Introduction to Machine Learning: Work 3

Pedro Agúndez[*], Bruno Sánchez[*], María del Carmen Ramírez[*], Antonio Lobo[*]

December 8, 2024

Abstract

Abstract

[*] Universitat de Barcelona
pedro.agundez@estudiantat.upc.edu
bruno.sanchez.gomez@estudiantat.upc.edu
maria.del.carmen.ramirez@estudiantat.upc.edu
antonio.lobo@estudiantat.upc.edu

1 Introduction

Introduction.

2 Methodology

Introduction to methodology.

2.1 K-Means

K-Means intro.

2.1.1 Hyperparameters

1. **k:**
    - The number of clusters to partition the dataset. Determines the complexity and granularity of the clustering.
2. **Distance Metrics:**
    - **Euclidean Distance:** Calculates the root of the sum of squared differences between feature values. Standard metric for continuous data:

      $$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

    - **Manhattan Distance:** Computes the sum of absolute differences between feature values, suitable for both categorical and continuous data:

      $$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

    - **Clark Distance:** Accounts for proportional differences between feature values, enhancing interpretability for attributes with varying scales:

      $$d(x, y) = \sqrt{\sum_{i=1}^{n} \left(\frac{|x_i - y_i|}{x_i + y_i + \epsilon} \right)^2}$$

      where ϵ is a small constant to avoid division by zero.
3. **Additional Parameters:**
    - **Initial Centroids:** Pre-defined initial cluster centers used as the starting point for the clustering algorithm.
    - **Maximum Iterations:** Limits the number of iterations to prevent excessive computational time, with a default of 10 iterations.

2.1.2 Clustering Methodology

- **Clustering Process:**
    1. Assign each data point to the nearest centroid using the specified distance metric.
    2. Recalculate centroids by computing the mean of all points in each cluster.
    3. Repeat assignment and recalculation until convergence or maximum iterations are reached.
- **Convergence Criteria:**

  - Clusters are considered stable when centroids no longer significantly change between iterations.

- **Variance Computation:**
  - Total within-cluster variance (E) is calculated by summing squared distances of points to their respective cluster centroids.
  - Provides a measure of clustering compactness and quality.

This methodology allows for flexible clustering configurations, enabling analysis across different datasets and hyperparameter values.

2.2 Global K-Means

Out of the proposed improvements to the K-Means algorithm, the first we chose to implement was the **Global K-Means** algorithm [3], which focuses on following a deterministic and systematic approach to "optimal" centroid initialization and cluster formation. Additionally, we have also implemented the improvements to the Global K-Means algorithm itself, proposed in the original article by Likas et al.: *Fast Global K-Means*, and *Initialization with k-d Trees.* By addressing the limitations of traditional K-Means, this enhanced methodology introduces novel strategies, including PCA-based data partitioning and iterative error-reduction mechanisms, to improve both accuracy and computational efficiency.

This section outlines the hyperparameter configurations and clustering methodology adopted for the Global K-Means algorithm, which was implemented in the `GlobalKMeansAlgorithm` class.

2.2.1 Hyperparameters

We consider the same hyperparameters as for the standard K-Means algorithm (Section 2.1), except for 2 significant modifications:

1. **Initial Centroids:**
   - Global K-Means no longer accepts a collection of initial centroids as a hyperparameter, since the goal of this algorithm is rooted in the deterministic calculation of the "best possible" centroids, which substitutes their random initialization.
2. **Number of Buckets:**
   - Controls initial data partitioning, by defining the number of candidate points that we will consider as possible centroids throughout the algorithm.
   - Its default value is $2 \cdot k$, but we also test values $3 \cdot k$ and $4 \cdot k$.

2.2.2 Clustering Methodology

- **Initialization with k-d Trees:**
  1. Use k-d tree partitioning based on Principal Component Analysis (PCA).
  2. Recursively partition data samples into buckets.
  3. Select candidate points based on bucket centroids.
- **Fast Global K-Means Algorithm:**
  1. Initialize first centroid as dataset mean.
  2. Iteratively add centroids by:
     - For each $k' = 2, \dots, k$, we already have $k' - 1$ centroids.
     - Compute guaranteed error reduction for candidate points with respect to the $k' - 1$ centroids,

$$b_n = \sum_{j=1}^{N} \max\left(d_{k'-1}^{j} - ||x_n - x_j||^2, 0\right) ,$$

where $d_{k'-1}^{j}$ is the squared distance between x_j and the closest centroid among the $k' - 1$ obtained so far. The pair-wise squared distances between points are precomputed at the start.

    - Select point with maximum guaranteed error reduction.
    - Run k'-means with the $k'-1$ centroids plus the selected point, unitl convergence.

3. Repeat until k clusters are formed.

This methodology provides a sophisticated approach to centroid initialization and clustering, leveraging PCA-based partitioning and error reduction strategies in order to achieve an improvement in consistency and speed with respect to the base K-Means algorithm.

2.3 Fuzzy C-Means

We selected the **generalized suppressed Fuzzy C-Means** (gs-FCM) algorithm, an improvement over traditional FCM, which often shows multimodal behavior near cluster boundaries (Fig. 1a). This issue, where fuzzy memberships remain high for unrelated clusters, was addressed by Höppner and Klawonn [2].

The suppressed Fuzzy C-Means (s-FCM) algorithm [1] enhances convergence speed and classification accuracy without minimizing the traditional objective function J_{FCM}. It introduces a suppression step during each iteration to reduce non-winner fuzzy memberships, which is mathematically equivalent to virtually reducing the distance to the winning cluster's prototype (Fig. 1b) [5].

Szilágyi et al. [5] defined the quasi-learning rate η of s-FCM, analogous to learning rates in competitive algorithms:

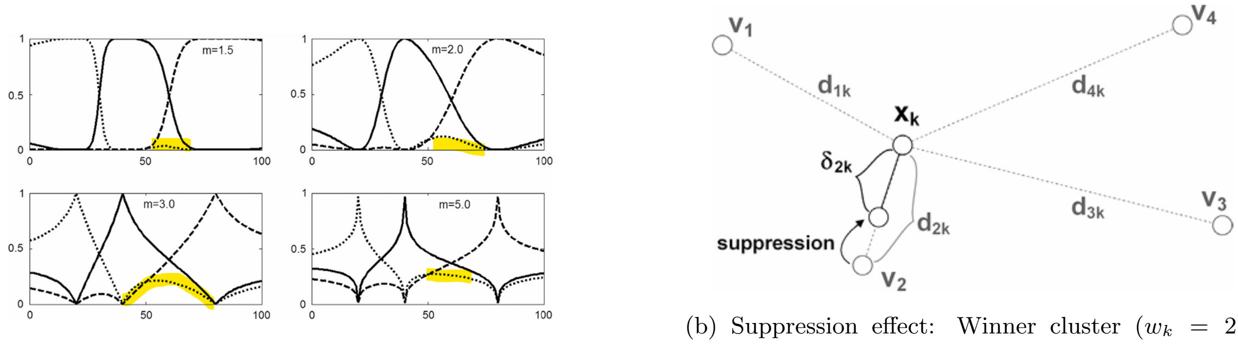
$$\eta(m, \alpha, u_{wk}) = 1 - \frac{\delta_{wk}}{d_{wk}} = 1 - \left(1 + \frac{1-\alpha}{\alpha u_{wk}}\right)^{(1-m)/2}.$$

Building on this, gs-FCM modifies the learning rate to decay linearly with increasing winner membership u_{wk} for faster convergence, as proposed in sg_ρ-FCM [4]:

$$\eta(u_{wk}) = 1 - \rho u_{wk}, \quad \text{where } 0 \leq \rho \leq 1.$$

This approach ensures a logical adaptation of membership weighting, expressed as:

$$\alpha_k = \left[1 - u_w + u_w\left(1 - f(u_w)\right)^{2/(1-m)}\right]^{-(1-m)}.$$



(a) Multimodal fuzzy memberships near cluster boundaries for varying fuzzy exponent m.

(b) Suppression effect: Winner cluster ($w_k = 2$) gains increased membership while non-winners are suppressed.

Figure 1: Figures adapted from [4].

3 Results

To systematically evaluate the different configurations of each clustering algorithm, the following procedure is followed to extract results for each of the 3 data sets, in order to later perform an statistical analysis:

1. **Data Preparation**: The dataset is loaded and the data samples are separated from their labels into separate 2 sets. This way, we perform the clustering analysis in a completely non-supervised way, and we then utilize the labels to extract supervised metrics of the cultering results.

2. **Parameter Configuration**: A comprehensive set of values for the algorithm's hyperparameters is defined. These combinations reflect various ways to tune the clustering algorithm.

3. **Model Evaluation**: For each parameter combination, the clustering algorithm is applied on the unlabeled data and then evaluated with different metrics. This step yields the following metrics: total cluster variance (E), Adjusted Rand Score (ARI), F1-score, Davies-Bouldin Index(DBI), Silhouette score, Calinski-Harabasz score, and execution time. Together, these metrics (some supervised, some non-supervised) measure the effectiveness and efficiency of the clustering.

   Note: Not all clustering algorithms are based on centroids, hence the total cluster variance is not computed for those which are not.

4. **Results Compilation**: The performance metrics for each parameter combination are recorded in a structured format. These results are saved as a dataset that summarizes the outcomes of all evaluations, forming a basis for analysis.

5. **Statistical Analysis**: After results are compiled across all configurations, statistical analysis is performed to identify the best-performing configurations. This analysis helps determine the most reliable and effective parameter settings for accurate and efficient clustering. We will discuss our results in the following sections.

3.1 K-Means

We have tested on each dataset 57 different configurations of the K-Means algorithm, by using the 3 different distance metrics with 19 different values of the k (from 2 to 20). For each of these configurations, we have run the K-Means 10 times, in order to account for the randomness of the centroid initialization. This results in a total of 570 runs of the K-Means algorithm for each dataset. From the evaluation metrics extracted for each of these runs, we study the effect of each of the 2 hyperparameters and achieve conclusions about them through statistical analysis.

3.1.1 Preliminary Study

Before starting with the more rigorous statistical analysis, let us first observe some preliminary patterns about the measured metrics and the effect of each hyperparameter on the clustering performance.

In Figure 2 we summarize the relationship between the different metrics that were measured. It is a matrix plot where the lower triangle is a heatmap of the Pearson correlations between each pair of metrics, the diagonal elements are the histogram distributions of values of each metric, and the upper triangular has for each pair of metrics the plot of their values for all runs. It is interesting to observe that, while we would expect all of the metrics to agree on the identified trends, there are some cases where the opposite behaviour is displayed. An example of this is the negative correlation between E (total variance) and DBI (Davies-Bouldin Index): since the DBI measures cluster compactness, we would expect it to directly correlate to E; however, we observe that there is a negative correlation between them. This specific figure was extracted from the results on the Mushroom dataset, but the conclusions are the same for the others (the plots can be found in the `code` floder).

Parallelly, a different set of interesting relationship are displayed in Figure 3, where we can see heatmaps of the F1 Score and the Time across the different pairwise hyperparameter configurations. We can observe a general trend regarding execution time: it seems to have considerably larger values for the Clark distance metric compared to those of the other 2, which reflects the higher computational cost that this distance metric has. Additionally, we see a noteworthy divergence in the F1 Score trends with respect to the values of k: in the Mushroom dataset (which has 2 classes), lower values of k seem to achieve a better F1 Score; meanwhile, in the Pen-based dataset (which has 10 classes), intermediate values of k (between 7 and 11) seem to achieve the best scores. This was to be expected, yet it still is compelling to see it reflected so clearly in the results.

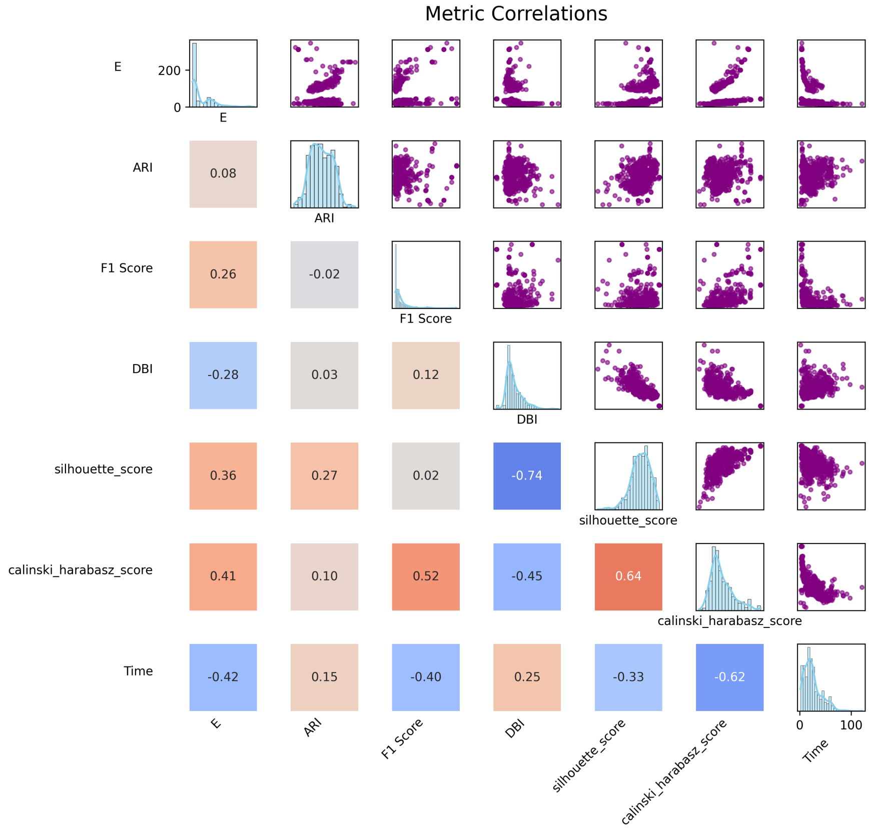
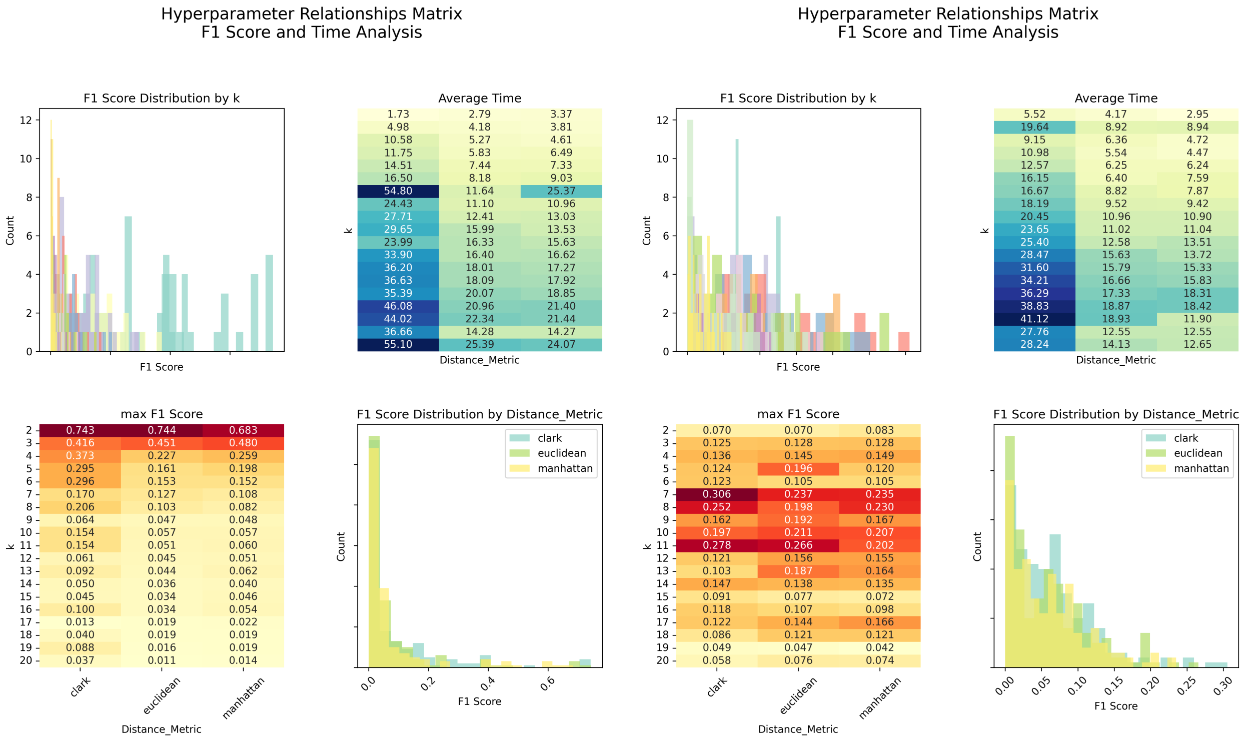


Figure 2: Metrics correlations summary



(a) Mushroom pairplot matrix

(b) Pen-based pairplot matrix

Figure 3: Hyperparameter pairplot matrices based on F1 Score and Time

3.1.2 Statistical Analysis

For the statistical analysis, we have first performed Friedman tests to determine if there are significant differences in each of the metrics among the different possible values of each hyperparameter. With a confidence level of $\alpha = 0.15$, all of the metrics agree across all 3 datasets that **there are significant differences** when varying across values of k, and also when changing the distance metric.

Note: Due to the vast amount of statistical results extracted, not all of them will be displayed; only the most relevant ones will be explicitly highlighted. All of the results can be found in the `code` folder, inside `Analysis/plot_and_tables/KMeansReports`, for each dataset.

Since we have found significant differences between different values of the hyperparameters, we have then applied post-hoc tests on all of the metrics to find more specifically where these differences lie:

- For the k, we have performed Nemenyi tests on each of the 3 datasets.

  1. **Hepatitis:** Most of the metrics agree that the lower values of k show better clustering performance. We observe this in the grouped metric averages, which generally get worse as the k increases. The only exception is the DBI, for which the differences are mostly not significant. For the rest of metrics, k=2 always shows significant differences when compared with k=19 and k=20, for instance. These results make sense, since the Hepatitis dataset has 2 classes.
  2. **Mushroom:** In this case, there are some metrics that cannot find any case of pairwise significant differences. The two that find the most relevant differences are F1 Score and Calinski-Harabasz score, which in both cases we can see that values of k from 2 to 4 show significantly better clustering performance than values from 12 to 20. Once again, since the Mushroom dataset also has 2 classes, these conclusions were to be expected.
  3. **Pen-based:** For this dataset, most of the metrics do not identify significant differences among values of k. The only 2 exceptions are ARI and Calinski-Harabasz score: the ARI identifies values 2 through 5 to be significantly worse than those greater than 10, while there are no differences among high values of k; meanwhile, the Calinski-Harabasz score seems to find significant differences between the lowest values (below 5) and the highest values (above 14), favoring the small values. Additionally, by studying the F1 Score averages, we see a clear trend of better performance between values 7 and 10 (though statistically these differences are not meaningful). Overall, these results are not very conclusive, and we only deduce that the K-Means might not be the most appropiate clustering algorithm for the Pen-based dataset.

- For the distance metric, we decided on applying Bonferroni tests instead of Nemenyi, considering the Euclidean distance to be the "control" value.

  1. **Hepatitis:**
  2. **Mushroom:**
  3. **Pen-based:**

4 Conclsuion

Conclusion.

References

[1] J.L. Fan, W.Z. Zhen, and W.X. Xie. Suppressed fuzzy c-means clustering algorithm. Pattern Recognition Letters, 24:1607–1612, 2003.

[2] F. Höppner and F. Klawonn. Improved fuzzy partitions for fuzzy regression models. International Journal of Approximate Reasoning, 32:85–102, 2003.

[3] Aristidis Likas, Nikos Vlassis, and Jakob J. Verbeek. The global k-means clustering algorithm. Pattern Recognition, 36(2):451–461, 2003.

[4] László Szilágyi and Sándor M. Szilágyi. Generalization rules for the suppressed fuzzy c-means clustering algorithm. Neurocomputing, 139:298–309, 2014.

[5] L. Szilágyi, S.M. Szilágyi, and Z. Benyó. Analytical and numerical evaluation of the suppressed fuzzy c-means algorithm: a study on the competition in c-means clustering models. Soft Computing, 14:495–505, 2010.