal Effect of Each Stage Choice (Mean over All Pipelines Containing the Choice)

Silhouette is not available for Target encoding and Mutual Information, because K-Means cannot use them. D3 RMSE is on the log scale.

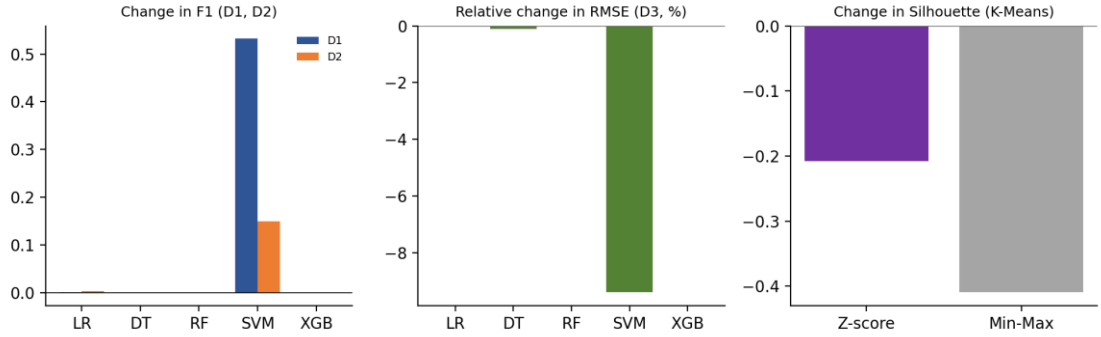

Fig. 4. Effect of Z-score scaling by estimator (compared with no scaling, all other stages equal, no dimensionality reduction): change in F1 on D1 and D2, relative change in RMSE on D3, and change in Silhouette for K-Means.

The five hypotheses were checked with matched comparisons, in which two pipelines differ only in the stage being tested. The outcome is summarized in Table VII.

	Verdict	Evidence (matched comparisons)
H1	Supported when PCA is not used	With no reduction, Z-score scaling changed F1 by −0.000 (DT), −0.001 (RF), and +0.000 (XGB) on D1, by −0.000, −0.000, and +0.000 on D2, and RMSE by −0.001, −0.000, and +0.000 on D3. With PCA, scaling mattered: on D1 F1 rose by +0.272 (DT), +0.151 (RF), and +0.312 (XGB).
H2	Partly supported	SVM gained +0.532 F1 on D1 and +0.149 on D2 from Z-score scaling, and its D3 RMSE fell by 0.073. LR gained almost nothing without PCA (+0.001 on D1, +0.003 on D2) but gained +0.546 on D1 with PCA. K-Means did not benefit: Silhouette fell by 0.208 (Z-score) and 0.409 (Min-Max).
H3	Supported for imputation error and run time; not for F1	$RMSE_{imp}$ for KNN vs Mean/Mode: 0.757 vs 0.998 (D1), 0.644 vs 0.962 (D2), and 0.648 vs 0.772 (D3), which is 24%, 33%, and 16% lower. KNN added 0.46 s (D1), 0.74 s (D2), and 0.50 s (D3) of preprocessing time per fit. The matched F1 gain was only +0.002 (D1) and +0.003 (D2).
H4	Supported for regression; not for classification or K-Means	On D3, IQR clipping lowered RMSE for every model: −0.343 (LR, from 1.010), −0.124 (SVR), −0.062 (RF), −0.037 (XGB), and −0.031 (DT). On D1 its F1 effect was within 0.001. On D2 it lowered F1 for LR (−0.034) and SVM (−0.083). It lowered the K-Means Silhouette from 0.302 to 0.036.
H5	Partly supported	Target encoding gave a much smaller matrix: 19 vs 46 features (D1), 19 vs 54.5 (D2), and 7 vs 17 (D3). On D2 its F1 effect against One-Hot was +0.008 (DT), −0.027 (RF), and +0.019 (XGB), so it matched but did not consistently beat One-Hot for trees.

Table VII: Outcome of the Hypotheses

Two observations help to explain these outcomes. First, scaling and PCA interact. PCA uses variance, so features with large units dominate the components when they are not scaled. The worst pipelines combine PCA with no scaling: for example, Mean/Mode + IQR + One-Hot + None + PCA has a mean rank of 141.0 out of 162. Second, the K-Means result follows from how clusters are scored. Silhouette is measured in one common Z-scored space (Section III-B). Clusterings fitted on unscaled features are driven by the large Monetary and Frequency values and therefore agree best with this space, while clipping removes the very large customers who form separate clusters. The K-Means findings therefore describe agreement with this common space and not an absolute cluster quality.

Fig. 5 shows the trade-off between predictive quality and run time. KNN imputation moves the points to the right: it adds 0.57 s of preprocessing per fit on average, so a full fit takes 0.78 s, 3.7 times the Mean/Mode run time of 0.21 s, for a very small gain in F1. PCA lowers F1 without saving run time, because the PCA step itself adds cost at this data size.

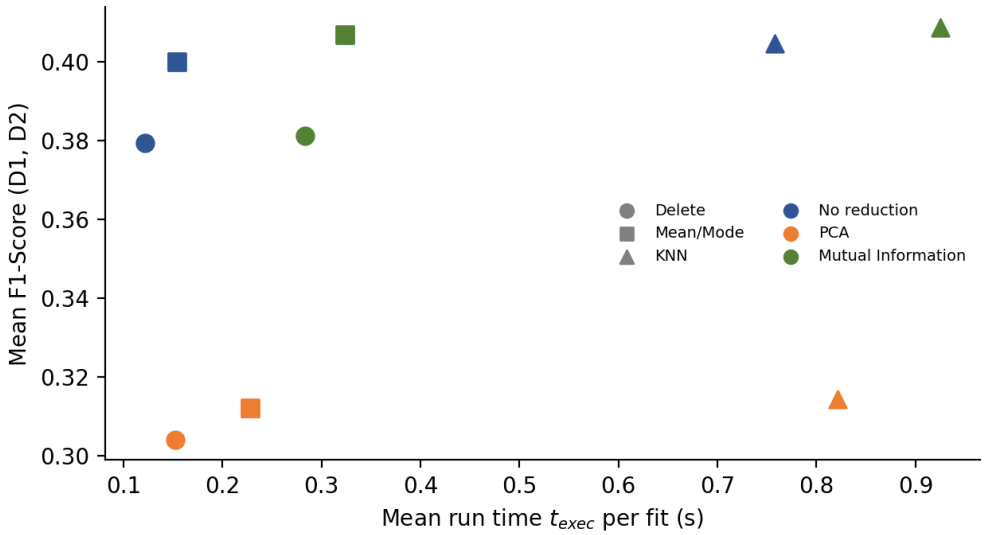

Fig. 5. Mean F1-Score against mean run time for each imputation method (marker) and reduction choice (colour), on D1 and D2.

Deletion needs a caution. On D2 it keeps only 1,398 of 4,800 training rows (29%), and on D1 it keeps 4,138 of 5,634 (73%). In matched comparisons, deletion lowered F1 by 0.014 (D1) and 0.022 (D2), and AUC-ROC by 0.010 and 0.019, compared with Mean/Mode imputation. Because F1 at a fixed threshold can move for reasons other than ranking quality, AUC-ROC should be read together with F1 on imbalanced data.

C. Comparison with Earlier Studies

Several findings agree with earlier work. De Amorim et al. [11] and Ahsan et al. [15] reported that the scaler affects results. Here Z-score scaling gave a significant gain over no scaling (mean ΔF1 = +0.105 on D1 and D2), the SVM gained most, and tree models without PCA were unchanged. The difference between Min-Max and Z-score was small (ΔF1 = −0.005) but significant (Table VIII). Jäger et al. [10] judged imputation both by its error and by the quality of the models trained afterward. The same gap appears here: KNN lowered RMSE_{imp} by 16% to 33%, but the gain in F1 (+0.003) was not significant after Holm correction. Alshdaifat et al. [19] found differences that vary by algorithm, and Johnson et al. [20] found no single best encoder. The encoder effects in this study are also model-dependent. Target encoding helped XGB (+0.013 on D1, +0.019 on D2) but hurt SVM (−0.019 on D1) and LR (−0.024 on D2), and Label encoding helped DT (+0.016 on D2) but hurt SVM (−0.117 on D2). One-Hot encoding was the most consistent choice: all five pipelines with the best mean ranks use it. This is in line with the One-Hot result reported for medical data by Chuang et al. [9], although that study used different data and resampling.

The insensitivity of tree ensembles to scaling also explains why random forest and gradient boosting are popular in applied studies with raw, unscaled inputs, such as EEG feature extraction [17] and building energy forecasting [31]. The strong scaling effect for the SVM matches its role as a distance-based classifier in applied work such as video activity recognition [16].

One finding differs. Pargent et al. [12] found that regularized target encoding worked best. Here it only matched One-Hot (mean ΔF1 = −0.002, Holm-adjusted p = 1.000), although it used 59% to 65% fewer features and less run time (0.36 s against 0.45 s per fit). A likely reason is that the categorical features here have few levels (at most 4 in D1, 11 in D2, and 7 in D3), where the main benefit of target encoding, a compact code for high-cardinality features, is small. This reason was not tested. Studies [6], [10], [11], [15], [12], [13] each examine one stage, so they cannot be compared with the interaction between scaling and PCA, or with the effect of clipping on D3, that this design reveals.

D. Statistical Reporting

The Friedman test was applied to the supervised tasks with k = 162 pipelines and N_s = 75 blocks (3 datasets × 5 estimators × 5 splits). It rejected the hypothesis that all pipelines perform alike: χ_F^2 = 4795.05 (df = 161), p < 0.001, with Kendall's W = 0.397. The test also rejected equality within each dataset: D1 χ_F^2 = 1948.23 (W = 0.484), D2 χ_F^2 = 1589.84

(W = 0.395), and D3 regression χ_F^2 = 2217.00 (W = 0.551), all with p < 0.001. For the 72 K-Means pipelines (N_s = 5) the result was χ_F^2 = 344.58 (df = 71), p < 0.001, W = 0.971. Agreement between blocks is therefore strongest for D3 and K-Means and weakest for D2, where F1 is low for all pipelines. All five pipelines with the best mean ranks use KNN imputation and One-Hot encoding. The pipeline with the best mean rank (38.3 of 162) was KNN + IQR + One-Hot + Min-Max + None, followed by KNN + IQR + One-Hot + Z-score + None (39.1).

Pairwise Wilcoxon signed-rank tests with Holm correction then compared selected stage choices (Table VIII). To make blocks comparable across tasks measured in different units, the scores in each block (F1, negative RMSE) were standardized within the block, and each test uses the mean difference over matched pipelines in each block. Parametric tests such as ANOVA were avoided because performance scores are usually not normally distributed.

Comparison (A vs B)	Median Δz	A better	W	p	Holm p	Mean ΔF1 (D1, D2)	Significant
KNN vs Mean/Mode imputation	+0.034	69%	985	0.020	0.081	+0.003	No
Mean/Mode vs deletion	+0.237	89%	66	< 0.001	< 0.001	+0.018	Yes
IQR clipping vs none	+0.003	59%	1106	0.092	0.276	−0.012	No
Target vs One-Hot encoding	−0.004	47%	1375	0.792	1.000	−0.002	No
Label vs One-Hot encoding	−0.081	27%	545	< 0.001	< 0.001	−0.019	Yes
Z-score vs no scaling	+0.727	92%	38	< 0.001	< 0.001	+0.105	Yes
Min-Max vs Z-score scaling	−0.136	24%	298	< 0.001	< 0.001	−0.005	Yes
PCA vs no reduction	−1.230	5%	35	< 0.001	< 0.001	−0.085	Yes
Mutual Information vs no reduction	−0.009	47%	1315	0.561	1.000	+0.004	No

Table VIII: Wilcoxon Signed-Rank Tests With Holm Correction (A = 0.05, 75 Supervised Blocks)

Five choices differ significantly from their reference after Holm correction. Mean/Mode imputation is better than deletion, Z-score scaling is better than no scaling, and Z-score is slightly better than Min-Max. Label encoding (compared with One-Hot) and PCA (compared with no reduction) are worse. KNN versus Mean/Mode imputation (raw p = 0.020, Holm p = 0.081), IQR clipping, Target versus One-Hot encoding, and Mutual Information versus no reduction show no significant difference. The blocks share training data across splits, so the p-values are probably somewhat too small (see Limitations).

E. Answers to the Research Questions and Limitations

RQ1 (Stage interaction): The stages do not act independently. The effect of scaling depends on the reduction stage: without PCA, scaling mattered only for the SVM, while with PCA it raised D1 F1 by +0.151 to +0.546 for every estimator. The effect of clipping depends on the task: it made no practical difference on D1 (F1 change within 0.001), lowered F1 for LR and SVM on D2, and lowered the error of every D3 regression model. The best pipeline overall by mean rank was KNN + IQR + One-Hot + Min-Max + None (38.3). The best mean F1 over the five models was 0.561 on D1 (KNN + IQR + One-Hot + Z-score + None) and 0.320 on D2 (KNN + IQR + One-Hot + Min-Max + None), and the lowest mean RMSE on D3 was 0.596 (KNN + IQR + One-Hot + Z-score + None).

RQ2 (Model family): Yes. Tree models (DT, RF, XGB) were unaffected by scaling when no PCA was used (ΔF1 within 0.001 on D1 and D2). Distance- and margin-based models reacted strongly: the SVM gained +0.532 F1 on D1 after scaling. Logistic regression was insensitive to scaling without PCA at the 0.5 threshold. Clipping helped the linear model most on D3 (RMSE from 1.010 to about 0.67) and helped the tree models least. K-Means scored lower after scaling or clipping under the common-space Silhouette.

RQ3 (Accuracy and time): KNN imputation lowered imputation error by 16% to 33% but gave no significant F1 gain (+0.002 on D1, +0.003 on D2) and raised the mean run time from 0.21 s to 0.78 s per fit (3.7 times). Target encoding cut the feature count by 59% to 65%, used less time than One-Hot, and changed F1 by only −0.002 (not significant). PCA lowered F1 by 0.085 and did not save time (0.37 s against 0.33 s). Mutual Information halved the feature count (11.9 against 23.1) with no significant loss of F1, but added 0.14 s per fit. In this experiment, a good balance was Mean/Mode imputation, One-Hot or Target encoding, and Z-score scaling without reduction; KNN imputation is worth its extra time only when imputation accuracy itself matters.

RQ4 (Significance): Yes. The Friedman test rejected equality of the pipelines for every task (p < 0.001; Section V-D). Holm-corrected Wilcoxon tests show significant effects for scaling, PCA, Label encoding, and deletion, and no significant effect for KNN versus Mean/Mode imputation, IQR clipping, Target versus One-Hot encoding, or Mutual Information selection.

Limitations.

1. The protocol was reduced to fit the available computing power: 5 splits instead of 10, Random Forest with 60 trees, XGBoost with 100 trees, SVM and SVR trained on 2,500 rows, and a 6,000-row sample of D2. Results for the full D2 file, with larger models, may differ.
2. The best pipeline for a dataset is the maximum over 162 pipelines on the same 5 splits, so its score is somewhat optimistic.
3. F1-Score at a fixed 0.5 threshold is low for the imbalanced D2, so differences on D2 are small compared with their spread.
4. Silhouette is scored in one common Z-scored space, which favours clusterings fitted on unscaled data.
5. Only MCAR missingness is injected. Real business data may be MAR or MNAR.
6. Hyperparameters are fixed. Tuning each pipeline could change the rankings.
7. Only static tabular data is used. Time series are not covered.
8. Cross-validation folds share training data, so p-values may look slightly more certain than they are.
9. D3 is aggregated to the customer level, so its missing values come only from the injected MCAR gaps, and its regression target is on the log scale.
10. Only three datasets are used, so the results may not hold for every business domain.

VI. CONCLUSION

This paper presented a leakage-free factorial experiment that compares five preprocessing stages (missing value imputation, outlier treatment, categorical encoding, feature scaling, and dimensionality reduction) on business analytics tasks. The design tested 162 pipelines on supervised tasks and 72 on clustering, using three public datasets (Telco Customer Churn, UCI Bank Marketing, and Online Retail II) and six algorithms. Scaling and dimensionality reduction had the largest effects. Z-score scaling raised F1 by +0.105 on average and was essential for the SVM, while tree models were unaffected unless PCA was used; PCA lowered F1 by 0.085. Deletion was worse than Mean/Mode imputation, and Label encoding was worse than One-Hot. KNN imputation lowered imputation error by 16% to 33%, but its gain in F1 was not significant and it increased run time 3.7 times. IQR clipping helped the regression task on skewed customer data but not the classification tasks.

The main contributions are: (1) a cross-validation protocol in which every preprocessing step, including listwise deletion and target encoding, is fitted on the training fold only; (2) a joint test of five stages instead of one stage at a time, on real business data; and (3) practical rules for choosing a pipeline, shown in Table IX. The results come from a reduced protocol (Section V-A), so they should be confirmed with larger models and the full D2 file. Future work will cover MAR and MNAR missingness, time-series data such as stock prices, where LSTM models need careful normalisation [37], privacy-preserving federated settings in which preprocessing statistics cannot be pooled [14], context-aware hybrid models [38], and automatic pipeline search with genetic algorithms [39] or other metaheuristics such as Coyote and Badger optimisation [40], so that data-driven decision models [41], including remote-sensing climate analysis [42], can be built on reliable preprocessing.

Scenario	Estimator family	Suggested pipeline (S1 + S2 + S3 + S4 + S5)	Evidence from this study
Churn or credit classification, mixed types	RF, XGBoost, LR	Mean/Mode (or KNN) + None + One-Hot (or Target) + Z-score + None	Best mean ranks use One-Hot and no reduction; scaling is harmless for trees and needed for margin models
Imbalanced campaign data with many categories	XGBoost, RF, LR	Mean/Mode or KNN + None or IQR + Target + Z-score + None or Mutual Information (50%)	Target encoding used 65% fewer features than One-Hot with no significant F1 loss (Table VIII); Mutual Information matched no reduction (ΔF1 = +0.004, not significant); P4 with XGB reached F1 0.359 (Table V)
SVM, kernel, or linear models	SVM, LR	Mean/Mode or KNN + None + One-Hot + Z-score + None	Scaling raised SVM F1 by +0.532 on D1; PCA without scaling gave the worst ranks
Skewed spend regression and segmentation	Linear Regression, SVR, XGB	Mean/Mode or KNN + IQR + One-Hot + Z-score + None	IQR clipping lowered D3 RMSE for all five models (Linear Regression from 1.010 to about 0.67)
Any task	All	Avoid deletion, Label encoding for linear and margin models, and PCA	Significant losses in Table VIII

Table IX: Pipeline Guidelines Supported By the Results

References

1. H. Wu, “A brief introduction to data preprocessing,” in Proc. 2nd Int. Conf. Engineering Management, Information Technology and Intelligence (EMITI), SciTePress, 2025, pp. 216–221, doi: 10.5220/0014325400004718.
2. B. Arivazhagan, S. Pandikumar, S. B. Sethupandian, and R. S. Subramanian, “Pattern discovery and analysis of customer buying behavior using association rules mining algorithm in e-commerce,” in Proc. Int. Conf. Electrical, Electronics, Information and Communication Technologies (ICEEICT), Trichy, India, 2022, pp. 1–5, doi: 10.1109/ICEEICT53079.2022.9768473.

3. C. Jamunadevi, S. Pandikumar, N. N, S. Prasath, and D. S, "Enhancing cybersecurity using machine learning strategies for phishing defense," in Proc. 9th Int. Conf. Communication and Electronics Systems (ICCES), Coimbatore, India, 2024, pp. 801–808, doi: 10.1109/ICCES63552.2024.10859402.
4. K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," Global Transitions Proceedings, vol. 3, no. 1, pp. 91–99, 2022, doi: 10.1016/j.gltp.2022.04.020.
5. T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," Journal of Big Data, vol. 8, art. 140, 2021, doi: 10.1186/s40537-021-00516-9.
6. M. K. Hasan et al., "Missing value imputation affects the performance of machine learning: A review and analysis of the literature (2010–2021)," Informatics in Medicine Unlocked, vol. 27, art. 100799, 2021, doi: 10.1016/j.imu.2021.100799.
7. A. Boukerche, L. Zheng, and O. Alfandi, "Outlier detection: Methods, models, and classification," ACM Computing Surveys, vol. 53, no. 3, art. 55, 2020, doi: 10.1145/3381028.
8. C. Jamunadevi, S. Prasath, V. E. Sathishkumar, and S. Pandikumar, "Enhancing loan approval prediction through advanced machine learning models," in Proc. 2nd Int. Conf. Sustainable Computing and Smart Systems (ICSCSS), 2024.
9. C.-W. Chuang, C.-K. Wu, C.-H. Wu, B.-C. Shia, and M. Chen, "Enhancing model generalizability in medical artificial intelligence: Systematic comparison of categorical encoding and sampling techniques for imbalanced data," JMIR Medical Informatics, vol. 14, art. e75655, 2026, doi: 10.2196/75655.
10. S. Jäger, A. Allhorn, and F. Bießmann, "A benchmark for data imputation methods," Frontiers in Big Data, vol. 4, art. 693674, 2021, doi: 10.3389/fdata.2021.693674.
11. L. B. V. de Amorim, G. D. C. Cavalcanti, and R. M. O. Cruz, "The choice of scaling technique matters for classification performance," Applied Soft Computing, vol. 133, art. 109924, 2023, doi: 10.1016/j.asoc.2022.109924.
12. F. Pargent, F. Pfisterer, J. Thomas, and B. Bischl, "Regularized target encoding outperforms traditional methods in supervised machine learning with high cardinality features," Computational Statistics, vol. 37, no. 5, pp. 2671–2692, 2022, doi: 10.1007/s00180-022-01207-6.
13. L. Yu, R. Zhou, R. Chen, and K. K. Lai, "Missing data preprocessing in credit classification: One-hot encoding or imputation?," Emerging Markets Finance and Trade, vol. 58, no. 2, pp. 472–482, 2022, doi: 10.1080/1540496X.2020.1825935.
14. S. Pandikumar, V. Pandimurugan, A. Rama Prasath, S. Karattil, S. C. B. Jaganathan, and P. Vigneshwaran, "A novel framework for differential privacy based federated continual learning for dynamic medical data analysis," SN Computer Science, vol. 7, no. 6, art. 664, 2026, doi: 10.1007/s42979-026-05245-5.
15. M. M. Ahsan, M. A. P. Mahmud, P. K. Saha, K. D. Gupta, and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," Technologies, vol. 9, no. 3, art. 52, 2021, doi: 10.3390/technologies9030052.
16. S. Manjula and K. Lakshmi, "Human abnormal activity pattern analysis in diverse background surveillance videos using SVM and ResNet50 model," in IoT and Analytics for Sensor Networks (Proc. ICWSNUCA 2021), Springer, 2022, pp. 47–60.
17. C. Jamunadevi, P. Ragupathy, P. Sritha, S. Pandikumar, and S. Deepa, "Performance analysis of random forest classifier in extracting features from the EEG signal," in Proc. Int. Conf. Advanced Computing Technologies and Applications (ICACTA), Coimbatore, India, 2022, pp. 1–5, doi: 10.1109/ICACTA54488.2022.9753364.
18. J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," Journal of Big Data, vol. 7, art. 28, 2020, doi: 10.1186/s40537-020-00305-w.
19. E. Alshdaifat, D. Alshdaifat, A. Alsarhan, F. Hussein, and S. M. F. S. El-Salhi, "The effect of preprocessing techniques, applied to numeric features, on classification algorithms' performance," Data, vol. 6, no. 2, art. 11, 2021, doi: 10.3390/data6020011.
20. T. Johnson, A. J. Liu, S. Raza, and A. McGuire, "A comparison of modeling preprocessing techniques," arXiv preprint arXiv:2302.12042, 2023.
21. R. Zebari et al., "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," Journal of Applied Science and Technology Trends, vol. 1, 2020, doi: 10.38094/jastt1224.
22. P. Ray, S. S. Reddy, and T. Banerjee, "Various dimension reduction techniques for high dimensional data analysis: A review," Artificial Intelligence Review, vol. 54, 2021, doi: 10.1007/s10462-020-09928-0.
23. S. Pandikumar, A. Bostani, D. Sundaranarayana, I. Fakhriddin, D. Alimjon, U. Ilkhom, and A. Juraboyev, "AI-driven nonlinear optimization for knowledge extraction and pattern analysis in IoT-enabled data mining systems," Results in Nonlinear Analysis, vol. 8, no. 3, pp. 36–46, 2025, doi: 10.31838/rna/2025.08.03.005.
24. K. R. Shahapure and C. Nicholas, "Cluster quality analysis using silhouette score," in Proc. IEEE 7th Int. Conf. Data Science and Advanced Analytics (DSAA), 2020, pp. 747–748, doi: 10.1109/DSAA49011.2020.00096.
25. J. Carrasco, S. García, M. M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," Swarm and Evolutionary Computation, vol. 54, art. 100665, 2020, doi: 10.1016/j.swevo.2020.100665.
26. L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022.
27. scikit-learn developers, "scikit-learn: Machine learning in Python, user guide." [Online]. Available: https://scikit-learn.org/stable/user_guide.html. Accessed: Nov. 5, 2026.
28. XGBoost developers, "XGBoost documentation." [Online]. Available: https://xgboost.readthedocs.io. Accessed: Nov. 5, 2026.
29. A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," Information Sciences, vol. 622, pp. 178–210, 2023, doi: 10.1016/j.ins.2022.11.139.
30. S. Pandikumar, H. R. Jakaraddi, C. Menaka, and A. R. Nambiar, "DermAI: A triple-CNN ensemble learning approach for early detection of skin tone abnormalities in clinical settings," in Proc. Int. Conf. Sustainable Communication Networks and Application (ICSCNA), Theni, India, 2024, pp. 1621–1626, doi: 10.1109/ICSCNA63714.2024.10864008.
31. C. Jamunadevi, N. Rajasekaran, S. Pandikumar, R. Raja, and K. P. Narendra Prasanth, "Energy performance forecasting in buildings using machine learning," in Proc. Int. Conf. Communication, Computing and Internet of Things (IC3IoT), Chennai, India, 2024, pp. 1–6, doi: 10.1109/IC3IoT60841.2024.10550233.
32. S. Manjula, A. Bostani, A. Srinivasan, R. Sindoori, R. Madinakhon, B. Bozorov, and U. Amirbek, "Nonlinear clustering optimization for scalable data mining in cloud and quantum computing environments," Results in Nonlinear Analysis, vol. 8, no. 3, 2025.
33. IBM, "Telco customer churn," IBM sample data set, distributed via Kaggle. [Online]. Available:

https://www.kaggle.com/datasets/blastchar/telco-customer-churn. Accessed: Nov. 5, 2026.
34. S. Moro, P. Rita, and P. Cortez, "Bank Marketing," UCI Machine Learning Repository. [Online]. Available: https://archive.ics.uci.edu/dataset/222/bank+marketing. Accessed: Nov. 5, 2026.
35. D. Chen, "Online Retail II," UCI Machine Learning Repository, 2019, doi: 10.24432/C5CG6D. [Online]. Available: https://archive.ics.uci.edu/dataset/502/online+retail+ii. Accessed: Nov. 5, 2026.
36. S. Bates, T. Hastie, and R. Tibshirani, "Cross-validation: What does it estimate and how well does it do it?," Journal of the American Statistical Association, vol. 119, no. 546, pp. 1434–1445, 2024, doi: 10.1080/01621459.2023.2197686.
37. S. Pandikumar, S. B. Sethupandian, M. S. Saravanan, S. N. Prasad, and M. Arun, "Deep learning based long short-term memory recurrent neural network for stock price movement prediction," Indian J. Science and Technology, vol. 15, no. 11, pp. 474–480, 2022, doi: 10.17485/IJST/v15i11.27.
38. P. Savarimalai, M. Chellappan, M. S. Nidhya, M. M. Thomas, S. Nallathambi, and M. Selvaraj, "A context-aware intelligent scheduling framework for modern operating systems using hybrid machine learning models," J. Computational and Cognitive Engineering, vol. 5, no. 2, pp. 321–332, 2026, doi: 10.47852/bonviewJCCE62026664.
39. S. Pandikumar, B. Sapaev, K. Jafarov, J. Saidmuradova, G. Bekhruz, S. Elmurodov, and S. Muratova, "An artificial intelligence and genetic algorithm optimized biosensor system for real-time detection of environmental toxins and their biomedical impact," Natural and Engineering Sciences, vol. 10, no. 3, pp. 561–574, 2025, doi: 10.28978/nesciences.1811144.
40. S. Manjula, M. Sharmila Begum, and P. Ilangovan, "HARCE: Human activity recognition in complex environments: an improved YOLOv8s model with Coyote and Badger Optimization for enhanced detection accuracy," Int. J. Syst. Assur. Eng. Manag., 2026.
41. T. Megala, K. Srinivasan, K. Sathishkumar, M. Ramalingam, S. Pandikumar, A. Mummoorthy, and R. Naveenkumar, "A systematic investigation of climate change impacts and disaster risk reduction strategies using data-driven analytical models," Int. J. Artificial Intelligence and Machine Learning, vol. 6, no. 5s, pp. 252–262, 2026.
42. M. Sailaja, M. Prema Kumar, B. Swarna Jyothi, G. L. Narasamba Vanguri, S. Manjula, and D. Divya Priya, "Remote sensing–based UAV imaging in heat pattern analysis impact on climate change detection using fuzzy stacked lasso elastic-net model," Remote Sens. Earth Syst. Sci., vol. 7, no. 4, pp. 699–708, 2024, doi: 10.1007/s41976-024-00158-4.