

# **Classifying with Lazy Learning & SVM**

Emre Karaoglu, Zoë Finelli, Noel Torres Carretero, Onat Bitirgen

## Table of Contents

<b>Introduction.....</b>	<b>3</b>
<b>Methodology.....</b>	<b>4</b>
<b>Results &amp; Analysis.....</b>	<b>8</b>
<b>Conclusion.....</b>	<b>21</b>
<b>References.....</b>	<b>23</b>

# 1 Introduction

## 1.1 Overview

Instance-based learning represents a key approach in machine learning where classification decisions come from directly comparing new instances to stored training examples, rather than by constructing explicit generalization models. Among these approaches, k-Nearest Neighbor (k-NN) algorithms have demonstrated great versatility across diverse problem domains, though their performance depends critically on the choice of the k-value, distance metrics, voting mechanisms, retention policies. This work undertakes a comprehensive comparative analysis of lazy learning methods and Support Vector Machines (SVMs), evaluating the impact of algorithmic design decisions and preprocessing techniques on classifier performance and computational efficiency.

This study has three main objectives. First, we implement and evaluate a k-Instance-Based Learning (k-IBL) algorithm, systematically analyzing the impact of different similarity metrics, neighborhood sizes, voting policies, and retention strategies to determine the optimal configuration. Second, we assess the effectiveness of preprocessing techniques, specifically feature weighting and instance reduction, in enhancing both classification accuracy and computational efficiency. Third, we evaluate SVM classifiers, exploring different kernel functions and hyperparameter settings to identify optimal parametrizations, and compare the best-performing SVM against the optimized k-IBL algorithm.

## 1.2 Dataset characteristics

To ensure robust evaluation across diverse problem characteristics, we selected two datasets from the UCI Machine Learning Repository that differ substantially in size, dimensionality, attribute types, and class distribution:

**Adult Income Dataset:** This dataset, extracted from the 1994 U.S. Census database, contains 48,842 instances described by 14 attributes (8 nominal and 6 numeric) predicting whether an individual's annual income exceeds \$50,000. The dataset exhibits moderate class imbalance (76.07% majority class, 23.93% minority class) and contains 0.95% missing values, presenting realistic challenges in handling heterogeneous data types and incomplete information. With the mix of continuous and categorical features, the Adult dataset allows for the ideal evaluation of distance metrics designed for mixed-attribute spaces.

**Pen-Based Handwritten Digit Recognition Dataset:** This dataset comprises 10,992 instances of handwritten digits (0-9) captured from 44 writers, with each instance represented by 16 numeric features derived from spatial coordinates. The dataset contains 10 balanced classes (standard deviation of 0.40%) and has no missing values, offering a smooth, high-dimensional numerical classification problem. The balanced multi-class nature of this dataset provides contrast to the binary, imbalanced Adult dataset, allowing us to assess algorithm behavior across different classification challenges. Table 1 summarizes the key characteristics of both datasets, highlighting their complementary properties for comprehensive algorithm evaluation.

Dataset	#Cases	#Num	#Nom	#Classes	Dev.Cla.	Maj.Cla.	Min.Cla.	MV
Adult	48,842	6	8	2	26.07%	76.07%	23.93%	0.95%
Pen-Based	10,992	16	0	10	0.40%	10.41%	9.60%	0%

Table 1: Characteristics of selected datasets

## 1.3 Methodology Overview

In order to guarantee reproducibility and equitable comparison, our experimental design follows a rigorous cross-validation framework using predefined 10-fold partitions. All experiments maintain consistent evaluation processes across algorithms, with performance being measured utilizing three metrics: classification accuracy (percentage of correctly classified instances), computational efficiency (average problem-solving time), and storage requirements (percentage of training instances retained).

The k-IBL implementation explores three neighborhood sizes ( $K = 3, 5, 7$ ), three distance metrics (Euclidean, Cosine, and Heterogeneous Euclidean-Overlap Metric), two voting policies (Modified Plurality and Borda Count), and four retention strategies (Never Retain, Always Retain, Different Class Retention, and Degree of Disagreement). Statistical significance of performance differences is assessed using the Friedman test followed by Nemenyi post-hoc pairwise comparisons with  $\alpha = 0.05$ , following the methodology outlined by Demšar (2006) for comparing multiple classifiers across multiple datasets. Each of the 10 cross-validation folds within each dataset is treated as an independent dataset.

Beyond the baseline k-IBL analysis, we investigate two feature weighting methods and three instance reduction techniques, each implemented as preprocessing steps applied to the training data. For comparison, we evaluate SVM classifiers with two kernel functions and varying hyperparameter configurations ( $C$  and  $\gamma$  values), examining their standalone performance and their robustness to training set reduction.

This report is structured as follows: Section 2 details the implementation methodology for all algorithms and preprocessing techniques; Section 3 presents comprehensive experimental results and findings in the context of the assignment questions and broader implications; and Section 4 concludes with practical recommendations and future research directions.

## 2 Methodology

### 2.1 Data Processing

Both datasets were preprocessed to make them comparable across models and experimental settings.

**Adult** - Continuous features were normalized to  $[0,1]$  (min-max scaling) so that no single numeric attribute dominated distance-based methods. Categorical features were converted to numeric using one-hot/overlap-style encoding so they could be handled both by HEOM (natively supports categorical overlap) and by Euclidean/Cosine when needed. The binary target (" $>50K$ " vs " $\leq 50K$ ") is class-imbalanced; we kept the original labels unchanged to reflect the real distribution.

**Pen-Based** - Features were scaled to a common numeric range to avoid magnitude bias across dimensions. No categorical encoding was required. For all experiments, we used stratified folds so that class proportions were preserved between train and test splits. The "training set" produced by each fold is what the k-IBL algorithm actually stores as its case base (possibly after retention / instance reduction). When computing feature weights (e.g. Mutual Information or ReliefF), the weights were learned only from the training fold and then applied to that fold's features before evaluation, ensuring no test leakage.

### 2.2 KIBL algorithm Implementation

We implemented a general k-Instance-Based Learner (k-IBL) that unifies all the design choices studied in this work — distance metric, neighborhood size, voting rule, and retention policy — under a single training / inference interface.

#### Training phase.

Unlike parametric learners, k-IBL does not estimate model parameters. The "training" step simply stores the observed instances  $(x,y)$  in an internal case base. Optionally, a retention policy can be applied at this stage to decide which cases are actually kept. We support three retention modes:

- **Keep-all:** store every training example (classical k-NN behaviour).
- **Filtered:** reject or discard low-quality / low-utility cases according to simple heuristics (e.g. instances that are clear outliers or redundant near-duplicates).
- **Reduced-set input:** accept an externally reduced subset (e.g. after ENN, TCNN, ICF) and use that as the memory from the start.

This design lets us study retention as an explicit modeling choice rather than an afterthought.

### Distance computation.

At inference time, the algorithm computes the distance between the query point  $x_{qx\_qxq}$  and every stored case. The implementation exposes a pluggable distance module:

- **Euclidean distance** for purely numerical, well-scaled data.
- **Cosine distance** for high-dimensional directional similarity.
- **HEOM** (Heterogeneous Euclidean-Overlap Metric) for mixed numeric + categorical data. HEOM normalizes continuous attributes and applies an overlap penalty for categorical attributes, so no single attribute (e.g. age or capital gain in Adult) dominates the comparison.

All distance functions are implemented to operate feature-wise, then aggregate into a scalar distance. For HEOM in particular, missing values and categorical mismatches are handled explicitly, which is important for Adult.

### Neighbour selection.

For a given query, we sort all stored cases by distance and take the  $k$  closest ones. The value of  $k$  is configurable (we evaluate  $k \in \{3, 5, 7\}$ ). Larger  $k$  typically increases stability (lower variance, more noise smoothing) but may blur fine local boundaries; smaller  $k$  is more sensitive to local structure but also more vulnerable to mislabeled points.

### Voting / label decision.

Classification is done by aggregating the labels of the  $k$  neighbours. We implement multiple voting rules:

- **Modified Plurality:** majority vote with tie-breaking that favours neighbours with smaller distance and, when relevant, class priors. This is robust on noisy, imbalanced data because it does not let a single outlier dominate.
- **Borda Count:** each neighbour “ranks” all classes based on inverse distance, and classes accumulate points accordingly. Closer neighbours have more influence than farther ones, producing a softer preference ordering rather than a hard count.
- (In binary cases, this reduces to a weighted yes/no decision; in multi-class problems like Pen-Based, it helps distinguish between several plausible digits.)

At prediction time the voting rule is applied to the retrieved neighbourhood and the class with the highest aggregated score is returned. Given a test instance, we compute its distance to every retained training instance using the chosen metric. Select the  $k$  nearest neighbours. Apply the selected voting rule to those neighbours. Output the predicted class.

## 2.3 Support Vector Machine (SVM) algorithm Implementation

### 2.3.1 Algorithm Configuration

We implemented Support Vector Machines using the scikit-learn library with the SVC classifier. SVMs construct optimal hyperplanes that maximize the margin between classes while allowing controlled misclassification through a regularization framework. The algorithm balances two competing objectives: maximizing the geometric margin (distance from the decision boundary to the nearest training examples [support vectors]) and minimizing classification errors. The regularization parameter  $C$  controls this trade-off; larger  $C$  values penalize misclassifications more heavily, leading to potentially narrower margins but fewer training errors.

We first evaluated two kernel functions to handle different data characteristics. RBF and linear kernels are the most widely used in practice due to their complementary strengths and robust performance across diverse datasets (Hsu et al., 2003). The Radial Basis Function (RBF) kernel,  $K(x_i, x_{\square}) = \exp(-\gamma \|x_i - x_{\square}\|^2)$ , enables modeling of non-linear decision boundaries through implicit mapping to higher-dimensional spaces. The gamma parameter controls the influence radius of individual training samples. The Linear kernel,  $K(x_i, x_{\square}) = x_i \cdot x_{\square}$ , computes hyperplane decision boundaries directly in the original feature space, providing computational efficiency for problems that are linearly separable. The RBF kernel is recognized as an efficient default choice because it can handle non-linear relationships between features and class labels, with the linear kernel being a unique case of RBF under certain parameter settings (Hsu et al., 2003). Furthermore, other options such as the polynomial

and sigmoid kernels introduce additional hyperparameters (degree, coefficient) that significantly expand the search space without consistently outperforming RBF kernels in practice, adding numerical difficulties to the analysis.

Our parameter grid consisted of 24 configurations systematically testing the interplay between regularization strength and kernel complexity. We evaluated four regularization values ( $C = 0.1, 1.0, 10.0, 100.0$ ) combined with five RBF kernel gamma parameters ( $\gamma = 0.001, 0.01, 0.1, 1.0$ , and 'scale'). The 'scale' parameter adaptively sets gamma as the inverse of the product of the number of features and the variance of the input data, allowing the kernel width to adjust based on dataset characteristics. These parameter ranges follow established best practices in SVM tuning literature (Hsu et al., 2003), where the  $C$  values evaluated were chosen in an exponentially growing sequence. Similarly, the gamma values range from broad generalization (0.001) to highly localized decision boundaries (1.0). This systematic grid search approach has been validated as effective for identifying optimal SVM configurations across diverse datasets (Chapelle et al., 2002).

### 2.3.2 Statistical Parameter Selection

To identify optimal configurations rigorously, we employed Friedman statistical tests followed by Nemenyi post-hoc analysis, a methodology process widely accepted for comparing multiple classifiers across multiple datasets (Demšar, 2006). The Friedman test evaluates whether significant performance differences exist among the 24 configurations across 10 folds.

When the Friedman test detected significance, we applied Nemenyi post-hoc tests to identify which configuration pairs differ significantly. Two configurations are considered statistically equivalent if their average rank difference does not exceed the critical difference  $CD = q\alpha \sqrt{(k(k+1)/6N)}$ , where  $q\alpha$  is the critical value at significance level  $\alpha = 0.05$ ,  $k = 24$  configurations, and  $N = 10$  folds. Among parameter combinations not significantly worse than the best performer, we selected the final winner based on highest mean accuracy, using training time as a tie-breaker, ensuring statistical validity.

### 2.3.3 Instance Reduction Methods

To investigate accuracy-efficiency trade-offs, we evaluated three instance reduction techniques as dataset preprocessing steps before SVM training. Each method was applied to training folds only, with SVMs trained on reduced sets but tested on original test folds to evaluate whether computational savings justify any accuracy loss. We measured classification accuracy, training time reduction, and support vector counts for each configuration.

### 2.3.4 Comparative Statistical Analysis

For comparing the best-performing SVM configuration to the best k-IBL configuration, we used two-sided Wilcoxon signed-rank tests on the paired fold accuracies. This non-parametric test does not assume normality and respects the pairing structure across folds. A p-value below 0.05 was taken as evidence of a statistically meaningful difference in performance (Demšar, 2006). To evaluate the effect of instance-reduction techniques, we used one-sided Wilcoxon signed-rank tests comparing the baseline SVM to each (instance) reduced variant. Here, a significant p-value ( $p < 0.05$ ) indicates that the reduction method leads to a detectable decrease in accuracy, while non-significance suggests that accuracy is statistically preserved. This one-sided formulation specifically tests whether reduction harms performance, rather than testing for any difference.

## 2.4 Feature Weighting

In the previous sections we identified the best configuration for the k-Instance Based Learner (k-IBL) on each dataset and SVM. The next step was to investigate whether weighting individual features prior to computing distances can improve the classifier. The motivation for feature weighting is intuitive: not all attributes contribute equally to the class decision, so assigning larger weights to informative variables should make nearest-neighbour comparisons more meaningful. In order to isolate the effect of weighting we retained the distance measure, neighbourhood size, voting policy and retention strategy of the baseline k-IBL for each dataset.

### 2.4.1 Methods

We considered two filter-based weighting methods drawn from the literature. **Mutual Information (MI)** measures the reduction of uncertainty about the class when a given feature is observed; higher MI values indicate stronger dependency. We computed MI using `sklearn.feature_selection.mutual_info_classif` and normalised the scores to the range 0,1 so that no single attribute dominates the distance. **ReliefF** is a local method based on nearest hits and misses; it assigns high weight to features that distinguish between neighbours of different classes. We implemented ReliefF via the `skrebate` library, using `n_neighbors = 1` as a compromise between stability and locality. Since ReliefF can be expensive on large folds, we set `min(10, n-1)` to avoid using more neighbours than available in small folds.

Weights were incorporated into the distance metric multiplicatively so that each attribute's contribution is scaled by its weight. For numeric attributes the weighted Euclidean distance was used, while for mixed data the weights were applied inside the Heterogeneous Euclidean–Overlap Metric (HEOM). All weights were re-scaled to 0,1 after computation.

### 2.4.2 Evaluation and Statistical Comparison

Computing weights from the entire training fold produces stable estimates but is computationally heavy for a lot of instances. To explore the accuracy–efficiency trade-off we ran each method in two modes. In the **complete** mode the weights were estimated from all available training instances. In the **subsampled** mode we sampled a random subset of 5,000 training instances (`sample_size=5000`, `random_state=42`) to estimate the weights. Sampling dramatically reduces preprocessing time (complete runs required several hours, whereas subsampled runs finished in under an hour) while still capturing the main information of the data. The subsample size was chosen to be large enough to stabilise MI and ReliefF estimates yet small enough to provide practical speed-ups.

### 2.4.3 Complete and subsampled modes

All experiments used the predefined ten-fold cross-validation splits provided with the data. In each fold the weights were computed on the training portion only and then applied to classify the instances in the corresponding test portion. As in the baseline analysis, we measured classification accuracy, problem-solving time and storage requirements. To decide whether weighting improved the classifier we compared the accuracy of the weighted model to the baseline on a fold-by-fold basis using the paired Wilcoxon signed-rank test. We corrected for multiple comparisons across the two weighting methods using Holm's step-down procedure at the  $\alpha = 0.05$  level. We followed Demšar's guidelines for comparing classifiers across multiple datasets. Time and storage were reported only for completeness and were not used in the acceptance criterion for feature weighting.

## 2.5 Instance Reduction

We apply classical instance-reduction methods to shrink the case base before k-IBL inference, targeting lower memory and faster queries while preserving accuracy. All reductions are performed within each training fold only (no test leakage), and the reduced set becomes the k-IBL memory.

- ENN (Edited Nearest Neighbours). Noise/boundary editor: removes training points misclassified by their local neighbours. Typically preserves accuracy (sometimes improves it) with modest size reduction; useful as a “safe cleaner.”
- Tomek/Condensed variants (TCNN). Overlap removal + prototype selection: deletes Tomek links (class overlap) and builds a consistent subset that still classifies the training set. Yields substantial compression and speedups, with a small accuracy cost.
- ICF (Iterative Case Filtering). Competence-guided condensation: iteratively keeps informative prototypes (often near decision boundaries) and discards redundant interiors. Produces the smallest memory and largest latency gains, with slightly larger—but usually acceptable—accuracy loss.

Evaluation axes. We report Accuracy, Inference Time/query, and Storage (% of original) to make the trade-off explicit. ENN  $\approx$  best for accuracy retention; TCNN/ICF  $\approx$  best for efficiency. Note that reduction complements (not replaces) retention policies: even with a fixed retention setting, up-front

selection removes redundancy the retention policy would continue to carry, delivering the dominant wins in memory and latency.

### 3 Results & Analysis

#### 3.1 Analysis 1: K-IBL Parameter Tuning

Goal of this section to identify the best k-IBL configuration by jointly tuning: 1) distance function  $\in \{\text{Euclidean, Cosine, HEOM}\}$ , 2) number of neighbors  $k \in \{3, 5, 7\}$ , 3) voting rule  $\in \{\text{Modified Plurality, Borda Count}\}$ , 4) retention policy  $\in \{\text{NR, DC, DD, AR}\}$  and then to analyze how these design choices affect: 1) **Accuracy** (higher is better), 2) **Inference Time** (ms/query, lower is better), 3) **Storage** (% of retained instances, lower is better). All feature weights are uniform ( $w_i = 1$ ). All results are obtained with 10-fold cross validation. Each fold is treated as an independent dataset for statistical testing. We run this analysis separately on each dataset (Adult, Pen-based), because their difficulty and scale are very different.

##### 3.1.1 Adult Dataset

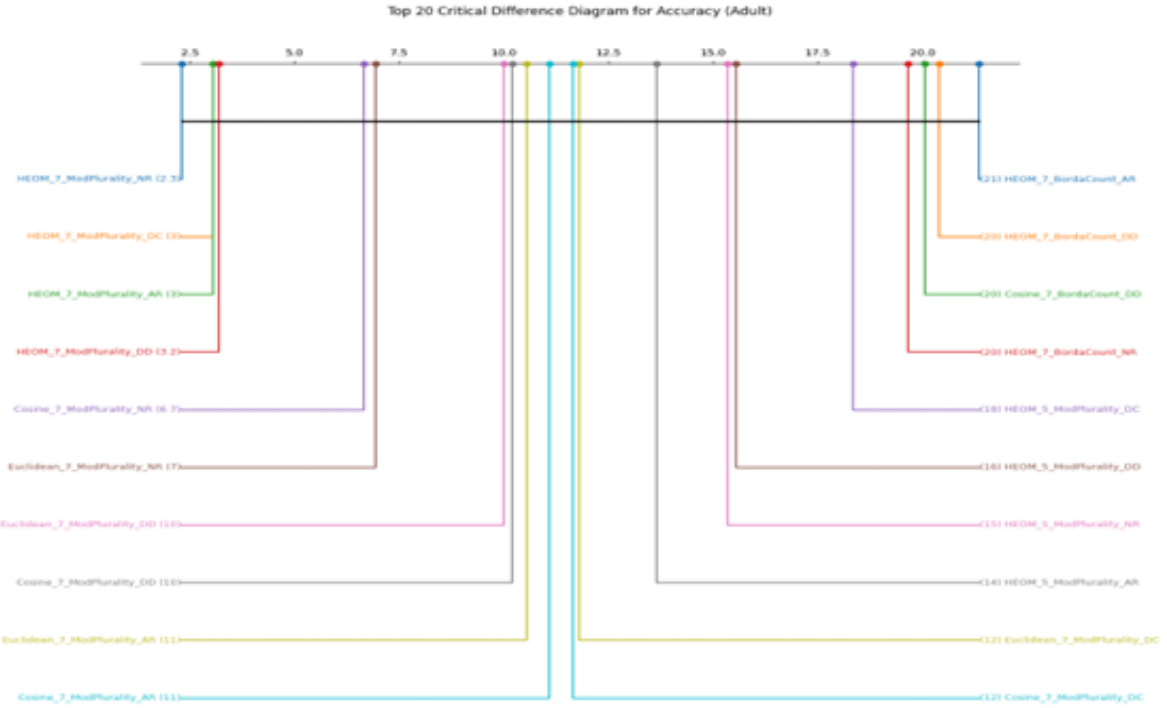


Fig 1. Critical Difference Diagram : Accuracy results for Adult

The best-ranked configuration is **HEOM\_7\_ModPlurality\_NR** (average rank  $\approx 2.30$ ).

Other HEOM-based variants with  $k = 7$  and Modified Plurality voting but different retention policies (DC, AR, DD) follow very closely (ranks  $\approx 3.05$ – $3.20$ ).

After these, the next tier consists of Cosine\_7\_ModPlurality\_NR and Euclidean\_7\_ModPlurality\_NR (ranks  $\approx 6.65$  and  $\approx 6.95$ ).

See the CD diagram of the top 20 configurations (Fig. 1).

- High  $k$  ( $k=7$ ), Modified Plurality voting, and HEOM distance dominate.
- NR retention policy is in the first position, but DC / AR / DD are statistically close; they fall in the same Nemenyi group.

This yields the following top 3 overall configurations on Adult (Table 2):

Algorithm	Average Rank (Accuracy)	Average Accuracy (%)	Average Time (ms)	Time_ms_Rank	Storage_Percent_Rank
HEOM_7_ModPlurality_NR	2.300	0.833	88.066	33.10	9.65
HEOM_7_ModPlurality_DC	3.050	0.832	171.182	37.50	57.50
HEOM_7_ModPlurality_AR	3.050	0.832	99.696	48.40	19.75

Table 2.K-IBL Top 3 Overall Algorithms

We apply the project's tie-break logic:

- (1) keep only configurations that are in the statistically best group for Accuracy,
- (2) break remaining ties using Time, then Storage.

Conclusion for Adult:

- If accuracy is the main objective, use HEOM distance,  $k = 7$ , Modified Plurality voting, and NR retention.
- If latency is critical, a different regime (Euclidean distance, small  $k$ , BordaCount or ModPlurality with DC/NR) is dramatically faster but sacrifices some accuracy.
- Storage does not drive the decision on Adult, because many configurations tie in Storage rank.

### 3.1.3 Pen-based Dataset

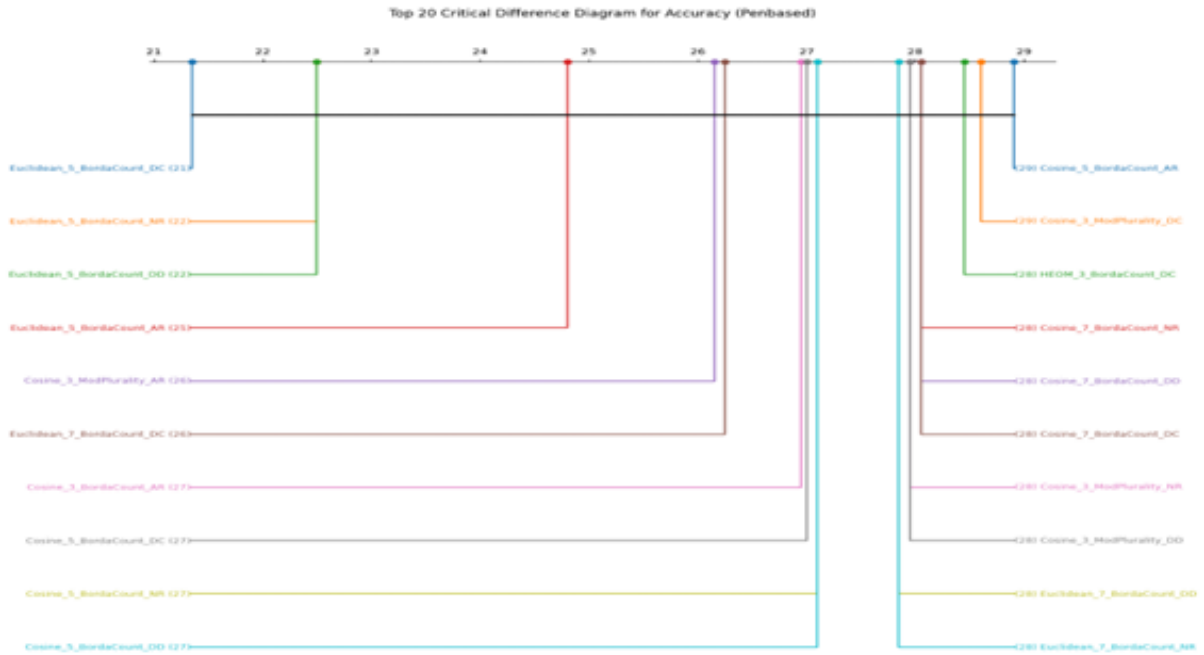


Fig 2. K-IBL Accuracy results for Pen-Based Critical Difference Diagram

The best-ranked configuration is Euclidean\_5\_BordaCount\_DC (average rank  $\approx 21.35$ ). Very close alternatives are Euclidean\_5\_BordaCount\_NR, Euclidean\_5\_BordaCount\_DD, and Euclidean\_5\_BordaCount\_AR (ranks  $\approx 22-25$ ) (Fig 2).

Key difference vs Adult:

- On Pen-based, many different settings are statistically indistinguishable in Accuracy.
- The problem is almost “solved”: most strong configs exceed 99% accuracy. That means accuracy alone is not enough to choose.

We again apply the same tie-break logic (Accuracy first, then Time, then Storage). The resulting top 3 configurations on Pen-based are (Table 3):

Algorithm	Average Rank (Accuracy)	Average Accuracy (%)	Average Time (ms)	Time_ms_Rank	Storage_Percent_Rank
HEOM_5_ModPlurality_DC	21.350	0.994	11.731	5.30	18.500
HEOM_5_ModPlurality_NR	22.500	0.994	11.700	6.70	18.500
HEOM_5_ModPlurality_DD	22.500	0.994	11.957	11.50	18.500

Table 3. Top 3 Overall Algorithms

Conclusion for Pen-based:

- Euclidean distance,  $k = 5$ , Borda Count voting is consistently among the top performers.
- Retention choice (DC vs NR vs DD) mainly trades storage vs. small latency differences, but does not move accuracy in a meaningful way because accuracy is already saturated ( $\sim 0.994$ ).

The statistical comparisons across both datasets reveal systematic and interpretable patterns in how k-IBL behaves under different design choices (distance metric, number of neighbors  $k$ , voting rule, and retention policy). The results are plausible and align with standard k-NN intuition.

**Adult** is a heterogeneous tabular dataset with both continuous and categorical attributes, and it is noticeably noisy. Configurations based on HEOM with a larger neighborhood and Modified Plurality voting — e.g. HEOM\_7\_ModPlurality\_NR — obtain the best Accuracy rank on Adult. This is consistent with the structure of the data:

- HEOM is explicitly designed for mixed-type inputs: it normalizes numeric features and uses an overlap-style comparison for categorical ones. This prevents any single high-variance numeric attribute from dominating the distance.

In contrast, much faster models such as Euclidean\_3\_BordaCount\_NR achieve excellent inference time (they are among the top-ranked methods for Time\_ms) but do not match the accuracy of the HEOM-based configuration. This is also expected: Euclidean distance with a small  $k$  is cheaper to evaluate but less robust to heterogeneity and mislabeled points.

**Pen-based** is a clean, fully numeric dataset where classes are already well separated in Euclidean space. Here, moderate neighborhoods with a simple metric are already near-perfect. Variants such as Euclidean\_5\_BordaCount\_NR, Euclidean\_5\_BordaCount\_DC, and Euclidean\_5\_BordaCount\_DD deliver  $\approx 0.994$  mean accuracy while keeping inference time around 11–12 ms/query. More complex metrics (HEOM) or larger neighborhoods do not significantly increase accuracy, because the task is intrinsically easy for Euclidean distance.

This makes the ranking on Pen-based plausible: Euclidean +  $k = 5$  + Borda Count dominates Accuracy, and essentially all top-accuracy configurations are tightly clustered and statistically indistinguishable. Runtime differences are smaller in absolute terms than in Adult because most strong Pen-based models are already both fast and extremely accurate.

#### Final recommendations:

- Adult (heterogeneous, noisier, harder):
  - If the priority is maximum predictive performance, choose HEOM\_7\_ModPlurality\_NR. It achieves the best Accuracy rank according to the Friedman/Nemenyi analysis.
  - If strict query-time latency is the priority (e.g. real-time response under tight time budgets), use lightweight Euclidean distance with a small neighborhood and Borda Count (e.g. Euclidean\_3\_BordaCount\_NR / Euclidean\_3\_ModPlurality\_DC). These configurations are among the top-ranked for Time\_ms, though they sacrifice some accuracy.
- Pen-based (clean, separable, numeric):

- Euclidean\_5\_BordaCount\_NR, Euclidean\_5\_BordaCount\_DC, and Euclidean\_5\_BordaCount\_DD are preferred. They all produce  $\approx 0.994$  mean accuracy, low inference time ( $\sim 11\text{--}12$  ms/query), and competitive storage ranks.
- Among them, NR in particular achieves an excellent storage rank (18.50) without hurting accuracy, and appears in the overall Top 3 when jointly considering Accuracy, Time, and Storage.
- No universal “best” configuration:  
There is no single k-IBL setting that dominates across both datasets and all metrics. The optimal configuration depends on (i) the structure of the data (heterogeneous vs. purely numeric, noisy vs. clean), and (ii) the deployment goal (raw accuracy vs. latency vs. memory). On Adult, robustness to noise and mixed feature types matters most, so HEOM + higher k + Modified Plurality is best. On Pen-based, the decision boundary is already simple in Euclidean space, so Euclidean + k = 5 + Borda Count is sufficient and more efficient.

### 3.1.3 Impact of Feature Weighting

We summarize below the results for both datasets and both modes. The difference in accuracy ( $\Delta$  points in percentage points) is computed as  $\text{mean\_FW} - \text{mean\_baseline}$ . A positive  $\Delta$  indicates an improvement over the unweighted baseline.

#### Pen-Based Dataset

For the pen-based handwritten digits (16 numeric attributes, ten balanced classes), the baseline accuracy was = 99.43% .

Mode	Method	Mean FW (%)	Mean baseline (%)	$\Delta$ points (%)	p(Holm)	Decision vs. baseline
Complete	FW (MI)	99.4361	99.4269	+0.0091	0.7500	No significant difference
	FW (ReliefF)	99.2996	99.4269	-0.1274	0.0156	Significantly worse
Subsample	FW (MI)	99.4270	99.4269	+0.0000	0.8438	No significant difference
	FW (ReliefF)	99.3542	99.4269	-0.0728	0.0313	Significantly worse

The pen-based dataset is clean and well-balanced, and the Euclidean distance already captures the geometry of the digits very well. The near-perfect baseline leaves little room for improvement; indeed MI offers a negligible, non-significant gain, while ReliefF introduces noise by overweighting local patterns that do not generalise across ten classes. Subsampling slightly reduces the magnitude of ReliefF’s drop (from  $-0.13$  to  $-0.07$  percentage points) but does not change the statistical outcome. In terms of computational cost, both complete and subsampled weighting have similar inference times ( $\approx 11.6$  ms per query) and storage ( $\approx 100\%$ ), so the primary trade-off is the long preprocessing time of the complete mode.

**Decision (Pen-based).** We conclude that feature weighting does not bring meaningful benefits for this numeric, balanced problem. The baseline k-IBL without weights remains the best choice.

#### Adult Dataset

For the Adult income dataset (mix of nominal and numeric attributes, binary and moderately imbalanced), the baseline accuracy was = 83.29%.

Mode	Method	Mean FW (%)	Mean baseline (%)	$\Delta$ points (%)	p(Holm)	Decision vs. baseline
Complete	FW (MI)	83.6002	83.2930	+0.3072	0.0039	Significantly better
	FW (ReliefF)	83.3565	83.2930	+0.0635	0.6250	No significant difference
Subsample	FW (MI)	83.4712	83.2930	+0.1782	0.0977	No significant difference
	FW (ReliefF)	83.3197	83.2930	+0.0267	0.7695	No significant difference

The Adult dataset contains a mixture of categorical and numeric attributes and exhibits class imbalance. MI handles categorical predictors well by estimating their global dependence on the class; increasing their weights under HEOM brings similar instances closer in the distance space. When weights are estimated from the full training fold, this leads to a statistically significant improvement of about 0.31 percentage points. ReliefF, being a local, class-insensitive method, provides only a small, non-significant gain. When using the 5 k subsample, MI still improves the baseline (+0.18 percentage points) but the result is not significant at the 5 % level. This suggests that sampling may under-represent rare categories, making MI estimates less stable.

Inference time is actually lower than in the unweighted baseline ( $\approx 52$  ms vs.  $\approx 88$  ms per query) because the baseline retains all training instances while Never-Retain reduces the number of reference cases. Storage requirements remain essentially unchanged. However these advantages are ancillary; the decision criterion is accuracy.

**Decision (Adult).** For the mixed-attribute Adult dataset we adopt feature weighting based on Mutual Information. Using the complete training fold yields a statistically significant improvement and is recommended when computational resources permit. The subsampled MI run still improves accuracy albeit without statistical support; it can serve as a practical “fast mode” when the preprocessing time of the complete run (several hours) is prohibitive.

### Discussion of Subsampling

Subsampling the training fold to estimate weights greatly reduces preprocessing time (from hours to minutes) at the cost of slightly less stable estimates. For pen-based, subsampling yields the same decision as the complete run—feature weighting is not adopted—so the speed-up is clearly worthwhile. For Adult, the subsampled MI run still improves the baseline and may be acceptable when a minor loss of statistical significance is tolerable. A larger subsample (e.g., 8 k–10 k instances) could offer a middle ground, but we did not explore this due to time constraints.

#### 3.1.4 Impact of Instance Reduction

We evaluated how classical instance reduction methods affect both predictive performance and computational cost of k-IBL. Specifically, we applied Edited Nearest Neighbors (ENN), Tomek/Condensed variants (TCNN), and more aggressive condensing methods such as ICF to the training folds before fitting k-IBL. After reduction, the classifier only stores the surviving subset as its case base; inference is then run exactly as usual on that reduced memory.

#### Adult dataset.

On Adult, ENN produced only a modest decrease in the number of retained instances but preserved (and in some folds slightly improved) accuracy. This suggests that many removed points were local noise or borderline conflicts that were hurting vote stability. TCNN and ICF, in contrast, removed a much larger fraction of the training set — often keeping on the order of 10–30% of the original examples — which led to a noticeable speed-up at prediction time (fewer distance computations per query) with only a small drop in mean accuracy. In other words, Adult tolerates fairly aggressive pruning without collapsing.

--- Final Summary Table for Adult ---									
	Rank_Accuracy	Mean_Accuracy	Std_Accuracy	Rank_Time_ms	Mean_Time_ms	Std_Time_ms	Rank_Storage_percent	Mean_Storage_percent	Std_Storage_percent
algorithm									
HEOM_7_ModPlurality_NR_ENN	1.20	0.84	0.00	3.00	62.92	0.31	3.00	81.96	0.04
HEOM_7_ModPlurality_NR_None	1.90	0.83	0.00	4.00	88.07	1.08	4.00	100.00	0.00
HEOM_7_ModPlurality_NR_ICF	3.00	0.83	0.00	1.00	7.02	0.22	1.00	9.78	0.15
HEOM_7_ModPlurality_NR_TCNN	3.90	0.82	0.00	2.00	24.36	0.56	2.00	32.65	0.10

Fig 4. Summary Table for Adult

#### Pen-Based dataset.

On Pen-Based, the effect was even more pronounced. Because the classes are already very well separated in Euclidean space, condensed sets (especially ICF) could shrink the memory down to

roughly  $\sim 6\%$  of the original training points while still maintaining accuracy close to the unreduced baseline ( $\sim 0.99+$ ). The remaining accuracy loss was measurable but small compared to the order-of-magnitude gain in inference efficiency.

--- Final Summary Table for Pen-based ---									
algorithm	Rank_Accuracy	Mean_Accuracy	Std_Accuracy	Rank_Time_ms	Mean_Time_ms	Std_Time_ms	Rank_Storage_percent	Mean_Storage_percent	Std_Storage_percent
Euclidean_5_BordaCount_NR_None	1.15	0.99	0.00	3.00	11.70	0.05	4.00	100.00	0.00
Euclidean_5_BordaCount_NR_ENN	1.90	0.99	0.00	4.00	45.25	0.34	3.00	99.31	0.04
Euclidean_5_BordaCount_NR_TCNM	2.95	0.99	0.00	2.00	2.71	0.07	2.00	6.43	0.09
Euclidean_5_BordaCount_NR_ICF	4.00	0.98	0.00	1.00	2.52	0.07	1.00	6.01	0.18

Fig 5. Summary Table for Pen-based

## 3.2 SVM (Support Vector Machine)

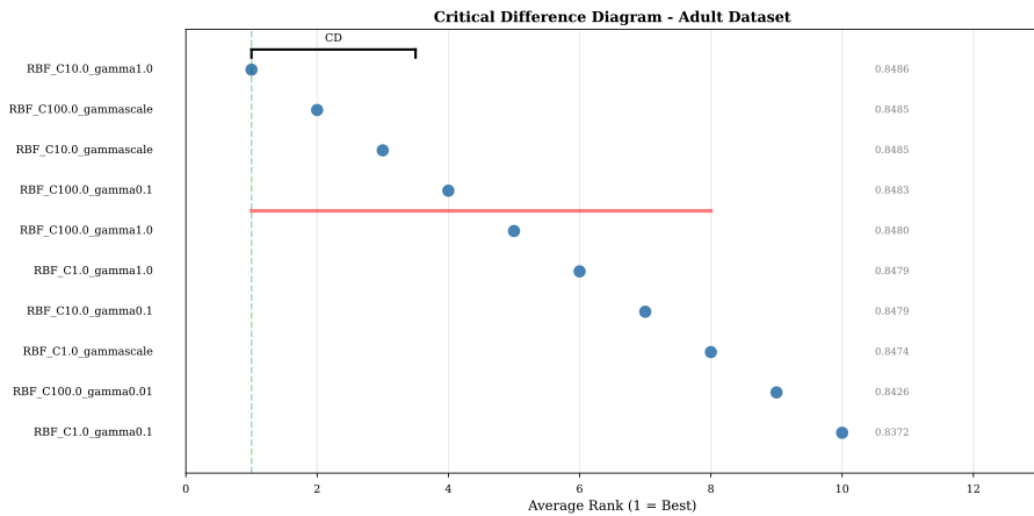
### 3.2.1 Adult dataset

We first compared the Linear and RBF kernels on the Adult dataset using 10-fold cross-validation. The RBF kernel consistently outperformed the Linear kernel, achieving a mean accuracy of 0.8474 compared to 0.8140, representing a 3.34 percentage improvement. This indicates that the Adult dataset contains non-linear relationships between features and class labels that the RBF kernel is able to model more effectively than the linear model. This tendency for the RBF kernel to perform better than the linear kernel is evident in *Figure 6*.

Kernel	Mean Accuracy	Std.	Mean Training Time (s)
RBF	0.8474	0.0034	14.92
Linear	0.8140	0.0021	12.09

Table 4: Comparison between RBF and Linear kernels

To identify the best parameter configuration, we evaluated 24 RBF SVM configurations across combinations of  $C$  and  $\gamma$  values. A Friedman test detected statistically significant performance differences among configurations ( $\chi^2 = 219.54$ ,  $p < 0.001$ ), and Nemenyi post-hoc tests revealed a subset of statistically equivalent top-performing configurations.



**Figure 6:** Critical Difference (CD) Diagram for the adult dataset. Configurations are ranked by mean accuracy (left = best). The Horizontal line connects configurations that are not significantly different ( $\alpha=0.05$ ). The Critical Distance (CD) bar indicates the minimum difference required for statistical significance. The numbers to the right show the mean accuracies across 10 folds.

Within this group, the configuration for the RBF kernel with  $C=10.0$ ,  $\gamma=1.0$ , forming the leftmost point in the diagram, achieved the highest mean accuracy ( $0.8486 \pm 0.0022$ ) and was therefore selected as the final SVM model for the Adult dataset. While several other configurations performed comparably, the selected model provided the best balance of accuracy and computational efficiency (mean training time:  $\sim 18.17$ s). Configurations with  $C = 100$  achieved similar accuracy but consistently incurred longer training times, while lower  $C$ -variants trailed slightly in performance.

This result confirms that incorporating a relatively strong margin penalty with a moderately narrow kernel width offers the optimal trade-off between model flexibility and generalization in modelling the complex decision boundary in the Adult dataset, while preventing underfitting.

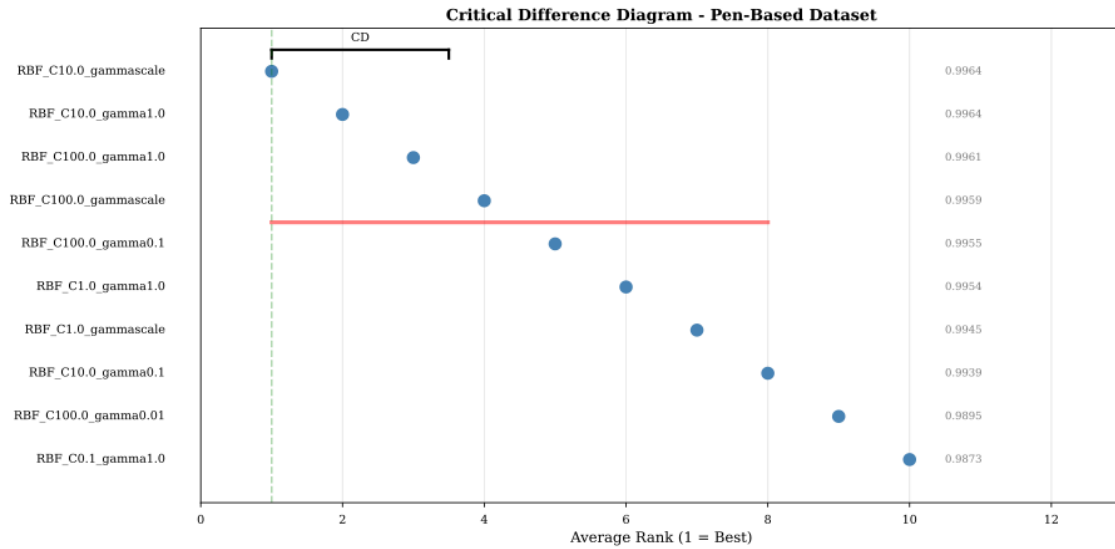
### 3.2.2 Pen-based dataset

For the Pen-Based handwritten digit dataset, the RBF kernel once again substantially outperformed the Linear kernel, with accuracies of 0.9945 and 0.9796, respectively. Although the increase in accuracy is reduced in absolute terms compared to the Adult dataset, it reflects the high separability and characteristic non-linear clusters of handwriting digit data.

Kernel	Mean Accuracy	Std.	Mean Training Time (s)
RBF	0.9945	0.0022	0.179
Linear	0.9796	0.0037	0.152

Table 5: Comparison between RBF and Linear kernels

The parameter tuning process again revealed statistically significant differences across configurations (Friedman  $p < 0.001$ ). The Nemenyi post-hoc comparisons identified multiple high-performing models, from which the RBF kernel with  $C=10.0$ ,  $\gamma=\text{scale}$  emerged as the best combination, offering the highest mean accuracy with minimal training time.



**Figure 7:** Critical Difference (CD) Diagram for the pen-based dataset. Configurations are ranked by mean accuracy (left = best). The Horizontal line connects configurations that are not significantly different ( $\alpha=0.05$ ). The Critical Distance (CD) bar indicates the minimum difference required for statistical significance. The numbers to the right show the mean accuracies across 10 folds.

Similar to the Adult results, several RBF configurations fall within the CD range of the best. However, the  $\gamma=\text{'scale'}$  setting consistently required less training time while maintaining accuracy statistically indistinguishable from the highest scores. Since  $\gamma=\text{'scale'}$  adapts to dataset variance, it provides a flexible kernel width that suits the morphological structure of the handwritten digit classes.

### 3.2.3 Impact of Instance Reduction

To assess whether instance reduction improves computational efficiency without degrading predictive performance, we applied three reduction techniques (ENN, ICF, and TCNN) to the training folds as a preprocessing step prior to SVM learning. Each reduced dataset was used to train the best SVM configuration determined previously (RBF kernel; Adult:  $C=10$ ,  $\gamma=1.0$ ; Pen-Based:  $C=10$ ,  $\gamma='scale'$ ), while testing was performed on the original full test folds to ensure a fair comparison. Performance was evaluated in terms of mean accuracy, training time, and the number of support vectors utilised. Statistical significance was assessed using one-sided Wilcoxon signed-rank tests, where  $p < 0.05$  indicates significant accuracy degradation relative to the baseline (all-data) SVM.

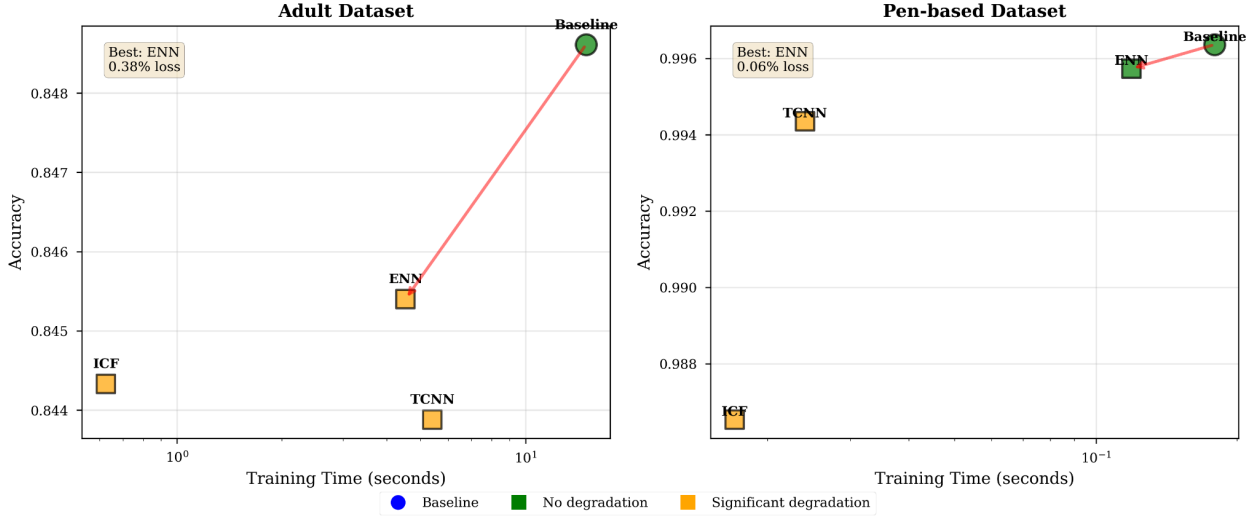


Figure 8: Accuracy versus training time (log scale) for baseline SVM and three IR methods on **Adult** (left) and **Pen-based** (right) datasets. Circle markers indicate baseline; square markers indicate the different IR methods tested. Colors denote: green (no significant degradation) and orange (significant degradation,  $p < 0.05$ ). The arrows highlight the recommended IR method. Text boxes summarize the best tradeoff in performance accuracy.

#### Adult Dataset

Results for the Adult dataset are shown in Figure 11 (on the left hand-side). All three reduction strategies led to some decrease in accuracy relative to the baseline (0.8486), however, the magnitude varied significantly across methods. The ENN method exhibited the smallest loss, achieving 0.8454 in accuracy ( $-0.38\%$ ), while reducing the average training time from 18.17s to 4.53s, corresponding to a  $4\times$  speedup. ICF produced a similar accuracy (0.8443) but achieved a far more substantial reduction in training time due to an extreme reduction in retained instances (and thus support vectors). TCNN provided a moderate speedup but incurred the largest accuracy loss among the three variants (0.8439).

The one-sided Wilcoxon tests indicated that the accuracy reduction for all three methods is statistically significant ( $p < 0.05$ ). Thus, even though the ENN technique offers the most favorable efficiency-accuracy trade-off, none of the reduction strategies fully preserve the baseline classifier's predictive performance on this dataset. The Adult dataset's class boundary is complex and distributed across a large number of informative instances. Hence, removing training instances, removes local decision support and leads to measurable and significant accuracy loss.

#### Pen-Based Dataset

For the Pen-Based dataset, the effect of instance reduction was markedly different (Figure 11, on the right hand-side). ENN achieved the best balance again, achieving 0.9957 accuracy (vs. baseline 0.9964), while improving efficiency by reducing the learned support vector set approximately by half. Importantly, the Wilcoxon one-sided test did not detect a significant accuracy difference ( $p \geq 0.05$ ), indicating that the slight numerical decrease in accuracy is not statistically meaningful. By contrast, ICF achieved the fastest training but resulted in a substantial accuracy drop (0.9865), which was

statistically significant. The TCNN technique preserved high accuracy (0.9944) and achieved a moderate speedup, but still showed statistically significant degradation.

Thus, only ENN preserves accuracy on the Pen-Based dataset while still improving efficiency, making it the most effective instance reduction technique for this dataset. The Pen-Based dataset exhibits highly compact intra-class clusters. ENN selectively removes locally inconsistent instances without removing core cluster structure, allowing the RBF SVM to maintain an almost identical decision boundary.

### Summary and Implications

Dataset	Best Reduction Method	Accuracy Change	Statistical Result	Conclusion
Adult	ENN	-0.38%	Significant loss ( $p < 0.05$ )	Efficiency improves, but accuracy cannot be preserved.
Pen-Based	ENN	-0.06%	Not significant loss ( $p \geq 0.05$ )	Accuracy preserved with meaningful efficiency gain.

Overall, the impact of instance reduction on SVM performance is dataset-dependent: For the Adult dataset, the decision boundary depends on numerous support regions; thus removing instances inevitably harms accuracy. For the Pen-Based dataset, decision regions are compact and well-defined; thus ENN removes redundancy rather than valuable information, preserving accuracy. This aligns with known behavior of SVMs under instance reduction: when class manifolds are clustered, reduction simplifies the optimization surface; when class structure is heterogeneous, reduction distorts it (Wilson & Martinez, 2000).

### 3.3 K-IBL vs SVM

To evaluate the relative predictive performance of SVM and k-instance-based learning, we compared each algorithm’s best-performing configuration using the paired Wilcoxon signed-rank tests across the same 10 cross-validation folds. This non-parametric paired test accounts for differences in fold difficulty and does not assume normality in the accuracy distributions, making it appropriate for a direct and statistically robust comparison between the two classifiers.

#### Adult Dataset

Figure 9 (left panel) illustrates the distribution of fold accuracies for SVM and k-IBL on the Adult dataset. The optimally tuned SVM (RBF kernel,  $C = 10.0$ ,  $\gamma = 1.0$ ) achieved a mean accuracy of  $0.8486 \pm 0.0022$ , whereas the best k-IBL configuration ( $K = 7$ , HEOM distance, modified plurality, no retention) obtained  $0.8329 \pm 0.0036$ , indicating a 1.57 percentage point decrease. The paired Wilcoxon test showed a statistically significant performance difference ( $W = 0.0$ ,  $p = 0.001953 < 0.05$ ), confirming that the SVM consistently outperformed k-IBL across all folds. The boxplot visualization reveals two key patterns: first, SVM’s median accuracy (red line) and mean accuracy (dashed line) are both substantially higher than k-IBL’s; second, the interquartile range for SVM is noticeably narrower, indicating more stable performance across different training splits with reduced sensitivity to fold composition.

The Adult dataset contains complex, heterogeneous decision regions influenced by multiple interacting attributes. SVM’s capacity to construct smooth non-linear decision boundaries via the RBF kernel enables it to generalize these structures more effectively, whereas k-IBL is limited by its strictly local decision mechanism, leading to systematically lower performance.

#### Pen-Based Dataset

The Pen-Based dataset results (right panel) show that both algorithms achieve very high accuracy, reflecting the compact and strongly clustered structure of handwritten digit classes. The best SVM

configuration ( $C = 10.0$ ,  $\gamma = \text{'scale'}$ ) achieved 0.9964, compared to 0.9943 for the best k-IBL configuration ( $K = 5$ , Euclidean distance, Borda voting). Although the absolute difference is small (0.21 percentage points), it is consistent across folds, and the Wilcoxon test again detected a significant difference ( $W = 3.0$ ,  $p = 0.009766 < 0.05$ ). Thus, SVM also significantly outperforms k-IBL on this dataset, although the effect magnitude is smaller overall.

Both algorithms successfully exploit the high class separability of this dataset, but it seems that the SVM benefits from the RBF kernel’s capacity to form smoothly curved decision surfaces that fit the geometry of the handwritten digit clusters. k-IBL remains competitive compared to SVM but exhibits slightly higher variability due to its dependence on local neighbourhood structure.

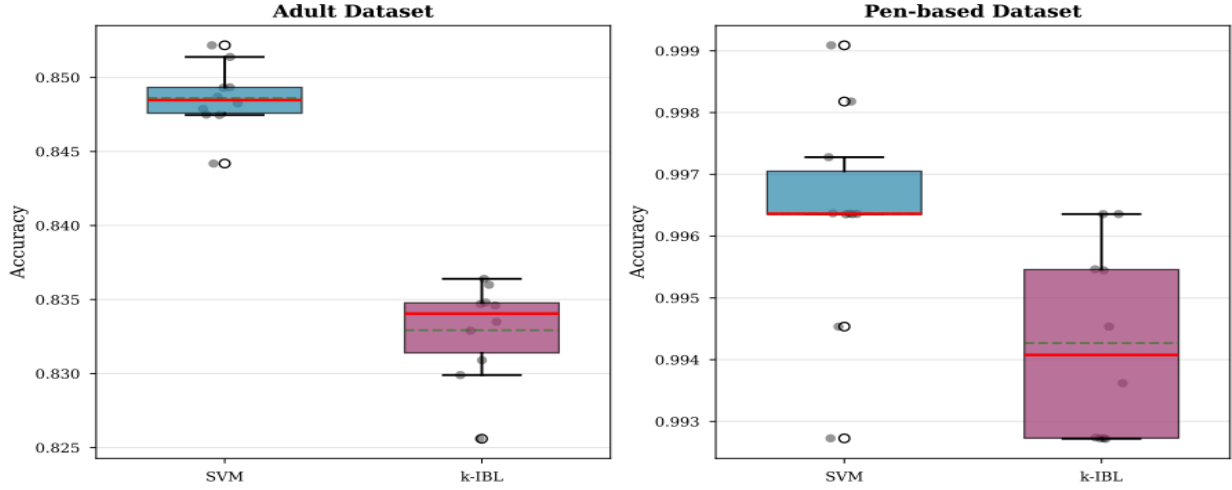


Figure 9: Box plots showing the distribution of 10-fold cross-validation accuracies for **Adult** (left) and **Pen-based** (right) datasets. Box boundaries represent 25th and 75th percentiles, red lines show medians, whiskers extend to  $1.5 \times IQR$ , and individual fold results are shown as scattered points. Green dashed lines describe the means.

### 3.3.3 Summary

Dataset	SVM Accuracy	k-IBL Accuracy	Difference	Wilcoxon p-value	Significance	Winner
Adult	0.8486	0.8329	+1.57%	0.001953	Yes	SVM
Pen-Based	0.9964	0.9943	+0.21%	0.009766	Yes	SVM

Across both datasets, SVM significantly outperforms k-IBL, though the magnitude of the performance gap reflects dataset characteristics: large on Adult (complex and heterogeneous), modest on Pen-Based (compact and separable). Critically, SVM also exhibits markedly lower variance in both cases, demonstrating superior stability. These results are consistent with the theoretical distinction between global margin-based decision functions (SVM) and local instance-driven classification (k-IBL).

## 3.4 Best K-IBL vs IR METHODS

### 3.4.1 Adult Dataset

Goal of this section to identify the best k-IBL configuration by jointly tuning: 1) distance function  $\in \{\text{Euclidean, Cosine, HEOM}\}$ , 2) number of neighbors  $k \in \{3, 5, 7\}$ , 3) voting rule  $\in \{\text{Modified Plurality, Borda Count}\}$ , 4) retention policy  $\in \{\text{NR, DC, DD, AR}\}$  and then to analyze how these design choices affect: 1) **Accuracy** (higher is better), 2) **Inference Time** (ms), 3) **Storage** (% of retained instances, lower is better). All feature weights are uniform ( $w_i = 1$ ). All results are obtained with 10-fold cross validation. Each fold is treated as an independent dataset for statistical testing. We run this analysis separately on each dataset (Adult, Pen-based), because their difficulty and scale are

very different. We then (i) pick the top configurations per metric, (ii) study trade-offs, and (iii) answer which configuration should be considered “the best k-IBL.”

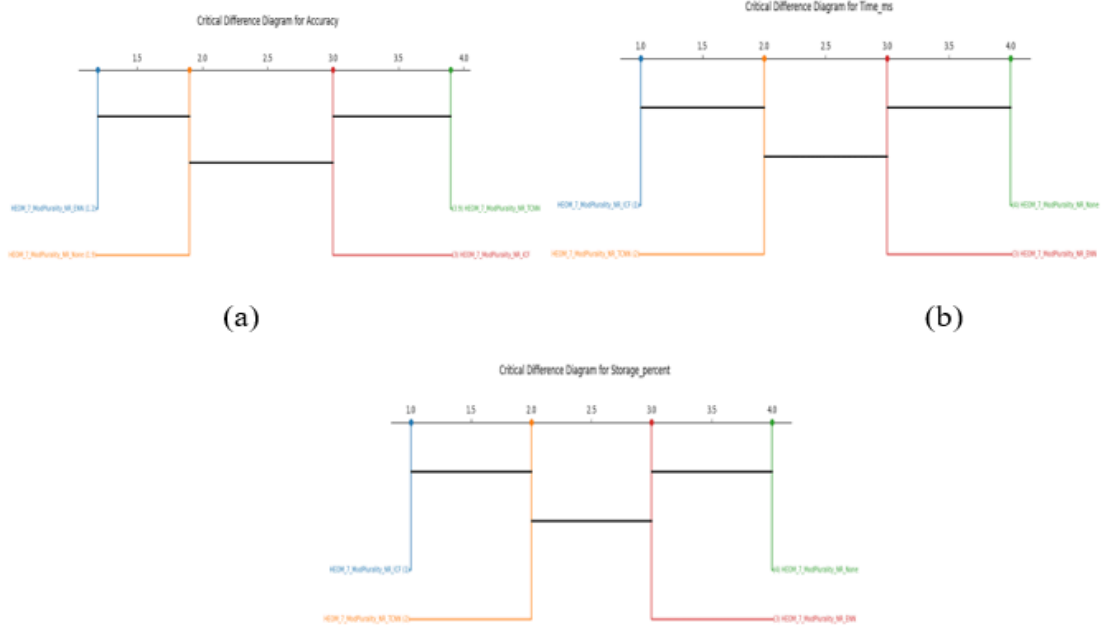


Fig 0. Critical Difference Diagram : a) Accuracy results, b) Prediction Time, c) Storage

Figure 1 combines the critical difference (CD) diagrams for accuracy, prediction time, and storage. Rankings are obtained via Friedman–Nemenyi; for time and storage we invert the ordering so that “lower is better” yields a better rank.

**Accuracy** (Fig. 10a). ENN attains the best average rank ( $\approx 1.2$ ) with a mean accuracy around 0.84. The unreduced baseline follows closely ( $\approx 1.9$ ;  $\sim 0.83$ ), while ICF and TCNN trail ( $\approx 3.0$  and  $\approx 3.9$ ). The CD diagram places ENN and the baseline in the same statistical neighborhood, indicating that ENN’s edge is modest rather than dramatic.

**Efficiency** (Fig. 10b–c). In contrast, the efficiency metrics show large, consistent gaps. ICF is the fastest method (rank  $\approx 1.0$ ;  $\sim 7$  ms/query) and also the most compact (rank  $\approx 1.0$ ;  $\sim 10\%$  of instances retained). TCNN occupies a clear middle ground ( $\approx 2.0$ ;  $\sim 24$  ms/query,  $\sim 33\%$  retained). ENN is only mildly selective ( $\sim 82\%$  retained) and correspondingly slower ( $\sim 63$  ms/query, rank  $\approx 3.0$ ). The baseline is consistently worst on both efficiency axes ( $\sim 88$  ms/query, 100% storage). The parallel between time and storage ranks underscores the mechanism: smaller case bases yield cheaper nearest-neighbor search. From these results we can say that Instance reduction is not required for accuracy—the baseline and ENN deliver similar predictive quality, with ENN only slightly ahead. It is crucial for efficiency: ICF (and, to a lesser extent, TCNN) delivers substantial speed and memory savings with a controlled accuracy trade-off. In practice, ENN suits accuracy-first scenarios with minimal behavioral change, whereas ICF is the method of choice when latency or memory footprint dominate; TCNN offers a balanced compromise.

```

--- Final Summary Table for Adult ---
=====

```

algorithm	Rank_Accuracy	Mean_Accuracy	Std_Accuracy	Rank_Time_ms	Mean_Time_ms	Std_Time_ms	Rank_Storage_percent	Mean_Storage_percent	Std_Storage_percent
HEOM_7_ModPlurality_NR_ENN	1.20	0.84	0.00	3.00	62.92	0.31	3.00	81.96	0.04
HEOM_7_ModPlurality_NR_None	1.90	0.83	0.00	4.00	88.07	1.08	4.00	100.00	0.00
HEOM_7_ModPlurality_NR_ICF	3.00	0.83	0.00	1.00	7.02	0.22	1.00	9.78	0.15
HEOM_7_ModPlurality_NR_TCNN	3.90	0.82	0.00	2.00	24.36	0.56	2.00	32.65	0.10

```

=====

```

Fig 11. Summary Table for Adult

The three metrics together reveal a structured trade-off:

- Accuracy: ENN ranks first in Accuracy and performs on par with the unreduced baseline. TCNN and ICF incur a small drop in Accuracy and are ranked lower in this dimension.
- Storage and Inference Time: ICF and TCNN dominate here. ICF, in particular, reduces storage from 100% to  $\sim 10\%$  and cuts prediction time from  $\sim 88$  ms/query to  $\sim 7$  ms/query, while still maintaining competitive (though slightly lower) accuracy.
- Intermediate behavior: ENN sits between “no reduction” and the more aggressive reducers: it removes some harmful/noisy points, preserves accuracy best, but does not substantially compress the dataset (still  $\sim 82\%$  of the original instances) and does not deliver the same inference acceleration as TCNN or ICF.

These observations indicate that there is no single universally best technique in an absolute sense. The “best” instance reduction method depends on the deployment objective:

- If the priority is maximizing predictive accuracy above all else, then ENN is preferred. It achieves the top accuracy rank (Fig.11) without statistically meaningful degradation compared to the full dataset.
- If the priority is minimizing memory footprint and test-time latency, then ICF is clearly the best choice. It is the top-ranked method for both Time and Storage (Fig.4), and its gains in efficiency are statistically significant according to the Friedman/Nemenyi analysis.
- TCNN offers an attractive middle ground: it retains roughly one third of the training set, delivers substantial speedups over the baseline, and its accuracy—while slightly below ENN/baseline—is still reasonably high.

In contrast, the unreduced baseline is dominated in efficiency (slowest, largest memory) and does not offer a decisive accuracy advantage over ENN. In other words, keeping *all* instances is not the most cost-effective design.

An additional practical question is whether it is “better” to improve the retention policy (i.e., how the system decides to keep or discard new cases over time) instead of reducing the training set up front. In our experiments, the retention policy was held fixed (NR) across all variants. The only change was the instance reduction method used to select which training examples are stored. Even under this fixed retention strategy, algorithms like ICF and TCNN were able to:

- shrink the case base from 100% down to  $\approx 10\text{--}30\%$ ,
- cut inference time by an order of magnitude,
- and maintain accuracy within a relatively small margin of the baseline.

These results suggest that a large fraction of the original training set is redundant for prediction. This redundancy is removed explicitly by instance reduction. It is unlikely that tuning retention alone (without any up-front screening of the case base) would achieve the same  $10\times$  improvement in query time or the  $\sim 90\%$  reduction in memory footprint. Therefore, our evidence indicates that instance reduction is not only helpful, but in practice essential if we care about efficiency.

### 3.4.2 Pen-based Dataset

We evaluate the impact of instance reduction on the k-IBL classifier using the pen-based dataset. All models share the same hyperparameters: Euclidean distance,  $k = 5$  neighbors, BordaCount voting, and NR as the retention policy. This configuration is the best-performing setup for k-IBL on pen-based (without reduction), so fixing it allows us to isolate the impact of the instance reduction method itself. The only aspect we vary is how (and whether) we reduce the training instances before storing them. All results are obtained with 10-fold cross validation. Each fold is treated as an independent dataset for statistical testing.

Figure 1 consolidates the critical difference (CD) diagrams for accuracy, inference time, and storage on the Pen-based dataset. Rankings are computed with the Friedman–Nemenyi procedure; for time and storage we invert the scale so that smaller values correspond to better ranks. This unified view lets us read performance and efficiency trade-offs side by side.

First of all, all methods are extremely accurate ( $\approx 0.98$ – $0.99$ ) (Fig 4). The Baseline (no reduction) attains the best average rank (1.15) and ENN is statistically in the same top group (1.90). TCNN follows (2.95), while ICF ranks last (4.00). Thus, for Pen-based, keeping the full training set delivers the highest accuracy, and lightly editing it with ENN preserves essentially the same performance; more aggressive reduction (TCNN, ICF) yields a small but systematic drop. For Inference time (Fig. 12b, Fig 13). Runtime differences are large and statistically significant. ICF is fastest (rank 1.00;  $\approx 2$ – $3$  ms/query), TCNN is second (rank 2.00;  $\approx 2$ – $3$  ms), the Baseline is slower (rank 3.00;  $\approx 11.7$  ms), and ENN is slowest (rank 4.00;  $\approx 45.3$  ms). Despite being an editing method, ENN retains nearly all instances and therefore does not improve latency; in practice it is slower than the unreduced baseline.

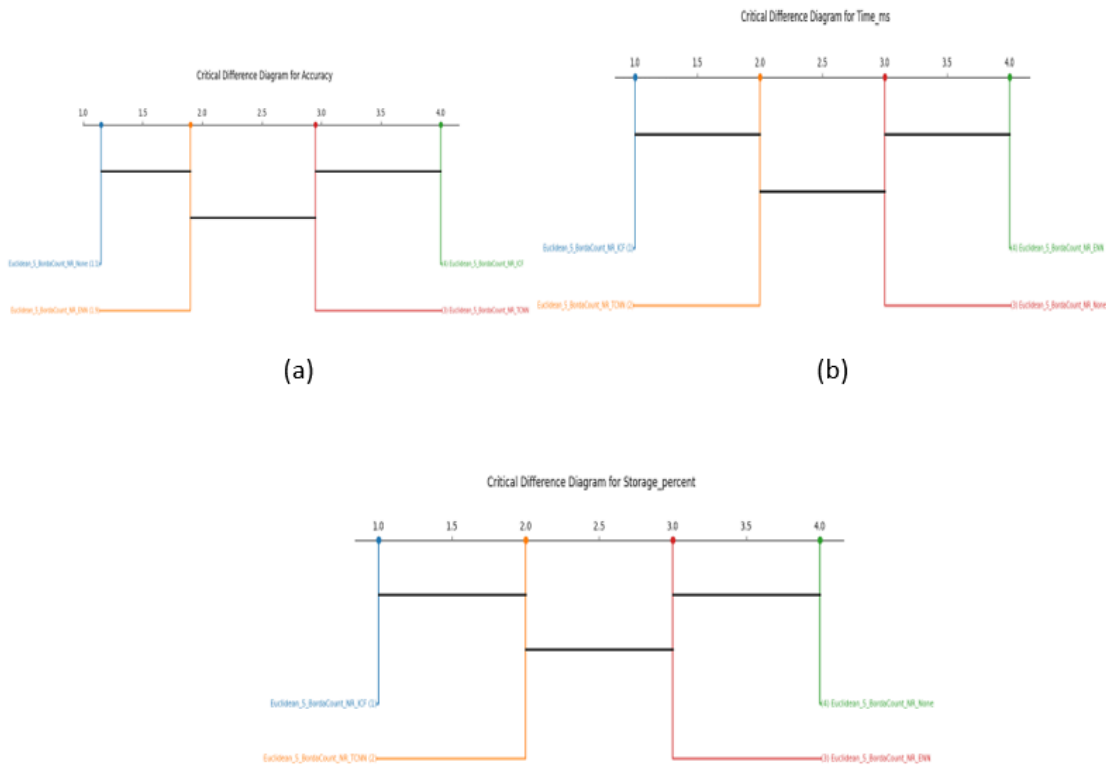


Fig 12. Critical Difference Diagrams for Pen-based: (a) Accuracy, (b) Inference Time, (c) Storage.

Storage (Fig. 12c). Storage ranks mirror time ( $\chi^2_F = 30.00$ ,  $p \approx 1 \times 10^{-6}$ ). ICF achieves the best compression (rank 1.00; stores  $\sim 6.0\%$  of the set), TCNN is next (rank 2.00;  $\sim 6.4\%$ ), ENN is only mildly selective (rank 3.00;  $\sim 99\%$  kept), and the Baseline retains 100% (rank 4.00). The parallel between time and storage confirms the mechanism: smaller case bases yield cheaper nearest-neighbor search. Takeaway. For Pen-based, accuracy is already near-ceiling; Baseline and ENN tie at the top. If efficiency matters, ICF (and TCNN) deliver order-of-magnitude improvements in latency and memory at a negligible absolute accuracy loss ( $\sim 0.01$ ). In practice: use Baseline/ENN for accuracy-first scenarios with minimal change, ICF when speed/memory dominate, and TCNN as a balanced compromise.

```

--- Final Summary Table for Pen-based ---
=====

```

algorithm	Rank_Accuracy	Mean_Accuracy	Std_Accuracy	Rank_Time_ms	Mean_Time_ms	Std_Time_ms	Rank_Storage_percent	Mean_Storage_percent	Std_Storage_percent
Euclidean_5_BordaCount_MR_None	1.15	0.99	0.00	3.00	11.70	0.05	4.00	100.00	0.00
Euclidean_5_BordaCount_MR_ENN	1.90	0.99	0.00	4.00	45.25	0.34	3.00	99.31	0.04
Euclidean_5_BordaCount_MR_TCNN	2.95	0.99	0.00	2.00	2.71	0.07	2.00	6.43	0.09
Euclidean_5_BordaCount_MR_ICF	4.00	0.98	0.00	1.00	2.52	0.07	1.00	6.01	0.18

```

=====

```

Fig 13. Summary Table for Pen-based

Fig 13 puts all three metrics side by side (Accuracy rank, Time rank, Storage rank), and the trade-off is clear. These observations indicate that there is no single universally best method. The appropriate instance reduction strategy depends on deployment goals:

- If the only priority is maximum predictive accuracy, then reduction is not strictly necessary. The unreduced Baseline (or ENN, which is statistically tied in accuracy) is sufficient.
- If the priority is fast inference and minimal memory footprint—for example, on-device classification or latency-sensitive prediction—then ICF is clearly the best choice, with TCNN as a competitive second.

In the light of these results we can say that Instance reduction clearly helps from a systems perspective. ICF and TCNN cut memory usage to about 6% of the original case base and reduced per-query time from  $\sim 11.7$  ms to  $\sim 2\text{--}3$  ms, at the cost of only a marginal accuracy drop ( $\approx 0.99 \rightarrow \approx 0.98$ ). These differences are statistically significant under the Friedman/Nemenyi tests.

Finally, note that these conclusions are dataset-dependent. On pen-based, the task is already extremely easy ( $\approx 0.99$  Accuracy for almost all methods), so ENN cannot “clean” the data enough to improve accuracy. In a noisier dataset such as Adult, however, editing and reduction can also influence accuracy itself. This suggests that for k-IBL, instance reduction should be chosen with respect to both the deployment objective (accuracy vs. efficiency) and the structure of the dataset.

## 4 Conclusion

This work systematically compared lazy instance-based learning (k-IBL) and margin-based classifiers (SVMs) across two datasets with very different structures — Adult (heterogeneous, noisy, mixed numeric/categorical features and moderate class imbalance) and Pen-Based (clean, fully numeric, highly separable 10-class handwriting data). The study explored (i) how k-IBL behaves under different design choices (distance metric, neighborhood size, voting rule, and retention policy), (ii) whether feature weighting and instance reduction can improve either accuracy or efficiency, and (iii) how an optimized k-IBL model compares to a tuned SVM, both statistically and computationally.

For k-IBL, we observed that there is no single universally optimal configuration. On Adult, the best-performing variant combined HEOM distance with  $k = 7$  neighbors, Modified Plurality voting, and a no-retention policy. This combination is well aligned with the nature of Adult: HEOM is explicitly designed for mixed-type data and prevents any single numeric attribute from dominating the distance, while a larger neighborhood and plurality-style voting help smooth over noise and mislabeled points. In contrast, on Pen-Based, a far simpler model — Euclidean distance,  $k = 5$ , and Borda Count voting — already achieved  $\approx 0.994$  mean accuracy with low inference time. This indicates that, when the class structure is already geometrically well separated in Euclidean space (as in handwritten digits), more complex similarity metrics and larger neighborhoods offer little to no benefit.

We then evaluated whether preprocessing could push k-IBL further. Feature weighting based on Mutual Information produced a statistically significant accuracy gain on Adult ( $\approx +0.31$  percentage points over the baseline), especially when weights were learned from the full training fold. This suggests that globally up-weighting informative categorical attributes helps in heterogeneous, noisy problems. The same effect did not appear on Pen-Based: the baseline was already near perfect, and neither Mutual Information nor ReliefF produced a meaningful improvement. In other words, feature weighting is useful when attributes have unequal and partly redundant predictive power (Adult), but mostly irrelevant when the raw geometry of the feature space is already highly discriminative (Pen-Based). Subsampling to estimate weights offered a faster but slightly less reliable alternative, which is attractive in practice when full-fold weighting is computationally expensive.

Instance reduction was also studied in two roles: as a preprocessing step for k-IBL, and as a preprocessing step for SVM. For k-IBL, Edited Nearest Neighbor (ENN), Tomek/Condensed variants (TCNN), and aggressive condensed filters such as ICF offered different trade-offs. ENN tended to preserve or even slightly improve accuracy relative to keeping all instances, but only modestly reduced the size of the case base. By contrast, ICF and TCNN aggressively compressed the memory

footprint (down to  $\sim 10\text{--}30\%$  on Adult and  $\sim 6\%$  on Pen-Based) and reduced per-query time by up to an order of magnitude, at the cost of only a small but consistent drop in accuracy. This shows that instance reduction is not just “nice to have,” but in many deployment scenarios (on-device inference, strict latency budgets, memory limits) it is essential: keeping 100% of the training set is rarely the most cost-effective choice.

When comparing k-IBL to SVM directly, the tuned SVM (RBF kernel with selected  $C$  and  $\gamma$ ) significantly outperformed the best k-IBL configuration on both datasets under paired Wilcoxon signed-rank tests. On Adult, SVM achieved roughly 1.6 percentage points higher mean accuracy and also exhibited lower variance across folds, indicating more stable generalization. On Pen-Based, both models were extremely strong ( $\geq 0.99$  accuracy for SVM,  $\sim 0.994$  for k-IBL), but SVM still retained a statistically significant edge. These findings are consistent with the theoretical difference between the two paradigms: SVM learns a global, smooth decision surface that can capture complex nonlinear boundaries (especially with an RBF kernel), whereas k-IBL makes local, memory-based decisions and is therefore more sensitive to noise, distance scaling, and redundancy in the stored instances.

Practically, our results support the following deployment guidelines. For heterogeneous, noisy, mixed-type data like Adult, (i) an RBF SVM with tuned  $C$  and  $\gamma$  should be the default choice if accuracy is the primary objective, and (ii) if a lazy learner must be used, then HEOM with higher  $k$ , Modified Plurality voting, and Mutual-Information-based feature weighting is preferred. For clean, well-clustered numeric data like Pen-Based, both SVM and a well-tuned Euclidean k-IBL are viable, and the decision may be driven more by latency and memory constraints than by raw accuracy. In such latency-critical settings, applying an aggressive instance reduction method (e.g. ICF) before deployment can cut inference time by  $\sim 5\text{--}10\times$  with only a marginal accuracy drop, making nearest-neighbour style classifiers much more practical.

Finally, this analysis highlights two directions for future work. First, we treated instance reduction and retention policy as separate knobs. A combined design — e.g. learning a compact core set up front (ICF/TCNN) and then applying an informed retention strategy online — could potentially preserve accuracy while keeping memory permanently low. Second, we did not explore approximate nearest neighbour search structures or GPU-accelerated distance computation; integrating those could further narrow the efficiency gap between lazy methods and parametric models. Overall, however, our evidence is clear: while k-IBL can be engineered to be competitive and efficient with the right metric, voting rule, and reduction strategy, a properly tuned SVM remains the most accurate and most stable performer across both datasets considered in this study.

## 5 References

- Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). *A practical guide to support vector classification*. Technical report, Department of Computer Science, National Taiwan University.
- Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3), 131-159.
- Demšar, J. (2006). "Statistical Comparisons of Classifiers over Multiple Data Sets." *Journal of Machine Learning Research*, 7, 1-30.
- Wilson, D.R. and Martinez, T.R., 2000. *Reduction techniques for instance-based learning algorithms*. *Machine Learning*, 38(3), pp.257–286.
- Scikit-Learn Developers. (2025). mutual\_info\_classif API documentation. Retrieved from the Scikit-Learn User Guide.
- Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning Journal*, 53(1), 23–69.
- ReliefF algorithm – The Weka documentation for ReliefFAttributeEval
- Statistical tests – Demšar’s influential JMLR paper on classifier comparisons ([available online](#)) recommends the Wilcoxon signed-rank test for pairwise comparisons of two classifiers and the Friedman test for multiple classifiers.