ECF-means – Ensemble Clustering Fuzzification Means

A novel algorithm for clustering aggregation, fuzzification, and optimization

Gaetano Zazzaro, Angelo Martone
CIRA
Italian Aerospace Research Centre
Capua (CE), Italy
e-mail: {g.zazzaro, a.martone}@cira.it

***Abstract*—This paper describes a clustering optimization algorithm for Data Mining, called Ensemble Clustering Fuzzification (ECF) means, which combines many different clustering results in ensemble, achieved by *N* different runs of a chosen algorithm, into a single final clustering configuration. Furthermore, ECF is a simple procedure to fuzzify a clustering algorithm because each point in the original dataset is assigned to each cluster with a degree of membership. Moreover, a novel clustering validation index, called Threshold Index (TI), is also here defined. The proposed approach is applied to the well-known *k*-means clustering algorithm by using its Weka implementation and an ad-hoc developed software application. Two case studies are also here reported; the first one in the meteorological domain and the second one concerns the famous Iris dataset. All the outcomes are compared with the results of the simple *k*-means algorithm against which ECF seems to provide more effective and usable final configurations.**

Keywords-Clustering Optimization; Data Mining; Ensemble Clustering; Fuzzy Clustering; k-means; Weka.

I. INTRODUCTION

Clustering (or cluster analysis) is an unsupervised Machine Learning technique of finding patterns in the data. It is widely used [1] for Data Mining tasks, because it can be easily applied to understand, explore, prepare, and model data. It plays an outstanding role in many applications, such as scientific data exploration, information retrieval and text mining, web analysis, bioinformatics, and many others.

In the literature, there are many categories of algorithms for clustering: Heuristic-based, Model-based, Density-based [2]. Their common goal is to create clusters so that objects in the same cluster should be as similar as possible, whereas objects in one cluster should be as dissimilar as possible from objects in the other clusters. Usually, it is not easy to choose the most useful algorithmic approach, the most satisfying result, and therefore the most usable configuration. In fact, the different models for clustering may produce groupings that are very different from one another. Anyone applying a clustering algorithm immediately realizes how difficult it is to choose the final cluster configuration. We may have different results because we choose different algorithms, or different parameters of the fixed algorithm. Furthermore, the numerous available evaluation metrics often do not facilitate this choice because they lead to very discordant results.

In spite of the availability of a large number of validation criteria, the ability to truly test the quality of a final configuration remains vague and hard to achieve. Specific domain knowledge is not an aid because it is often hard to translate it into operating rules, neither the domain expert has a real target class for evaluating and comparing the results. So, why do not consider all the obtained configurations? That is, why do not find a method that summarizes all the results of clusterings? Meta-learning ensemble methods may be an answer. The idea is that no single model or criterion truly captures the optimal clustering, but a cooperation of models could provide a more robust solution. Cluster Ensemble, or Aggregation Clustering, or Multiview Clustering, aims to find a single clustering from multi-source basic clusterings on the same group of data objects [3]. However, these ensemble methods, such as voting-based clustering [4], consensus clustering [5], or clustering aggregation [6] do not assign a level of membership to every point in clusters.

In order to overcome the limits mentioned above, in this paper we present a strategy for cluster analysis. As will be evident, this simple method can be included within ensemble procedures. It is also an *a posteriori* criterion for optimization of the obtained groupings. This procedure takes in input any partitioning clustering algorithm for which it is possible to initially choose the *k* number of clusters to be determined and a seed for the random choice of the initial *k* centroids.

The *k*-means algorithm is one of the clustering algorithms that checks all the conditions listed. So, it is considered as a reference clustering algorithm. In Weka implementation of *k*-means [7] [8], the name of the algorithm is *SimpleKMeans*; in this version the seed parameter is *s*, that is the initialization value for the random number generator. Using the same seed value will always result in the same initial centroids then. Exploiting this seed parameter, many different configurations are evaluated and compared, and also used in our meta-algorithm for ensemble final configuration.

Finally, a "soft" interpretation of the clustering is presented, in order to better explore and understand the results, to find possible outliers in the dataset, and to fix the best parameters.

A. Structure of the paper

In Section II, we present some Cluster Analysis general outlines, including main definitions, its scope and its role in

 ISBN: 978-1-61208-654-5

Data Mining. Furthermore, some concepts regarding Ensemble Clustering, soft and hard clustering are mentioned.

In Section III, the original k -means algorithm is synthesized, exposing its pros and cons.

In Section IV, the Ensemble Clustering Fuzzification Means (ECF-means) is presented, including some main definitions, validation measures, and clustering validity indexes.

In Section V, the ECF-means SW application is explained.

In Sections VI and VII, we show how the implemented tool has been used in two different applications, underlining how it helped us to explore datasets, discover new knowledge and to group objects in order to train custom models.

Finally, in Section VIII, we show our general considerations and future works.

II. CLUSTER ANALYSIS

Clustering, or Cluster analysis, methods belong to intersection of Statistics, Machine Learning, and Pattern Recognition. It is a very useful method for discovery pattern in large amount of data. It is a technique to group a set of objects into subsets or clusters. Usually, objects are described by attributes, also called features. It has become one of the most widespread unsupervised techniques of Data Mining. It has multiple real applications, above all for the simplicity of the algorithms and their readings.

A. Introduction, Definitions and Scope

A Clustering algorithm produces a partition on an unlabeled data set, such that no cluster is empty, no two clusters intersect, and the union of all clusters is the data set itself.

The goal is to create clusters that are coherent internally, but substantially different from each other. In a nutshell, objects in the same cluster should be as similar as possible, whereas objects in one cluster should be as dissimilar as possible from objects in the other clusters.

Similarity between objects that belong to a cluster is usually measured by a metrics d. Two objects x and y are similar if the value of $d(x, y)$ is small; what "small" means depends on the context of the problem. d is defined by some distance measure. Typically, the Euclidean Distance (or simply the squared Euclidean Distance) is widely used in many applications (it is also used in the ECF-means) for the computation of similarities: $ED^2(x, y) = \sum_{i=1}^{n}(x_i - y_i)^2$.

It is important to underline that, also depending on the type of data, other many metrics are possible.

Numerous clustering algorithms are available in the literature and there are several points of view for examining clustering techniques; a very good landscape of Clustering algorithms can be retrieved in [2], and an in-depth and complete study of clustering techniques, algorithms and applications can be retrieved in [9].

B. Ensemble Clustering

Different clustering approaches or different views of the data can lead to different solutions to the clustering problem. Indeed, also initial settings of a fixed algorithm may produce clusters that are very different from one another. This evidence is closely related to the theory of Ensemble Clustering (or Multiview Clustering), that studies this issue from a broader perspective [3] [10]. Therefore, instead of running the risk of picking an unsuitable clustering algorithm, a cluster ensemble can be used in order to get a "better" clustering configuration. The idea is that no single model or criterion truly captures the optimal clustering, but a collective of models will provide a more robust final solution.

Most ensemble models use the following three steps to discovery the final clusters configuration:

1. Generate N different clusterings, by using different approaches, or different data selection, different settings of the same algorithm, or different clusterings provided by different runs of the same algorithm. These represent the ensemble components.
2. Combine the results into a single and more robust clustering, by using a rule or a set of rules (called meta-rule).
3. Evaluate the ensemble clustering result and compare it with the results of the N components.

As already mentioned, the ensemble components can be selected in a wide variety of ways. Some strategies for building clustering ensemble components follow:

1. By using different subsets of features. Each clustering configuration is found by means of overlapping or disjoint subsets of the original features set.
2. By selecting different subsets of the data, via random sampling.
3. The different components can be selected combining a variety of models and algorithms such as partitioning, hierarchical or density-based methods, random or deterministic algorithms, and so on.
4. The different components can correspond to different settings of the same algorithm.
5. The different components could be obtained from a single algorithm, randomizing the initial choice of the clusters centroids. Of course, an example is k-means; thus, the ensemble can be formed as the result of N different runs of the algorithm.

After the individual components have been obtained, it is often a challenge to find a meta-rule able to combine the results from these different solutions in order to create a unified ensemble clustering.

C. Hard and Soft Clustering

Clustering algorithms can also be classified into hard and soft algorithms. A hard clustering algorithm leads to a partition of crisp sets. In a crisp set, an element is either a member of the set or not. On the other hand, a soft clustering algorithm leads to fuzzy clusters. Fuzzy sets allow elements to be partially in a set. Each element is given a degree of membership in a set.

One of the most famous fuzzy clustering algorithms is Fuzzy C-means [11], which allows an object to belong to two or more clusters with a membership degree between zero (not an element of the set) and one (a member of the set). It has been widely used in many real-world application domains where well-separated clusters are typically not available.

 ISBN: 978-1-61208-654-5

The method presented in this article leads to a fuzzy partitioning of the starting dataset, by repeatedly applying the results of the k-means algorithm.

III. THE k-MEANS ALGORITHM

k-means is a simple clustering algorithm whose main goal is to find k non-overlapping clusters. Each final cluster is represented by its centroid that is typically the mean of the points in that cluster.

A. Introduction, scope and procedure

k-means is one of the oldest and still widely used algorithms for cluster analysis. Without any doubt, it represents the archetype of the clustering partitioning algorithms. Because of its mathematical simplicity, it is also the most studied unsupervised learning technique [12], and over the years, many of its variations and extensions have been implemented (for High-Dimensional Data, for Data Streams, Time Series, for Data with noise, and so on).

Its basic algorithmic structure is shown in the Figure 1.

k-means Clustering Algorithm

Input: S set of instances; k number of clusters
Output: set of k clusters with k centroids

1. Randomly initialize k cluster centers (centroids)
2. **While** termination condition is not satisfied {
3. Assign instances to the closest cluster center
4. Update cluster centers using the instances assignment
5. }

Figure 1 – k-means Algorithm.

The condition of termination of the process is satisfied when no point changes clusters.

B. Pros and Cons

The algorithm has been very successful thanks to its simplicity and also for its linear time complexity $O(knl)$, where n is the number of objects to be clustered and l is the number of iterations that the algorithm is performing.

Like most partitioning clustering algorithms, k-means has some disadvantages:

1. It is very sensitive to the presence of outliers and noise.
2. The number of clusters need to be specified by the user and often it's not simple to choose it.
3. It is not able to discover concave-shaped clusters.
4. Since the initial choice of k centroids is random, different selections can also lead to very different final partitions, especially for large datasets with many features.

The k-means algorithm always terminates, but it does not necessarily find the "best" set of clusters.

IV. ENSEMBLE CLUSTERING FUZZIFICATION MEANS

The initial selection of centroids can significantly affect the result of the k-means algorithm. To overcome this, the algorithm can be run several times for a fixed value of k, each time with a different choice of the initial k centroids.

In many software implementations of k-means, for example in its Weka version, it is possible to choose a seed parameter (s), useful for the random selection of the first initial centroids (s is the random number seed to be used). Using this parameter, it is possible to realize, as will be described in the following sections, a procedure able to optimize and reinforce the obtained partition.

A. Introduction and Definitions

Let $S \subseteq \mathbb{R}^m$ be a set of points. Let k be the desired number of clusters to be determined. Changing the seed (s) from 0 to $N-1$, N partitions of S can be generate by applying the k-means algorithm. Some of these partitions are exactly the same, considering or not the order of groupings. Others, however, differ for very few records, and others for many.

In the following $N \times k$ matrix, called Clustering Matrix C of S, each row is a partition of k clusters of S.

$$C = \begin{pmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,k} \\ C_{2,1} & C_{2,2} & \dots & C_{2,k} \\ \dots & \dots & C_{i,j} & \dots \\ C_{N,1} & C_{N,2} & \dots & C_{N,k} \end{pmatrix}$$

$C_{i,j}$ is the j-th cluster obtained at the i-th iteration of the clustering algorithm, with $i = 1, \dots, N$ and $j = 1, \dots, k$.

It is possible associate a new $N \times k$ matrix to C, called MU matrix, that is the matrix of the centroids of the clusters:

$$C \to \begin{pmatrix} \mu(C_{1,1}) & \mu(C_{1,2}) & \dots & \mu(C_{1,k}) \\ \mu(C_{2,1}) & \mu(C_{2,2}) & \dots & \mu(C_{2,k}) \\ \dots & \dots & \mu(C_{i,j}) & \dots \\ \mu(C_{N,1}) & \mu(C_{N,2}) & \dots & \mu(C_{N,k}) \end{pmatrix} = MU$$

$\mu(C_{i,j})$ is the arithmetic mean of the j-th cluster of the i-th iteration of the algorithm, with $i = 1, \dots, N$ and $j = 1, \dots, k$.

B. Clusters Sort Algorithm

The algorithm in Figure 2 is useful for sorting the clusters partitions of C matrix. This step is essential because k-means can produce different orders of clusters in different runs, even if the partitioning results can be the same.

Please note it is possible that the average of some elements of the second row C_2 in Algorithm 1 of Figure 2 has a minimum distance from two or more averages of elements of the first row C_1. In this case, the minimum value of the minimum values is chosen.

C. The ECF-means Algorithm

Let C be a Cluster Matrix of S, sorted by using the Algorithm 1. We define $\underline{C_j}$ as ***floor of*** C_j: $\underline{C_j} = \bigcap_{i=1}^{N} C_{i,j}$, with $j = 1, \dots, k$. It is possible that $\underline{C_j} = \emptyset$ ($j = 1, \dots, k$). Moreover, $\underline{S} = \bigcup_{j=1}^{k} \underline{C_j}$ is defined as the ***floor of*** S.

Let x be an element of S; we can count the number of clusters of the first column of C where x is, the number of clusters of the second column of C where x is, and so on. In this way, we can associate a new numerical vector to x, called ***attitude of*** x ($att(x)$):

$$att(x) = \left(att_1(x), att_2(x), \dots, att_k(x)\right),$$

where $att_j(x)$ is the number of clusters in the j-th column of C where x is located. $att_j(x) = N \Leftrightarrow x \in \underline{C_j}$ and $\sum_{j=1}^{k} att_j(x) = N$. In this manner, we are defining a function att_j ($j = 1, \dots, k$ and $I = \{1,2, \dots, N\}$):

$$att_j : x \in \bigcup_I C_{i,j} \rightarrow att_j(x) = \left|\left\{i \in I : x \in C_{i,j}\right\}\right|$$

where, as usual, $|A|$ is the number of the elements of the set A.

Algorithm 1: Clusters Sort Algorithm

Input: two different rows of C:
$C_1 = \left(C_{1,1}, C_{1,2}, \dots, C_{1,k}\right)$ and $C_2 = \left(C_{2,1}, C_{2,2}, \dots, C_{2,k}\right)$
Output: a new order of the second row:
$\left(C'_{2,1}, C'_{2,2}, \dots, C'_{2,k}\right) = C'_2$

C_1 represents the reference row of the current sorting procedure (e.g., obtained by fixing $s = 0$ in the Weka k-means algorithm).

1. Calculate the $2 \times k$ matrix of clusters centroids:
   $MU = \begin{pmatrix} \mu(C_{1,1}), \mu(C_{1,2}), \dots, \mu(C_{1,k}) \\ \mu(C_{2,1}), \mu(C_{2,2}), \dots, \mu(C_{2,k}) \end{pmatrix}$
2. Compute the Euclidean Distances (ED) in MU. The following $k \times k$ matrix is the Δ matrix of the EDs:
   $\Delta = \begin{pmatrix} d_{1,1} & d_{1,2} & \dots & d_{1,k} \\ d_{2,1} & d_{2,2} & \dots & d_{2,k} \\ \dots & \dots & \dots & \dots \\ d_{k,1} & d_{k,2} & \dots & d_{k,k} \end{pmatrix}$
   Where:
   $d_{i,j} = ED\left(\mu(C_{1,i}), \mu(C_{2,j})\right)$, with $i, j = 1, \dots, k$.
3. Calculate the minimum value of each row of Δ.
   $\min\{d_{1,1}, d_{1,2}, \dots, d_{1,k}\} = d_{1,\overline{j_1}} = min_1$
   $\min\{d_{2,1}, d_{2,2}, \dots, d_{2,k}\} = d_{2,\overline{j_2}} = min_2$
   …… …… ……
   $\min\{d_{k,1}, d_{k,2}, \dots, d_{k,k}\} = d_{k,\overline{j_k}} = min_k$
4. The second row C'_2 is:
   $\left(C'_{2,1}, C'_{2,2}, \dots, C'_{2,k}\right) = \left(C_{2,\overline{j_1}}, C_{2,\overline{j_2}}, \dots, C_{2,\overline{j_k}}\right)$
   Where:
   $C'_{2,1} = C_{2,\overline{j_1}}$ is the cluster (in C_2) that has the centroid with the minimum distance from the centroid of the first element of C_1.
   …… …… ……
   $C'_{2,k} = C_{2,\overline{j_k}}$ is the cluster (in C_2) that has the centroid with the minimum distance from the centroid of the k-th element of C_1.

Figure 2 – Clusters Sort Algorithm.

Finally, we can define the ***probability vector of x***, as:

$$p(x) = \left(\frac{att_1(x)}{N}, \frac{att_2(x)}{N}, \dots, \frac{att_k(x)}{N}\right)$$

Thanks to the simple mathematical notions of the current section, we are able to "soften" the "hard" k-means algorithm and we can have a new Fuzzy Clustering Algorithm. According to this approach, each element of the dataset belongs to each cluster with a different degree of membership, and the sum of these probabilities is equal to one.

Furthermore, the method can also be interpreted in a different way. Indeed, this "fuzzification" procedure can be used not only with k-means algorithm, but also for others partitional clustering algorithms for which it is possible to choose the number of clusters to be determined. In this way, the algorithm is part of the Ensemble algorithms. For these reasons, ECF-means is also a meta-algorithm because we reach a fuzzy partition of the dataset by using a multiple clustering algorithm schema.

Algorithm 2: ECF-means (Fuzzification of k-means)

Input: $S \subseteq \mathbb{R}^m$; number k of clusters to be determined; membership threshold t ($0 \leq t \leq 1$); number N of k-means iterations
Output: set of k clusters of level t; probability vector of each element x of S

1. **Apply** the k-means algorithm to S, fixing the random seed $s = 0$, obtaining the clusters $C_{0,1}, \dots, C_{0,k}$ ($C(0)$-configuration)
2. **foreach** $s = 1, \dots, N-1$
3. **Apply** the k-means algorithm to S, obtaining the clusters $C'_{s,1}, \dots, C'_{s,k}$ ($C'(s)$-configuration)
4. **Apply** the Clusters Sort Algorithm to $C'(s)$, considering $C(0)$ as reference, obtaining the clusters $C_{s,1}, \dots, C_{s,k}$ ($C(s)$-configuration)
5. **end**
6. **foreach** $j = 1, \dots, k$
7. **foreach** $x \in S$
8. **Calculate** $p_j(x) = att_j(x)/N$
9. **Fix** the cluster $C_j^t = \{x \in S |\, p_j(x) \geq t\}$
10. **end**
11. **end**

Figure 3 – ECF-means Algorithm.

The membership threshold t in the Algorithm 2 (Figure 3) is fixed by the user and it is very useful to change the "level" to clusters final configuration. If $t = 1$, then $C_j^1 = \{x \in S | p_j(x) = 1\} = \underline{C_j}$. Additionally, $\underline{S} = \bigcup_{j=1}^{k} C_j^1$. If $t = 0$, then $C_j^0 = \{x \in S | p_j(x) \geq 0\}$ and $\bigcup_{j=1}^{k} C_j^0 = C_j^0 = S$.

Let $p(x)$ be the probability vector of x and let $M = \max att(x) = \max\{att_1(x), att_2(x), \dots, att_k(x)\}$ be the maximum of $att(x)$, if this exists. We can define the position of M in $att(x)$ as $PMA(x)$, if this exists.

An element $x \in S$ is an ***o-rank fuzzy outlier of S*** if $p_j(x) - p_l(x) \leq o$, where $p_j(x)$ and $p_l(x)$ are the first two highest value components of $p(x)$. Note that, in this contest, the word "outlier" takes on a different meaning than its scientific usual use; but the definition of *o-rank fuzzy outlier* helps us to treat these points as special points, that need to be observed more closely, because they belong at least to two different clusters.

D. Validation Measures for Fuzzy Clustering

Clustering validation has long been recognized as one of the critical issues essential to success of clustering applications [9].

Let $U = (u_{li})$ $(1 \leq l \leq k, 1 \leq i \leq n)$ be the membership's matrix of a fuzzy partition of a dataset S with n records, and k is the number of clusters.

The first validity index for fuzzy clustering is the Partition Coefficient Index (PC) [13]. PC is based on U and it is defined as: $PC = \frac{1}{n}\sum_{l=1}^{k}\sum_{i=1}^{n} u_{li}^2$.

$PC \in [1/k, 1]$. Furthermore, a PC value close to $1/k$ indicates that clustering is "very fuzzy"; the value $1/k$ is obtained when $u_{li} = 1/k$, for each l, i.

Another index is the Partition Entropy Coefficient (PE): $PE = -\frac{1}{n}\sum_{l=1}^{k}\sum_{i=1}^{n} u_{li} log_a(u_{li})$.

$PE \in [0, log_a k]$. Furthermore, a low PE value indicates that clustering is "not very fuzzy". PE values close to the upper limit indicate an absence of any clustering structure within the dataset or the inability of the algorithm to extract it.

The main disadvantage of PC and PE is their monotonic evolution tendency with respect to k. To avoid this, a modification of the PC index can reduce the monotonic tendency and was defined by: $MPC = 1 - \frac{k}{k-1}(1 - PC)$, where $0 \leq MPC \leq 1$.

Finally, let we define a novel validity index, that we call the Threshold Index TI, by the following formula:

$$TI = \frac{|\underline{S}|}{|S|}$$

V. ECF-MEANS TOOL

With the purpose of testing the ECF-means algorithm, a software application has been designed and developed. It has been carried on using a Client/Server architectural pattern, where the Server part consists of the algorithm and other support utilities, while the Client part is made by a browser-based application, responsible of the ECF-means result visualization.

A. Software Implementation

The ECF-means web application is built up of two main modules: the first one wraps the ECF-means Algorithm, that has been implemented in Java programming language, and it makes use of the Weka k-means algorithm (*SimpleKMeans*) [7] [14] as clustering algorithm implementation.

The second module consists of the web application client part, that has been implemented by using JavaScript libraries, such as D3.js, as visualization library, and jQuery for Ajax asynchronous data communication and Document Object Model (DOM) manipulation tasks.

B. GUI & Data Visualization

The implemented tool provides a user-friendly GUI, by which it is very easy to load datasets, fix the ECF-means parameters, and understand the algorithm results visually.

The GUI can be divided into three functional blocks, as highlighted by red numbered circle in Figure 4.

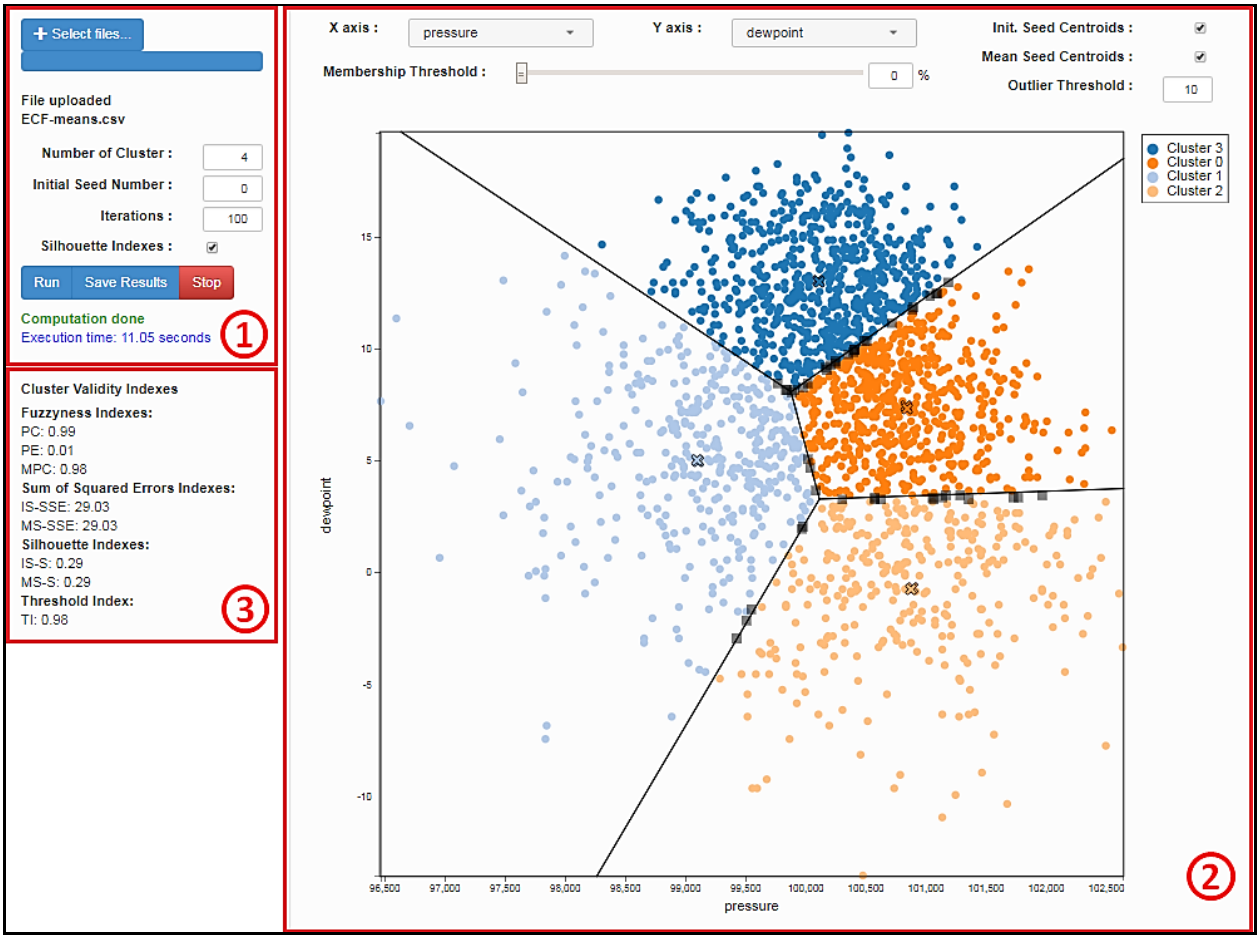

Figure 4 – Software GUI and Clustering Visualization.

Through the first functional block, user can upload a dataset from a local file system, in csv or arff formats; after that he can specify the number of clusters k (default is set to two), the initial seed number (default 0), and the number of iterations N to perform (default 100). Lastly, a set of buttons allow the following operations:

1. *Run*: runs the ECF-means algorithm and displays the results (clustering graphical visualization and validation measures output).
2. *Save Results*: saves results to an output csv file.

The second block is where clustering visualization takes shape: dataset points are displayed as circle with the color of the belonging cluster (resulting from the highest value of the probability membership vector) and with an opacity due to the degree of membership to the same cluster (stronger opacity means higher membership). If the attributes of the dataset are two, Voronoi lines (computed considering initial seed) are also displayed. In the top of the block, some input controls are used to affect data visualization. In particular, two combo boxes are used to allow the choosing of dataset's attributes that has to be displayed. Below this, a slider allows to set the degree of membership above which a point is displayed (Membership Threshold t).

Instead, rightmost input fields, in order from top to bottom, control, respectively:

1. The displayed of the Initial Seed Centroid and Mean Seed Centroid points (as a bold colored X).
2. The Outlier Threshold, that controls the o-rank fuzzy outliers, with the meaning that, if a data point has a difference between the two highest values of its probability membership vector less than this value, the point is displayed as a grey squared.

Lastly, in the third box the validation measures are displayed, as described in IV.D, such as PC, PE, MPC, and TI. In addition, Sum of Squared Errors (SSE) and Silhouette (S) measures have also been included; they are calculated considering Initial Seed (IS) and Mean Seed (MS), where MS is the mean value of the measure over all the N iterations, which lead to the definition of IS-SSE, MS-SSE, IS-S, and MS-S.

C. Output Results

The ECF application exports results in csv format, where each row of the output file represents a point x of the dataset. The application appends ECF-means algorithm results as additional columns to the attributes columns of the point x.

TABLE I. COLUMN NAMES MEANING

Column Names	Description
ISCDistance$_i$, with $i = 1, \dots, k$	Vector of Euclidean Distances between point x and Initial Seed Centroids
ISCMembership	Cluster membership derived from the position of the smallest value in ISCDistance vector
MSCDistance$_i$, with $i = 1, \dots, k$	Vector of Euclidean Distances between point x and Mean Seed Centroids
MSCMembership	Cluster membership derived from the position of the smallest value in MSCDistance vector
Membership$_i$, with $i = 1, \dots, k$	Probability vector of point x, $p(x)$
ECFMembership	Cluster membership derived from the $PMA(x)$

Table I shows these additional column names meaning, where Mean Seed Centroid (MSC) is the arithmetic mean value of all computed centroids in N iterations.

VI. CASE STUDY IN METEOROLOGICAL DOMAIN

In order to test the ECF-means algorithm and the validation measures, an historical dataset made up of 9200 meteorological observations has been collected. Data have been retrieved from ECMWF MARS Archive [15] containing the surface Synoptic observations (SYNOP) provided by 4 geographical sites: Charles De Gaulle (CDG) airport in Paris and Grazzanise, Milan, and Pantelleria airports in Italy.

TABLE II. LIST OF METEOROLOGICAL VARIABLES (FEATURES)

#	Name	#	Name
1	Pressure	6	cloud cover
2	three-hour pressure change	7	height of base of cloud
3	wind direction	8	Dewpoint
4	wind speed	9	Drybulb
5	Visibility	10	SITE

SYNOP observations are recorded every hour and the list of the meteorological variables [16] used for applying the ECF-means algorithm is reported in Table II. Each airport site has got 2300 records and the SITE attribute has 4 values.

A. k-means Application

By changing k value (number of clusters) from 2 to 7 and fixing the seed $s = 0$, the k-means algorithm has the silhouette measures of the Table III. These outcomes have been calculated by using the ECF-means application fixing *Initial Seed Number* = 0 and *Iterations* = 1. Considering this measure, the best clustering partition is obtained by selecting $k = 3$.

Fixing $k = 3$ and considering SITE attribute as Class attribute, Classes to Clusters (contingency table) is showed in Table IV. CDG and Milan have been inserted into the same cluster (Cluster 0) by the algorithm (4 sites in 3 clusters): it seems that the two sites have a lot in common! Thus, we try to merge these two sets, obtaining a new set called CDG+MIL.

TABLE III. k-MEANS RESULTS

k	Silhouette	k	Silhouette
2	0.34	5	0.38
3	0.49	6	0.36
4	0.37	7	0.31

TABLE IV. CLASSES TO CLUSTERS

0	1	2	Assigned to cluster
1593	410	297	CDG → Cluster 0
499	1260	541	Grazzanise → Cluster 1
1313	692	295	Milan → Cluster 0
387	589	1324	Pantelleria → Cluster 2
41%	32%	27%	

The incorrectly clustered instances are 3710 and represent 40.32% of the original dataset. k-means does not provide homogeneous clusters with respect to SITE attribute. From an intuitive point of view, the 3 sites (Grazzanise, Pantelleria and CDG+MIL) have an ambiguous meteorological nature and the

 ISBN: 978-1-61208-654-5

3710 unclustered instances are on the border between two or more sites. In other words, the datasets have overlapping areas, with "similar" meteorological conditions, and perhaps the sites are not so different, and they are not well-separated from each other.

If we expected the 3 sites to be able to determine (or clearly separate) even the 3 clusters, we are now disappointed. And nobody knows if this disappointment is due to k-means algorithm or to the fact that the sites are not completely different from each other from a weather (and consequently statistical) point of view.

B. ECF-means Application

The ECF-means algorithm tries to overcome the problem in which the k-means algorithm falls in this meteorological case study, and as we will see in next section, in part it succeeds, if only because it provides much more information on datasets, clusters, and on clustering results, thanks to which the data analyst can make more informed and useful choices.

An Ensemble combination of many runs of k-means by using ECF-means application showed other results and better performances than the single run of the previous section.

By fixing $k = 3$, the ECF-means algorithm provides the results of the Table V that shows how the metrics stabilize as the number I of iterations increases.

TABLE V. ECF-MEANS METRICS

I	MPC	TI	I	MPC	TI
2	0.98703	0.98054	150	0.97716	0.9762
5	0.9917	0.98054	200	0.97975	0.9762
10	0.99066	0.97828	250	0.98138	0.58293
25	0.98657	0.97828	300	0.98269	0.58293
50	0.9877	0.97828	**350**	**0.98339**	**0.58293**
75	0.97428	0.9762	400	0.98083	0.58293
100	0.97612	0.9762	500	0.98131	0.58293

TABLE VI. CLASSES TO CLUSTERS

0	1	2	Assigned to cluster	# of Records
2494	232	69	CDG+Milan → Cluster0	2795
67	1183	141	Grazzanise → Cluster1	1391
53	142	982	Pantelleria → Cluster2	1177
49%	29%	22%		tot. 5363

By selecting $I = 350$, ECF-means has the highest value of MPC $(= 0.98339)$ and the lowest value of TI $(= 0.58293)$. Thanks to the ECF-means application, we are able to select the floor $\underline{S}$ of whole dataset. It has got 5363 records that have the distributions in Table VI. The incorrectly clustered instances are 704 and represent 13.12% of $\underline{S}$.

C. Experimental Results

The results obtained lead to a clear improvement of the clustering: the clusters seem much more separate, if the contingency matrices are calculated starting from the floor set. ECF-means manages to break down the percentage of instances that are incorrectly clustered from 40.32% to 13.12%.

The elements belonging to $\underline{S}$ never fluctuate from one cluster to another (considering the 350 iterations) and constitute approximately 58.3% of the initial dataset. The elements of $S - \underline{S}$, on the other hand, have a more fuzzy nature and we found that 1369 points (about 15% of the initial dataset) have an *Outlier Threshold* (difference between the two highest values of his probability membership vector), less than 0.2.

These very fuzzy points can belong to more than one cluster and probably to more than one airport site (to overlapping areas).

VII. CASE STUDY 2: THE IRIS DATASET

The famous Iris dataset is a multivariate dataset that contains 3 classes of 50 instances each, where each class refers to a species of iris plant (Iris-setosa, Iris-virginica, and Iris-versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters. The use of this dataset is very common in classification and clustering tasks, where numerous results have been obtained.

The data set only contains two clusters with rather obvious separation: one of the clusters contains Iris setosa, while the other cluster contains both Iris virginica and Iris versicolor. This makes the dataset a good example for the ECF-means algorithm.

TABLE VII. 2-MEANS RESULTS

Index	Value	Index	Value
MPC	1.00	SSE	12.14
PE	0.00	TI	1.00

Moreover, fixing $k = 2$, the ensemble effect due to the random choice of the two initial centroids via s parameter seems to vanish, because all the iterations always lead to the same result; therefore the clustering validity indexes are (for each I) showed in Table VII.

A. ECF-means Application

Fixing $k = 3$, more interesting results are obtained by applying the ECF-means algorithm. The initial configuration ($s = 0$ and $I = 1$) has the contingency matrix of the Table VIII. The incorrectly clustered instances are 18 and represent the 12% of the original Iris dataset.

TABLE VIII. CLASSES TO CLUSTERS ($s = 0$ AND $I = 1$)

0	1	2	Assigned to cluster
0	0	50	Iris-setosa→ Cluster 2
40	10	0	Iris-versicolor → Cluster 0
8	42	0	Iris-virginica → Cluster 1
32%	35%	33%	

TABLE IX. ECF-MEANS VALIDITY INDEXES

Case	I	TI
1	2-31	0.91333
2	32-1500	0.50000

TABLE X. CLASSES TO CLUSTERS ($I = 31$)

0	1	2	Assigned to cluster
0	0	50	Iris-setosa→ Cluster 2
47	3	0	Iris-versicolor → Cluster 0
14	36	0	Iris-virginica → Cluster 1
41%	26%	33%	

 ISBN: 978-1-61208-654-5

TABLE XI. CLASSES TO CLUSTERS ($I = 31$ AND $\underline{S}$)

0	1	2	Assigned to cluster
0	0	50	Iris-setosa→ Cluster 2
40	3	0	Iris-versicolor → Cluster 0
8	36	0	Iris-virginica → Cluster 1
35%	28%	37%	

By changing the I parameter, mainly the TI index takes two values, as reported in Table IX. Considering case number 1, EFC-means provides the results of the Table X. The incorrectly clustered instances are 17 and represent the 11.33% of the original Iris dataset. Thanks to the ECF-means application, we are able to select the floor $\underline{S}$ of whole Iris dataset S. $\underline{S}$ has got 137 elements that have the distributions in Table XI. $\underline{S}$ has got 11 incorrected clustered instances that represent the 8% of $\underline{S}$. In conclusion, if $t = 1$, then $|C_0^1| = 48$, $|C_1^1| = 39$, and $|C_2^1| = 50$.

B. Experimental Results

Thanks to the obtained results, we can easily understand how the algorithm is able to optimize the partitioning of the data space with respect to the class that expresses the floral typology. The algorithm is able to find this partitioning in one fell swoop. By applying the simple k-means we may not be able to get the same partition. However, the most interesting result is that the algorithm is able to preserve the cluster with Iris-setosa label and to find the floating elements that are at the limits of the floral types. These "disturbing" elements can be analyzed separately in order to understand if they are, from some point of view, outliers or records that have undergone measurement errors.

Moreover, analyzing the floor of S, only a small fraction of Iris-virginica is mixed with Iris-versicolor and only the cluster 0 is modified by the procedure. Also in this case, the 13 elements of $S - \underline{S}$ can be analyzed separately in order to understand their fuzzy nature. These 13 elements have $att(x) = (21, 10, 0)$ and $p(x) = (0.68, 0.32, 0)$. Then they are 0.36-rank fuzzy outliers of S $(p_0(x) - p_1(x) \leq 0.36)$. Furthermore, 7 elements of $S - \underline{S}$ have Iris-versicolor label whilst 6 elements have Iris-virginica label, and all of them have the same maximum degree of membership that is equal to 0.68.

VIII. CONCLUSION AND FUTURE WORKS

In this work, we have presented an algorithm for Ensemble or Aggregation clustering that has, as a simple consequence, the fuzzy reinterpretation of the obtained groupings. We have applied the developed procedure to two different case studies by using a simple software application, which made us understand how this approach helps to explore the dataset, to optimize the results and to assign a degree of membership to each element of the original dataset.

We have had that all the most exciting results can be obtained by the active interaction with the software tool interface, thanks to which, by scrolling the sliders, changing parameters, and visualizing groupings, numerous properties of the dataset can be discovered. In future works, we are going to evaluate our method on other several data sets, for example Datasets from UCI ML Repository.

For the current application, we have chosen the simple k-means as the reference clustering algorithm. Furthermore, we can consider other algorithms in substitution or in addition to it, and this will surely be the next improvement of the tool.

ACKNOWLEDGMENT

The authors would mention the TECVOL II and Big Data Facility projects, both funded by the Italian PRORA, in which the meteorological database has been realized and the tool has been designed and developed.

REFERENCES

[1] P. Berkhin, "Survey of clustering data mining techniques," Technical report, Accrue Software, San Jose, CA, 2002.

[2] A. K. Jain, A. Topchy, M. H. C. Law, and J. M. Buhmann, "Landscape of Clustering Algorithms," IAPR International Conference on Pattern Recognition, vol. 1, pp. 260-263, Cambridge, UK, 2004.

[3] L. I. Kuncheva, "Combining Pattern Classifiers. Methods and Algorithms," Wiley-Interscience, A John Wiley & Sons, Inc., Publication, 2004.

[4] A. Strehl and J. Ghosh, "Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions," Journal of Machine Learning Research 3 (2002), pp. 583-617.

[5] S. Monti, P. Tamayo, J. Meisirov, and T. Golub, "Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data," Machine Learning, 52, pp. 91–118, 2003, Kluwer Academic Publishers.

[6] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering Aggregation," on 21st International Conference on Data Engineering (ICDE), pp. 341-352, 2005.

[7] R. R. Bouckaert et al., "WEKA Manual for Version 3-9-2," the University of Waikato, Hamilton, New Zealand, December 22, 2017.

[8] E. Frank, M. A. Hall, and I. H. Witten, "Data Mining: Practical Machine Learning Tools and Techniques," Morgan Kaufmann, 2016.

[9] G. Gan, C. Ma, and J. Wu, "Data Clustering. Theory, Algorithms, and Applications," ASA-SIAM Series on Statistics and Applied Probability, SIAM, Philadelphia, ASA, Alexandria, VA, 2007.

[10] C. C. Aggarwal, "Data Mining. The Textbook," Springer International Publishing, 2015.

[11] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The Fuzzy c-Means Clustering Algorithm," Computers & Geosciences Vol. 10, No. 2-3, pp. 191-203, 1984.

[12] J. Wu, "Advances in k-means Clustering. A Data Mining Thinking," Springer, 2012.

[13] R. N. Dave, "Validating fuzzy partition obtained through c-shells clustering," Pattern Recognition Lett. 17, pp. 613– 623, 1996.

[14] M. Hall et al. (2009), "The WEKA Data Mining Software: An Update," SIGKDD Explorations, Volume 11, Issue 1.

[15] ECMWF, Mars User Guide. User Support. Operations Dep. 2013.

[16] G. Zazzaro, G. Romano, and P. Mercogliano, "Data Mining to Forecasting Fog Events and Comparing Geographical Sites. Designing a novel method for predictive models portability," Int. Journal on Advances in Nets and Services, vol. 10 no 3 & 4, pp. 160-171, 2017.

 ISBN: 978-1-61208-654-5