Balanced Incomplete Block Design (BIBD) For Machine Learning Methods

Chaoan Li[*] Zhaoyu Wang[†]

April 15, 2025

Abstract

Comparing machine learning algorithms across diverse datasets is challenging due to variations in data characteristics. Traditional evaluation methods often produce inconsistent results. This study adopts a Balanced Incomplete Block Design (BIBD) to efficiently compare five classical classifiers—Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), k-Nearest Neighbors (k-NN), and Linear Discriminant Analysis (LDA)—while reducing experimental cost. Each dataset is treated as a block, and performance is assessed using classification accuracy and computational time, analyzed via ANOVA with residual diagnostics and Box–Cox transformation. Results show that while accuracy is comparable across models, LDA and LR are significantly more efficient than SVM and k-NN. We also highlight BIBD's limitation in potentially missing critical pairwise differences due to its incomplete structure.

Keywords: Balanced Incomplete Block Design; Machine Learning; Residual Analysis; Box–Cox Transformation

1 Introduction

As highlighted by Fernández-Delgado et al. [2014], comparing multiple machine learning algorithms across various datasets is both common and challenging, owing to intrinsic differences in data attributes such as dimensionality, noise, and class balance. Traditional evaluation strategies that rely on a limited set of fixed datasets may produce conflicting results, complicating the identification of the most effective algorithm.

To address these issues, we adopt a Balanced Incomplete Block Design (BIBD) from Wu and Hamada [2011] for a systematic comparison of five classification algorithms: Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), k-Nearest

[*] Email: chaoan@tamu.edu

[†] Email: wzy2000426@tamu.edu

Neighbors (k-NN), and Linear Discriminant Analysis (LDA). The BIBD framework enables us to minimize the total number of model-dataset evaluations while ensuring that every pair of models is compared uniformly, thus providing statistical balance.

Our experimental datasets span both binary and multiclass classification tasks with varying feature dimensions and sample sizes. Each dataset is treated as a block, with three models selected per block according to the BIBD structure. For every model-dataset pair, we record classification accuracy as well as computational times.

The recorded performance metrics are analyzed using analysis of variance (ANOVA) and Tukey's multiple comparison procedure within the BIBD framework. Importantly, we also perform residual analysis to assess whether a Box–Cox transformation of the response variable is needed to stabilize the variance and enhance model fit. This step informs the adequacy of the linear modeling approach and ensures robust inference. Although BIBD improves resource efficiency, we also highlight its potential to miss performance differences when certain model comparisons are not directly observed.

The full implementation and reproducibility code is available at: `https://github.com/ChaoanLi/BIBD4ML`.

2 Experimental Design

In experimental comparisons, a common approach is the Randomized Block Design (RBD) Wu and Hamada [2011], where all treatments are applied within each block to control for block-to-block variability. However, when the number of treatments is large or resources are limited, RBD becomes impractical due to the rapid growth in the total number of runs. In our case, evaluating all five machine learning methods on n sets of datasets would result in a total of $5n$ runs, which may be computationally expensive and redundant.

To address this, we employ a Balanced Incomplete Block Design (BIBD), which allows each block (dataset) to include only a subset of treatments while preserving balance and comparability across all treatment pairs. BIBD provides a statistically efficient way to reduce the number of experiments while ensuring that every pair of methods is evaluated together the same number of times. This structure maintains the integrity of the comparisons while significantly lowering the computational cost.

To design an efficient and balanced experimental structure, we implemented a BIBD tailored to the requirements of our study. The goal is to compare $t = 5$ machine learning methods (treatments) using b blocks (datasets), such that each block includes $k = 3$ treatments. Suppose each treatment is replicated r times and each pair of treatments appear in λ blocks. By the basic relations of parameters we have,

$$\begin{aligned} bk = rt \quad &\Rightarrow \quad 3b = 5r, \\ r(k-1) = \lambda(t-1) \quad &\Rightarrow \quad 2r = 4\lambda. \end{aligned}$$

From above, the minimum parameters of this design is $\lambda = 3, r = 6, b = 10$. Table 1 presents the incidence matrix representation of the Balanced Incomplete Block Design (BIBD) used in this project. In this 5×10 binary matrix, each row corresponds to one of the five machine learning methods (M_1 to M_5), and each column corresponds to one of the ten datasets (D_1 to D_{10}). A value of 1 at position (i, j) indicates that method M_i was applied to dataset D_j, while a 0 indicates that it was not.

Table 1: Incidence matrix of BIBD: 5 methods across 10 datasets

Method	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
M_1	1	1	1	1	1	1	0	0	0	0
M_2	1	0	1	0	1	0	1	1	1	0
M_3	1	0	0	1	0	1	1	0	1	1
M_4	0	1	1	0	0	1	1	1	0	1
M_5	0	1	0	1	1	0	0	1	1	1

One can justified it is a valid BIBD design with $t = 5, k = 3, \lambda = 3, r = 6, b = 10$ and achieves the minimum number of runs necessary to ensure pairwise comparison across all model combinations. This structure significantly reduces the computational burden compared to a RBD, which would require $5 \times 10 = 50$ model-dataset evaluations.

While the overall assignment of methods to datasets strictly follows the BIBD structure shown in Table 1, randomization is incorporated at multiple levels to eliminate potential biases. First, the labeling of both the five machine learning methods (M_1 to M_5) and the ten datasets (D_1 to D_{10}) is arbitrary and subject to random permutation. This ensures that no particular method or dataset receives preferential positioning in the analysis. Second, within each block, the order in which the three assigned methods are evaluated is also randomized. Together, these randomizations help reduce the influence of any systematic ordering effects or confounding variables related to data preprocessing, model training, or evaluation sequence. This randomized assignment is summarized in Tables 2 and 3, which display the permuted labels used throughout the experiment for machine learning methods and datasets, respectively.

Table 2: Randomized Method Labels

Order	3	1	5	4	2
Method	Decision Tree	SVM	Logistic Regression	kNN	LDA
Label	M4	M1	M2	M5	M3

In summary, our BIBD provides a rigorous and efficient foundation for the comparative analysis of machine learning classifiers, allowing us to draw meaningful conclusions about their relative performance while maintaining the statistical properties required for valid inference.

Table 3: Randomized Dataset Labels

Order	7	4	2	1	10	6	8	5	3	9
Dataset	Iris	Wine	Wine Quality	Adult Census Income	Bank Marketing	Breast Cancer Wisconsin	Mushroom	Car Evaluation	Heart Disease	Spambase
Label	D1	D8	D5	D4	D6	D10	D3	D9	D7	D2

3 Methodology

This section outlines the preprocessing steps, dataset characteristics, model implementation, and evaluation strategies used in our experiment.

3.1 Data Preprocessing

Each dataset is preprocessed as follows:

1. **Handling missing values:** Missing values are handled based on the nature and distribution of the data. In cases where any row contains a missing value, drop the entire row. If any column has 25% or more missing values, drop the entire column..

2. **Feature scaling:** To ensure that features contribute equally to the model, standardization is applied using `StandardScaler`. This method transforms each feature to have zero mean and unit variance:

$$x' = \frac{x - \mu}{\sigma}$$

   where μ is the mean and σ is the standard deviation of the feature. This is particularly important for models like SVM and k-NN that are sensitive to the scale of input features.

3. **Categorical encoding:** To convert these categorical variables into a numerical format suitable for machine learning models, One-Hot Encoding was applied. This technique creates binary (0 or 1) columns for each category in every feature, preserving all information without imposing any ordinal relationship among categories.

After encoding, the feature set expanded significantly, increasing the dimensionality but enabling the models to learn from categorical data effectively.

Additionally, the target variable (class) was label-encoded using LabelEncoder to convert class labels into numeric format for model training.

4. **Train-Test Split:** Each dataset was randomly split into training and testing sets using a 70:30 ratio to evaluate generalization performance. The training set size was determined by rounding 70% of the total sample size to the nearest integer, with the remaining samples used for testing.

5. **Cross-Validation:** For robust evaluation, k-fold cross-validation ($k = 5$) is applied to ensure stable performance metrics. For the output results, we select the best results of cross validation to compare their best performance.

3.2 Datasets

The five datasets used in this study are summarized in Table 4. They include both binary and multi-class classification tasks with varying sizes and feature dimensions. The data can be found on: https://archive.ics.uci.edu/datasets

3.3 Machine Learning Models

Five classical machine learning models were implemented and tested in this experiment: Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), k-Nearest Neighbors (k-NN), and Linear Discriminant Analysis (LDA). A brief description of each is as follows.

Decision Tree (DT) splits data based on the feature yielding the highest information gain. The Gini impurity is defined as $G = 1 - \sum_{i=1}^{C} p_i^2$, and information gain is calculated by $IG = H(\text{parent}) - \sum_k \frac{N_k}{N} H(k)$.

Support Vector Machine (SVM) finds a hyperplane maximizing the margin between classes. For the linear case, it solves:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{s.t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1.$$

Non-linear boundaries use kernel functions $K(x_i, x_j)$ to replace dot products.

Logistic Regression (LR) models the probability using the sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad z = \mathbf{w}^\top \mathbf{x} + b.$$

k-Nearest Neighbors (k-NN) classifies a sample by the majority vote among its k nearest neighbors, using Euclidean distance: $d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{l=1}^{n} (x_{il} - x_{jl})^2}$.

Table 4: Summary of the datasets used in the experiment

Dataset	**Type**	**Size**	**Description**	**Link**
Iris	3-classes	$(150, 4)$	Classify iris species based on sepal and petal measurements.	`https://archive.ics.uci.edu/ml/datasets/iris`
Wine	3-classes	$(178, 13)$	Classify wine samples into three varieties using chemical analysis.	`https://archive.ics.uci.edu/ml/datasets/wine`
Wine Quality	7-classes	$(6497, 11)$	Predict wine quality based on physicochemical attributes.	`https://archive.ics.uci.edu/ml/datasets/wine+quality`
Adult Income	Binary	$(48842, 14)$	Predict whether income exceeds $50K based on census features.	`https://archive.ics.uci.edu/ml/datasets/adult`
Bank Marketing	Binary	$(45211, 17)$	Predict if a client will subscribe to a term deposit from banking data.	`https://archive.ics.uci.edu/ml/datasets/bank+marketing`
Breast Cancer	Binary	$(569, 30)$	Classify whether tumors are malignant or benign.	`https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic`
Mushroom	Binary	$(8124, 22)$	Predict whether mushrooms are edible or poisonous based on their features.	`https://archive.ics.uci.edu/dataset/73/mushroom`
Car Evaluation	4-classes	$(1728, 6)$	Evaluate the acceptability of a car based on various features.	`https://archive.ics.uci.edu/dataset/19/car+evaluation`
Heart Disease	Binary	$(303, 13)$	Predict the presence of heart disease.	`https://archive.ics.uci.edu/dataset/45/heart+disease`
Spambase	Binary	$(4601, 57)$	Classify emails as spam or not spam based on word frequencies.	`https://archive.ics.uci.edu/dataset/94/spambase`

Linear Discriminant Analysis (LDA) projects data to maximize class separability using scatter matrices:

$$S_W = \sum_{i=1}^{C} \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T, \quad S_B = \sum_{i=1}^{C} N_i(\mu_i - \mu)(\mu_i - \mu)^T.$$

For two classes, the optimal projection is $\mathbf{w} = S_W^{-1}(\mu_1 - \mu_2)$.

3.4 Model Implementation

The following steps were followed in implementing and evaluating the models:

1. Implemented five machine learning algorithms using Python 3.11.12, including Logistic Regression, Linear Discriminant Analysis (LDA), Support Vector Machine (SVM), K-Nearest Neighbors (k-NN), and Decision Tree.

2. Measured classification accuracy and computational time for each model. Classification Accuracy was calculated using the formula:

   $$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \times 100\%$$

   Computational time was measured in seconds using the system clock, and includes:

   - **Training Time:** The time taken to train the model on the training dataset using the `fit()` function.
   - **Prediction Time:** The time taken by the trained model to make predictions on the testing dataset using the `predict()` function.

3. The parameters of each model are listed in Table 5

4. Applied 5-fold cross-validation to evaluate each model's performance. This approach ensures robust results by reducing the variance associated with a single train-test split and providing a more reliable estimate of generalization performance.

5. Recorded performance metrics for subsequent statistical analysis under the Balanced Incomplete Block Design (BIBD) framework to evaluate model differences across datasets.

3.5 Model Evaluation

Prior to conducting ANOVA, we first examine the residuals from an initial linear model to evaluate the key assumptions of homoscedasticity and normality. In cases where residual plots suggest violations—such as patterns indicating non-constant variance—we consider

Model	Parameters Used	Default or Custom	Reason for Choice
SVM	`kernel='rbf'`, `random_state=42`	Default with seed	'rbf' is default, random_state ensures reproducibility.
LR	`max_iter=200`, `random_state=42`	Custom	Increased max_iter for convergence, random_state for reproducibility.
LDA	None specified	Default	Defaults sufficient for small/mid datasets.
DT	`random_state=42`	Default with seed	random_state for reproducibility.
K-NN	None specified	Default	Using default `n_neighbors=5`, suitable for baseline.

Table 5: Parameter settings for each model (indicating whether default or custom) and reasons for these choices.

applying a power transformation to the response variable to stabilize variance and improve model fit.

One widely used approach is the Box–Cox transformation, which is defined as:

$$y^{(\Lambda)} = \begin{cases} \dfrac{y^{\Lambda} - 1}{\Lambda}, & \Lambda \neq 0 \\ \ln(y), & \Lambda = 0 \end{cases}$$

where y is the original (strictly positive) response value, and Λ is a parameter estimated to maximize the log-likelihood of the transformed data under the assumption of normally distributed residuals.

By applying the Box–Cox transformation before fitting the linear model, we aim to ensure the residuals meet ANOVA assumptions, thereby improving the validity of subsequent statistical inference.

We now proceed with the general linear model structure used for BIBD experiments. Let y_{ij} denote the observed (possibly transformed) response when treatment j is applied in block i.

- **Linear model for a BIBD experiment**:

$$y_{ij} = \eta + \alpha_i + \tau_j + \epsilon_{ij}$$

where η is the overall mean, α_i is the effect of block i, τ_j is the effect of treatment j, and ϵ_{ij} is the random error term associated with observation (i, j).

- **Corrected total sum of squares**:

$$\sum_i \sum_j (y_{ij} - \bar{y}_{..})^2$$

where the summation is taken over all observed (i, j) pairs. This expression captures the total variability in the data relative to the grand mean $\bar{y}_{..}$.

- **Adjusted treatment total.**

For treatment j,

$$Q_j = \sum_{i=1}^{b} y_{ij} - \sum_{i=1}^{b} n_{ij} \bar{y}_{i.}$$

where $\bar{y}_{i.}$ is the mean response in block i, and $n_{ij} = 1$ if treatment j appears in block i, and $n_{ij} = 0$ otherwise.

- **Sum of squares of adjusted treatment**:

$$\mathrm{SS}_{\mathrm{Treatment}} = \frac{k}{\lambda t} \sum_{j=1}^{t} Q_j^2$$

After computing the ANOVA table for each metric based on this model, we can conduct F-test. The resulting F-tests indicate whether there are statistically significant differences between models. If the null hypothesis of equal treatment effects is rejected by the F-test, we may proceed to identify which specific treatments differ using multiple comparisons. The least squares estimate of the treatment effect τ_j is given by:

$$\hat{\tau}_j = \frac{k}{\lambda t} Q_j$$

The t-statistic for comparing treatment j and i is then:

$$t_{ij} = \frac{\hat{\tau}_j - \hat{\tau}_i}{\hat{\sigma}\sqrt{2k/\lambda t}}$$

To determine whether two treatments are significantly different at significance level α, we use Tukey's multiple comparison method. Treatments i and j are considered significantly different if:

$$|t_{ij}| > \frac{1}{\sqrt{2}} q_{t,\, tr-b-t+1,\, \alpha}$$

where q denotes the studentized range distribution critical value. This procedure controls the family-wise error rate and enables pairwise comparisons across all treatments.

4 Results

In this section, we present the experimental results from our BIBD-based comparison of five machine learning models. The metrics evaluated include classification accuracy, overall execution time, and testing time. For each metric, we perform residual diagnostics to verify the model assumptions and explore if a potential Box–Cox transformation is appropriate. After that, we perform statistical analysis using ANOVA and paired comparisons as described in the BIBD section of Wu and Hamada [2011] for the transformed response matrix.

4.1 Classification Accuracy

Table 6 and 7 reports the classification accuracy of each method on the datasets where it was assigned. Here and thereafter, cells marked with "–" indicate that the method was not applied to the corresponding dataset as dictated by the BIBD. Dataset Abbreviations: "Wine Q.": Wine Quality, "Bank M.": Bank Marketing, "Heart D.": Heart Disease, "Car Eval.": Car Evaluation, "Cancer": Breast Cancer Wisconsin. "Adult": Adult Census Income.

Table 6: Accuracy on D1–D5

Method	Iris	Spambase	Mushroom	Adult	Wine Q.
SVM	0.9667	0.9318	0.9830	0.8497	0.5724
LR	0.9533	–	0.9830	–	0.5443
LDA	0.9800	–	–	0.8383	–
DT	–	0.9172	0.9844	–	–
k-NN	–	0.9109	–	0.8224	0.5660

Table 7: Accuracy on D6–D10

Method	Bank M.	Heart D.	Wine	Car Eval.	Cancer
SVM	0.8965	–	–	–	–
LR	–	0.8419	0.9776	0.9138	–
LDA	0.8926	0.8316	–	0.8947	0.9613
DT	0.8687	0.7776	0.9438	–	0.9455
k-NN	–	–	0.9719	0.8131	0.9684

Figure 1 illustrates the residuals versus fitted values plot for the pre-transformation data, providing a preliminary diagnostic of the response variable. Although the difference

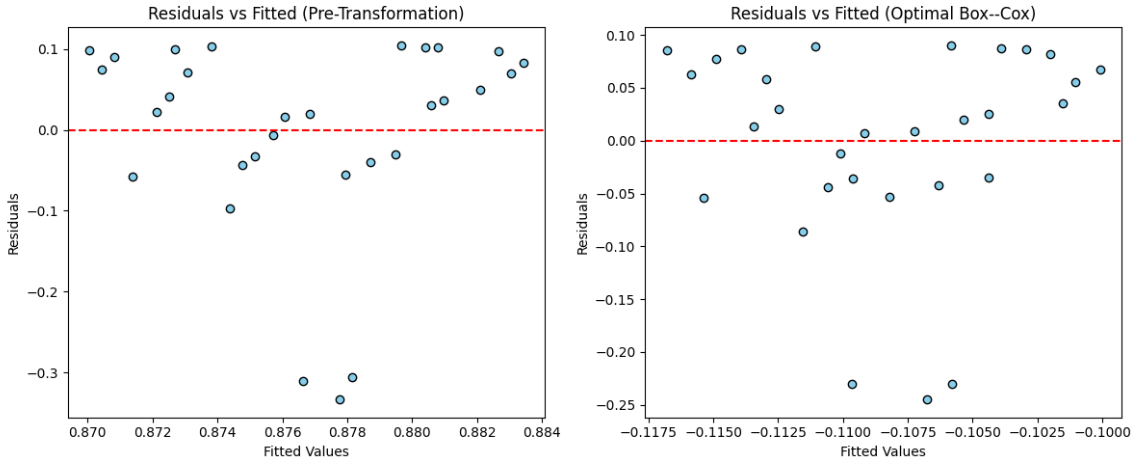

Figure 1: Residual plot of accuracy

is small, we any use the maximum likelihood Box–Cox variation with $\Lambda = 2.0$. It can be seen that the distribution basically conforms to the assumptions except for three outliers.

After the transformation, to statistically evaluate whether the observed accuracy differences among models are meaningful, we conduct an analysis of variance (ANOVA) based on the BIBD framework. The resulting ANOVA table, shown in Table 8, partitions the total variability into components attributable to blocks, treatments, and residual error. This allows us to test the null hypothesis that all models perform equally well in terms of classification accuracy. From the results of the F-test, blocks and treatments both have significant impacts on accuracy, which is consistent with our intuition.

Table 8: ANOVA Table for Classification Accuracy

Source	**DF**	**SS**	**MS**	**F-test**	**p-value**
Block	9	0.255441	0.028382	93.774899	3.951793e-12
Treatment (adjusted)	4	0.003084	0.000771	2.547251	7.981944e-02
Residual	16	0.004843	0.000303	–	–
Total	29	0.263367	–	–	–

Following the ANOVA, we apply Tukey's pairwise comparison procedure to further examine which specific pairs of models, if any, differ significantly. Table 9 reports the estimated differences in treatment effects, the corresponding t-statistics, and whether the differences are statistically significant at the $\alpha = 0.05$ level. From the results, we can know that although different machine learning algorithms have a significant impact on the accuracy results overall, there is no significant difference in the accuracy of the results obtained by two specific algorithms. Figure 2 shows how the pairwise t-statistics vary with different Box–Cox λ values, demonstrating from another perspective, that different transformations will lead to the same conclusion.

Table 9: Tukey Pairwise Comparisons Between Treatments for Classification Accuracy

Comparison	$\hat{\tau}_i - \hat{\tau}_j$	t**-statistic**	**Significant** ($\alpha = 0.05$)
SVM vs LR	-0.001601	-0.145478	No
SVM vs LDA	-0.002951	-0.268162	No
SVM vs DT	0.023557	2.140968	No
SVM vs k-NN	0.018021	1.637859	No
LR vs LDA	-0.001350	-0.122685	No
LR vs DT	0.025158	2.286445	No
LR vs k-NN	0.019622	1.783337	No
LDA vs DT	0.026508	2.409130	No
LDA vs k-NN	0.020972	1.906021	No
DT vs k-NN	-0.005536	-0.503108	No

4.2 Overall Execution Time

Table 10 and 11 present the overall execution time (in seconds), including both training and testing. These measurements provide insight into the total computational cost of model use.

Table 10: Overall Execution Time (seconds) on D1–D5

Method	Iris	Spambase	Mushroom	Adult	Wine Q.
SVM	0.007061	0.238736	0.332873	119.362869	1.144839
LR	0.020394	–	0.010402	–	1.088503
LDA	0.021561	–	–	0.241146	–
DT	–	0.060355	0.013481	–	–
k-NN	–	0.062739	–	1.481440	1.131901

Figures 3 similarly present the residuals versus fitted values. The residuals show a power function relationship with the fitted values. By maximizing the likelihood function of the residuals being normally distributed, we get the most likely Box-Cox transformation corresponding to $\Lambda = -0.222$. Through the transformation, the residual graph shows very good distribution properties, which is consistent with the hypothesis.

It is worth mentioning that although it may be difficult to explain the meaning of this transformation physically, we can still judge the quality of the algorithm by the size rela-

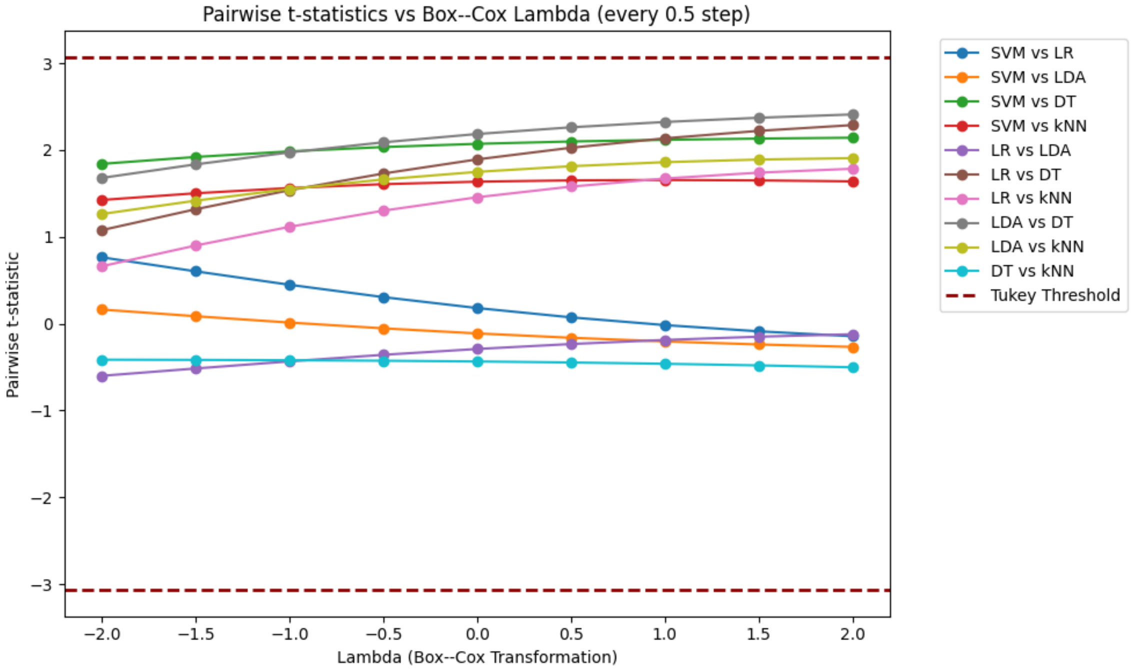

Figure 2: Scaled lambda plot of accuracy

Table 11: Overall Execution Time (seconds) on D6–D10

Method	Bank M.	Heart D.	Wine	Car Eval.	Cancer
SVM	13.844731	–	–	–	–
LR	–	0.005443	0.005909	0.015591	–
LDA	0.058432	0.004506	–	0.004242	0.004375
DT	0.256224	0.002785	0.002823	–	0.009897
k-NN	–	–	0.007124	0.028150	0.010286

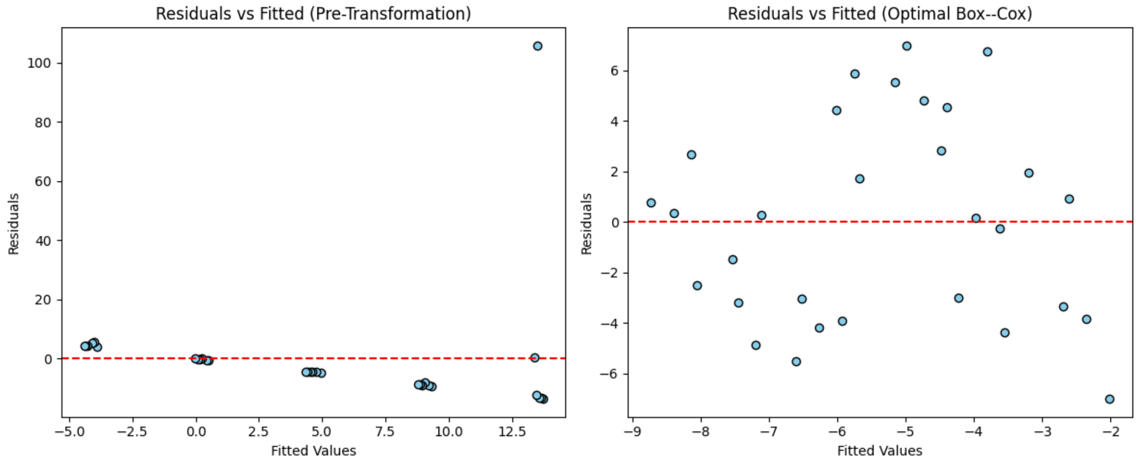

Figure 3: Residual plot of execution time

tionship, that is, the smaller value after the transformation, the shorter time consumption, and the better the model, which can achieve our goal.

To assess whether execution time differs significantly among models, we apply the same ANOVA framework as before. The results are presented in Table 12, followed by the corresponding pairwise comparisons in Table 13. These analyses reveal whether certain models are computationally more efficient than others in terms of total runtime. From the results, we know that the difference in data sets and algorithms has a significant impact on the time taken by the entire algorithm. And like the accuracy, the impact of data sets is more significant. This also proves the necessity of blocking, that is, the difference between groups is greater than the difference within groups. What's more, from the pirwise comparison result, LDA is shown better than SVM when considering the total time executed.

Also, Figure 4 shows how the pairwise t-statistics vary with different Box–Cox lambda values, offering guidance on whether a transformation is warranted. This shows that if logarithmic transformation is considered, DT can also achieve significant better results than SVM.

4.3 Testing Time

While training time contributes to the overall development cost, in many real-world applications—especially in industrial or production environments, the models are trained offline and deployed repeatedly for inference. As such, the testing (or prediction) time becomes a critical performance metric. For example, in high-frequency trading, real-time recommendation systems, or large-scale user-facing services, models are expected to deliver predictions with minimal latency. Therefore, it is important to evaluate and compare the

Table 12: ANOVA Table for Execution Time

Source	**DF**	**SS**	**MS**	**F-test**	**p-value**
Block	9	482.110827	53.567870	16.509	0.000002
Treatment (adjusted)	4	39.335656	9.833914	3.031	0.048831
Residual	16	51.915678	3.244730	–	–
Total	29	573.362162	–	–	–

Table 13: Tukey Pairwise Comparisons Between Treatments for Execution Time

Comparison	$\hat{\tau}_i - \hat{\tau}_j$	***t*-statistic**	**Significant** ($\alpha = 0.05$)
SVM vs LR	1.695394	1.488166	No
SVM vs LDA	3.523456	3.092784	Yes
SVM vs DT	2.926818	2.569073	No
SVM vs k-NN	1.170578	1.027499	No
LR vs LDA	1.828062	1.604618	No
LR vs DT	1.231425	1.080908	No
LR vs k-NN	-0.524815	-0.460667	No
LDA vs DT	-0.596638	-0.523711	No
LDA vs k-NN	-2.352878	-2.065285	No
DT vs k-NN	-1.756240	-1.541575	No

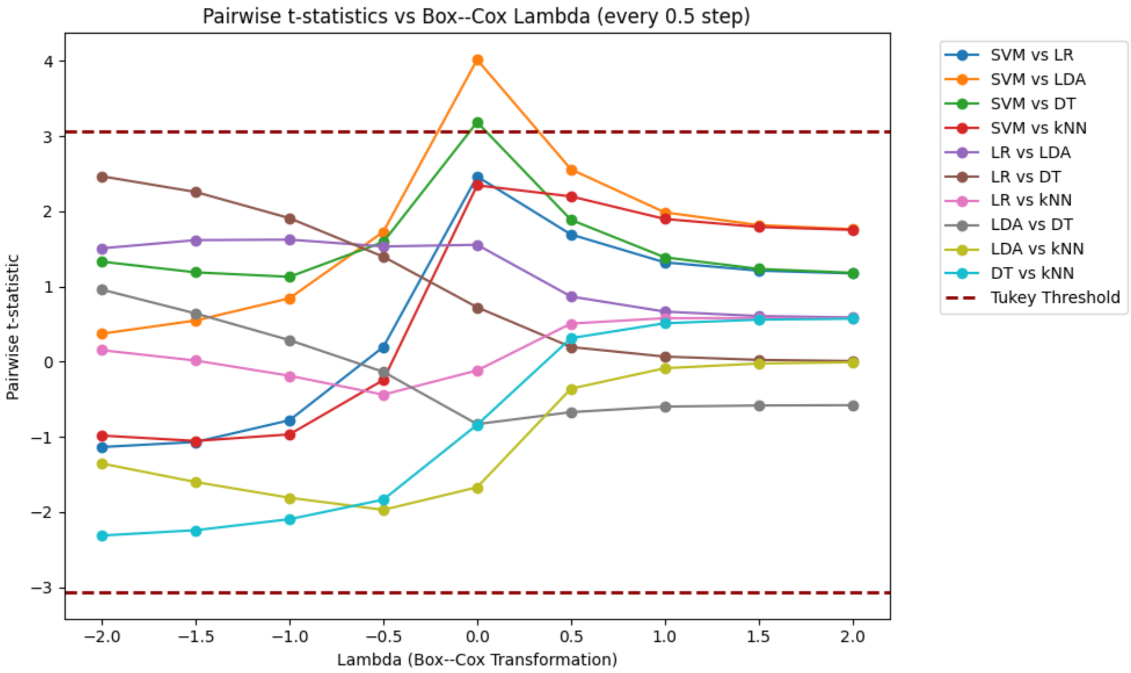

Figure 4: Scaled lambda plot of execution time

inference efficiency of each method independently from its training phase.

Table 14 and 15 show the average testing time (in seconds) for each method-dataset pair. These results reflect the efficiency of model inference.

Table 14: Testing Time (seconds) on D1–D5

Method	Iris	Spambase	Mushroom	Adult	Wine Q.
SVM	0.000768	0.079551	0.095866	12.183115	0.572419
LR	0.000329	–	0.000537	–	0.544251
LDA	0.000412	–	–	0.040539	–
DT	–	0.002471	0.002117	–	–
k-NN	–	0.055575	–	1.444059	0.565951

Table 15: Testing Time (seconds) on D6–D10

Method	Bank M.	Heart D.	Wine	Car Eval.	Cancer
SVM	4.423630	–	–	–	–
LR	–	0.000473	0.000362	0.000326	–
LDA	0.000970	0.000518	–	0.000363	0.000321
DT	0.002513	0.000383	0.000509	–	0.001244
k-NN	–	–	0.005310	0.025328	0.008906

Figures 5 shows the corresponding residual for testing time. Similarly as in Section 4.2, we get the most likely Box-Cox transformation corresponding to $\Lambda = -0.182$.

Next, we apply the same analysis to testing time, which reflects the efficiency of model inference in deployment scenarios. The corresponding ANOVA and Tukey comparison results are summarized in Tables 16 and 17, respectively. We also use Figure 6 to show how the pairwise t-statistics vary with transformations.

Table 16: ANOVA Table for Testing Time

Source	**DF**	**SS**	**MS**	**F-test**	**p-value**
Block	9	985.722546	109.524727	13.115	0.000008
Treatment (adjusted)	4	363.753662	90.938415	10.889	0.000186
Residual	16	133.621257	8.351329	–	–
Total	29	1483.097465	–	–	–

From the results, we can conclude that the LR, LDA, and DT algorithms are significantly better than SVM and k-NN in terms of testing time consuming.

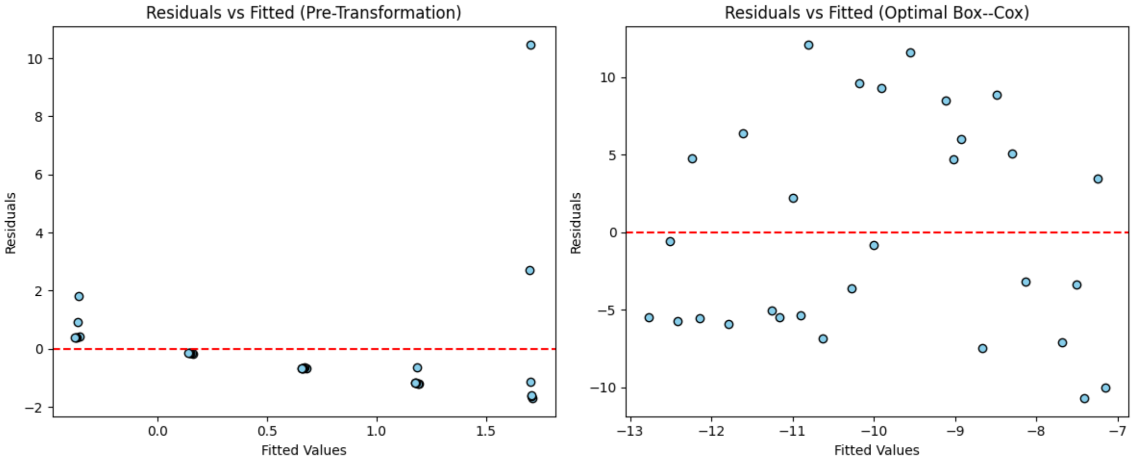

Figure 5: Residual plot of testing time

Table 17: Tukey Pairwise Comparisons Between Treatments for Testing Time

Comparison	$\hat{\tau}_i - \hat{\tau}_j$	t**-statistic**	**Significant** $(\alpha = 0.05)$
SVM vs LR	7.793097	4.263853	Yes
SVM vs LDA	8.564767	4.686059	Yes
SVM vs DT	7.155366	3.914930	Yes
SVM vs k-NN	0.212415	0.116219	No
LR vs LDA	0.771671	0.422206	No
LR vs DT	-0.637731	-0.348923	No
LR vs k-NN	-7.580682	-4.147635	Yes
LDA vs DT	-1.409402	-0.771129	No
LDA vs k-NN	-8.352353	-4.569841	Yes
DT vs k-NN	-6.942951	-3.798712	Yes

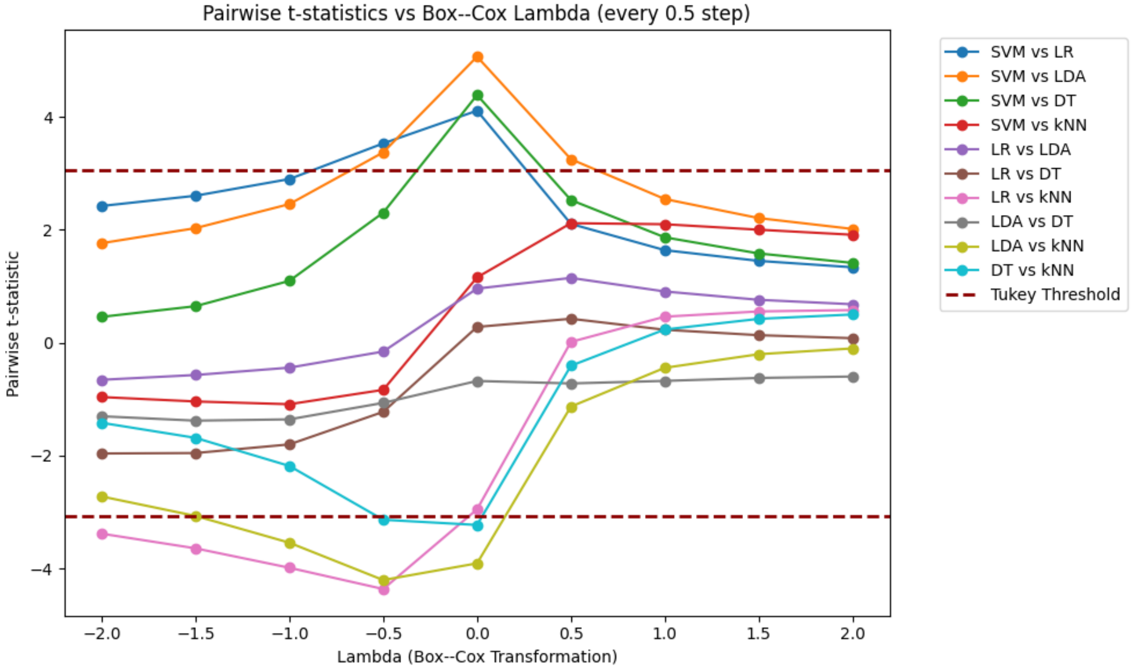

Figure 6: Scaled lambda plot of testing

5 Discussion

This study evaluates five classical classification algorithms using a Balanced Incomplete Block Design (BIBD), coupled with ANOVA, residual analysis, and Box–Cox transformation, to rigorously assess performance across classification accuracy, overall execution time, and testing time. Our findings are summarized below:

- **Effectiveness of Blocking:** Across all three metrics, the BIBD-based ANOVA indicates that both block and treatment effects are statistically significant. This confirms that dataset-specific variability (blocks) has a strong influence on observed performance, justifying the use of proper blocking in comparative algorithmic studies.
- **Classification Accuracy:** Although treatment effects are statistically significant, post-hoc comparisons (e.g., Tukey's test) reveal no significant pairwise differences. This suggests that under moderate-to-large sample sizes, all five algorithms reach stable performance, resulting in negligible performance gaps.
- **Overall Execution Time:** LDA consistently demonstrates better training efficiency compared to SVM. Unlike SVM, which involves iterative optimization (e.g., quadratic programming), LDA relies on closed-form solutions involving simple linear algebra, yielding shorter training durations across datasets.
- **Testing Time:** Logistic Regression, LDA, and Decision Trees are significantly faster than SVM and k-NN during inference. LR and LDA make predictions via simple dot products, while DT involves relatively shallow tree traversal. In contrast, SVM's reliance on support vectors and k-NN's nearest-neighbor search incur higher inference costs, especially in larger datasets.
- **Box–Cox Transformation Insights:** Residual diagnostics suggested heteroscedasticity in some response metrics. Applying the Box–Cox transformation improved variance stabilization and normality, thereby strengthening the validity of ANOVA assumptions. While transformed metrics (e.g., transformed time) may lose interpretability, the gain in model robustness is valuable.
- **Limitations:** First, while BIBD reduces experimental burden and ensures balance, it inherently limits the number of algorithm–dataset combinations observed. If critical performance differences between algorithms occur only on datasets excluded from direct comparison, such distinctions may be missed. This makes it challenging to conclusively rank models when performance differences are dataset-specific. Second, our analysis is confined to classification tasks; algorithmic behavior in regression, time-series, or unsupervised settings remains unexplored.

Summary. The proposed BIBD for Machine Learning (BIBD4ML) framework enables statistically rigorous and resource-conscious comparison of machine learning algorithms.

While classification accuracy appears largely consistent across methods, notable disparities arise in computational costs. LDA, LR, and DT are preferable for efficient inference, whereas SVM and k-NN may pose bottlenecks in large-scale or real-time applications. Diagnostic tools like the Box–Cox transformation further enhance the statistical validity of model evaluation under practical constraints.

References

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

CF Jeff Wu and Michael S Hamada. *Experiments: planning, analysis, and optimization.* John Wiley & Sons, 2011.