# K-Means over Incomplete Datasets using Mean Euclidean Distance

Loai AbdAllah[1] and Ilan Shimshoni[2]

[1] Department of Community Information Systems Zefat Academic College, Israel
Department of Mathematics and Computer Science, The College of Sakhnin for Teacher Education, Israel,
[2] Department of Information Systems, University of Haifa, Israel,

**Abstract.** Missing values in data are common in real world applications. In this research we developed a new version of the well-known k-means clustering algorithm that deals with such incomplete datasets. The k-means algorithm has two basic steps, performed at each iteration: it associates each point with its closest centroid and then it computes the new centroids. So, to run it we need a distance function and a mean computation formula. To measure the similarity between two incomplete points, we use the distribution of the incomplete attributes. We propose several directions for computing the centroids. In the first, incomplete points are dealt with as one point and the centroid is computed according to the developed formula derived in this research. In the second and the third, each incomplete point is replaced with a large number of points according to the data distribution and from these points the centroid is computed. Even so, the runtime complexity of the suggested k-means is the same as the standard k-means over complete datasets. We experimented on six standard numerical datasets from different fields and compared the performance of our proposed k-means to other basic methods. Our experiments show that our suggested k-means algorithms outperform previously published methods.

**Keywords:** Clustering, $k$-means, missing values, incomplete datasets

## 1 Introduction

K-Means is the most popular and the simplest partitional clustering algorithm. It has a rich and diverse history as it was independently discovered in different scientific fields [10, 14, 11]. Ease of implementation, simplicity, efficiency, and empirical success are the main reasons for its popularity. The k-means algorithm (which is an EM type algorithm) has two basic steps, performed at each iteration: (1) It associates each point with its closest centroid. (2) It computes the new centroids.

Developing such an algorithm for datasets with missing values is not a trivial challenge. It is important, since missing values are very common in real world datasets. They can be caused by human error, equipment failure, system generated errors, and so on. We were introduced to the problem of missing data

when we received datasets from Applied Materials (AMAT), a company which develops machines for the semiconductor industry. This data has many missing values.

In general there are two ways to run the k-means clustering algorithm over incomplete datasets: customizing the data or customizing the k-means algorithm. This means that we can preprocess the dataset so that it consists only of complete points and then run the standard k-means, or we can develop a k-means clustering algorithm that can deal with incomplete datasets. Our proposed method is of the latter type.

Based on [2, 7–9] there are three basic types of missing data:

1. Missing Completely at Random: Data are said to be MCAR if the failure to observe a value is not related to any other sample.
2. Missing at Random: Data are said to be MAR if the probability that a value is missing does not depend on the other missing values. Thus the conditional probability of missingness may depend on any known values.
3. Not Missing at Random: Data are said to be NMAR if the probability that a known value is missing depends on the value that would have been observed.

Several methods have been proposed to deal with missing data. These methods can be classified into two basic categories: (a) **Case deletion** method, this method assumes that the missing values are missing completely at random (MCAR). It therefore ignores all the instances with missing values and performs the analysis on the rest [16]. (b) Missing data imputation, which replaces each missing value with a known value according to the dataset distribution. A common method that imputes missing data is the **Most Common Attribute Value (MCA)** method. The value of the attribute that occurs most often is selected to be the value for all the unknown values of the attribute [5]. The **Mean Imputation (MI)** method replaces a data point with missing values with the mean of all the instances in the data. A variant of this method is to replace the missing data for a given attribute with the **M**ean of all known values of that **A**ttribute-**MA** (i.e., the mean of each attribute) in the coordinate where the instance with missing data belongs as described by [12]. All these methods assume MCAR since all of them based on the distribution of the whole data and do not take into account the correlations between the observed and the unobserved values. These imputation methods yield complete datasets. As a result the standard k-means clustering algorithm can be run.

There are also methods that run k-means over incomplete datasets without imputation, as described in [6, 4]. They estimate a Gaussian Mixture Model-GMM (an extended version of k-means) over datasets with missing values without imputation, but, as we show in this paper, our proposed method is different and yields better results.

AbdAllah and Shimshoni [1] developed a new method to compute the distance function between incomplete points. Their distance is not only efficient but also takes into account the distribution of each attribute. In the computation procedure they take into account all the possible values with their probabilities, which are computed according to the attribute's distribution. This is in contrast

to the MCA and the MA methods, which replace each missing value only with the mode or the mean of each attribute.

In a recent paper, Eirola et. al., [3] estimated the pairwise distance between incomplete samples using the Gaussian mixture model with the algorithm described in [6, 4]. Mixture models of Gaussians have been studied extensively to describe the distributions of data sets.

In this research we also developed a k-means algorithm that can run over incomplete datasets without a preprocessing procedure. To do so, we first need (1) a distance function to measure the similarity between incomplete points and each centroid in order to associate each point with the closest centroid; and (2) a formula for computing the new centroids of the clusters where each cluster may contain incomplete points.

As a result, in this research we decided to work with the *mean Euclidian distance* ($MD_E$) presented in [1] to measure the dissimilarity between the incomplete points. The $MD_E$ distance is not only efficient but also takes into account the distribution of each attribute. This distance assumes that the missing values are randomly distributed across all the samples. But, in real world datasets, the missing values may depend on information from the known values of the data. Thus, in this research we generalized this distance function to deal with other types of missing values.

We suggest three variants of k-means that can deal with incomplete datasets. All use the $MD_E$ distance to associate the points with the closest centers. It is important to note that by using this distance we are able to associate points with the centroids without knowing their exact geometric locations. The three directions differ in how to compute the new centroid for each cluster, and more specifically, in how to include the incomplete points within the mean computation procedure. The first direction assumes that each incomplete point represents one point and then it computes the mean according the developed formula for computing the mean. The other two directions assume that each incomplete point represents a set of complete points according to the data distribution, so they replace each incomplete point with a set of points and then compute the *mean* according to the new dataset. It is important to note that even though we replace each incomplete point with a large number of points, we use the histograms of the data distribution in order to make the suggested algorithm more efficient. As a result, the runtime complexity of the suggested k-means algorithms is the same as the standard k-means over complete datasets.

The proposed methods yield better results than previously published methods, as can be seen in the experiments. We experimented on six standard numerical datasets from different fields from the Speech and Image Processing Unit [15]. Our experiments show that the performance of the k-means algorithm using $MD_E$ distance function and the proposed *mean* and the k-means that use the histogram of the data were superior to k-means using other methods.

The paper is organized as follows. A review of the incomplete data distance function measure developed by [1] is described in Section 2. The mean computation is presented in Section 3. Section 4 describes several directions for integrat-

ing the ($MD_E$) distance and the computed *mean* within the k-means clustering algorithm. Experimental results of running several variants of the k-means clustering algorithm on the Speech and Image Processing Unit [15] datasets are presented in Section 5. Finally, our conclusions are presented in Section 6.

## 2 Incomplete Data Distance Measure

In this section we describe the method for measuring the distance between pairs of points when they may contain missing values developed by [1].

Let $A \subseteq \mathbb{R}^K$ be a set of points. For the $i$th attribute $A^i$, the conditional probability for $A_i$ will be computed according to the known values for this attribute from $A$ (i.e., $P(A^i) \sim \chi^i$), where $\chi^i$ is the distribution of the $i$th coordinate.

Given two sample points $X$ and $Y$ from $A$, the goal is to compute the distance between them. Let $x^i$ and $y^i$ be the $i$th coordinate values from points $X, Y$ respectively. There are three possible cases for the values of $x^i$ and $y^i$:

1. Two values are known: When the values of $x^i$ and $y^i$ are given, the distance between them will be defined as the Euclidian distance:

$$D_E(x^i, y^i) = (x^i - y^i)^2. \tag{1}$$

2. One value is missing: Suppose that $x^i$ is missing and the value $y^i$ is given. Since the value of $x^i$ is unknown, we cannot compute its Euclidian distance. Instead we model the distance as a random selection of a point from the distribution of its attribute $\chi^i$ and compute its distance. The expectation of this computation is our distance.
   As a result, we approximate the mean Euclidian distance ($MD_E$) between $y^i$ and the missing value $m^i$ as:

$$MD_E(m^i, y^i) = E[(x - y^i)^2] = \int p(x)(x - y^i)^2 dx = \left((y^i - \mu^i)^2 + (\sigma^i)^2\right).$$

   This metric measures the distance between $y^i$ and each suggested value of $x^i$ and takes into account the probability $p(x)$ for this value according to the evaluated probability distribution. It is important to note that in this computation the probability was computed according to the whole dataset. The authors did not take into account the possible correlations between the missing values and the other known values. It means that they assumed MCAR (missing completely at random) missing data type. The resulting mean Euclidian distance will be:

$$MD_E(m^i, y^i) = \left((y^i - \mu^i)^2 + (\sigma^i)^2\right), \tag{2}$$

   where $\mu^i$ and $(\sigma^i)^2$ are the *mean* and the *variance* for all the known values of the attribute.

3. The two values are missing: In this case, in order to estimate the mean Euclidian distance, we have to randomly select values for both $x^i$ and $y^i$. Both these values are selected from distribution $\chi^i$.
   We compute the expectation of the Euclidean distance between each selected value as we did for the one missing value. As a result the distance is:

$$MD_E(x_i, y_i) = \int\int p(x)p(y)(x-y)^2 dxdy = \Big((E[x]-E[y])^2 + \sigma_x^2 + \sigma_y^2\Big).$$

   As $x$ and $y$ belong to the same attribute, $E[x] = E[y] := \mu^i$ and $\sigma_x = \sigma_y := \sigma^i$. Thus:

$$MD_E(x^i, y^i) = 2(\sigma^i)^2. \tag{3}$$

Studying the equation described above, we conclude that the main limitation of this distance is its assumption that the missing data is MCAR. However, many real world datasets are not MCAR. So, if the missing are MAR then the probability $p(x)$ depends on the other observed values and then the distance will be computed as:

$$MD_E(m^i, y^i) = \int p(x|x_{obs})(x-y^i)^2 dx = \Big((y^i - \mu^i_{x|x_{obs}})^2 + (\sigma^i_{x|x_{obs}})^2\Big),$$

where $x_{obs}$ denotes the observed attributes of point $X$, and $\mu^i_{x|x_{obs}}$ and $(\sigma^i_{x|x_{obs}})^2$ are the conditional *mean* and *variance*, respectively.

On the other hand, if the missing values are of type NMAR, then the probability $p(x)$ that was used in Equation 2 will be computed according to this information and then the distance will be:

$$MD_E(m^i, y^i) = \int p(x|m^i)(x-y^i)^2 dx = \Big((y^i - \mu^i_{x|m^i})^2 + (\sigma^i_{x|m^i})^2\Big),$$

where $p(x|m^i)$ is the distribution of $x$ when $x$ is missing.

## 3 Mean Computation

In order to develop a k-means clustering algorithm over datasets with missing values, we still need a formula that uses the $MD_E$ distance to compute the *mean* of a given set that contains missing values.

Let $A \subseteq \mathbb{R}^K$ be a set of $n$ points that may contain points with missing values. Then the *mean* of this dataset is defined as:

$$\bar{x} = \arg\min_{x\in\mathbb{R}} \sum_{i=1}^{n} \left(distance(x, p_i)\right)^2,$$

for any $x \in \mathbb{R}^K$, where $p_i \in A$ denotes each point from the set $A$, and $distance()$ is a distance function.

Let $f(x)$ be a multidimensional function: $f : \mathbb{R}^K :\rightarrow \mathbb{R}$ which is defined as:

$$f(x) = \sum_{i=1}^{n} \left(distance(x, p_i)\right)^2,$$

In our case, the $distance() = MD_E$. Thus,

$$f(x) = \sum_{i=1}^{n} \left(distance(x, p_i)\right)^2 = \sum_{i=1}^{n} \Bigg( \underbrace{\sqrt{\sum_{j=1}^{K} MD_E(x^j, p_i^j)}}_{\text{The } MD_E() \text{ distance}} \Bigg)^2 = \sum_{i=1}^{n} \sum_{j=1}^{K} MD_E(x^j, p_i^j),$$

where $x^j$ is the coordinate $j$ and $p_i^j$ is the coordinate $j$ in point $p_i$. Since each point $p_i$ may contain missing attributes, and according to the definition of the $MD_E$ distance in the previous section, $f(x)$ will be:

$$f(x) = \sum_{j=1}^{K} \Bigg[ \underbrace{\sum_{i=1}^{n_j} (x^j - p_i^j)^2}_{\text{there are } n_j \text{ known coordinates}} + \underbrace{\sum_{i=1}^{m_j} \Big( (x^j - \mu^j)^2 + (\sigma^j)^2 \Big)}_{\text{there are } m_j \text{ missing coordinates}} \Bigg].$$

$\bar{x}$ is the solution of $f'(x) = 0$, but since $f(x)$ is a multidimensional function then $\bar{x}$ is the solution of $\nabla f = \overrightarrow{0}$, where

$$\nabla f = \Big( f'_{x^1}, f'_{x^2}, ..., f'_{x^k} \Big) = 0,$$

is the gradient of function $f$. In our derivation we will first deal with one coordinate and then we will generalize it for all the other coordinates.

$$\Rightarrow f'_{x^l} = 2 \sum_{i=1}^{n_l} (x^l - p_i^l) + 2 \sum_{i=1}^{m_l} (x^l - \mu^l) = 0$$

$$\Rightarrow n x^l = \sum_{i=1}^{n_l} p_i^l + m_l \mu^l \Rightarrow x^l = \frac{\sum_{i=1}^{n_l} p_i^l}{n} + \frac{m_l \mu^l}{n}$$

$$\Rightarrow x^l = \frac{n_l}{n} \frac{\sum_{i=1}^{n_l} p_i^l}{n_l} + \frac{n - n_l}{n} \mu^l = \mu^l.$$

Thus, we simply get:

$$x^l = \mu^l. \tag{4}$$

Repeating this for all the coordinates yields $\bar{x} = (\mu^1, \mu^2, ..., \mu^k)$. In other words, each coordinate of the mean is the mean of the known values of that coordinate.

In the same way, we derive a formula for computing the weighted mean for each coordinate $l$, yielding:

$$\overline{x}_w^l = \frac{\sum_{i=1}^{n_l} w_i x_i^l + \sum_{i=1}^{n_l} w_i \mu^l}{\sum_{i=1}^{n} w_i},$$

where $w_i$ is the weight of point $x_i$. Thus, in order to compute the *weighted mean* of an attribute $l$, where some of its values are missing, we must distinguish between two cases: known and missing values. If the value is known we multiply it with its weight. When, however, the value is missing, we replace it with the mean of the known values of the attribute $l$ and then multiply it by the matching weight. We then sum the terms and divide them by the sum of all the weights.

## 4 K-Means Clustering using the $MD_E$ Distance

The aim of this research is to develop $k$-means clustering algorithms for incomplete datasets. The $MD_E$ distance and the *mean* are general and can be used within any algorithm that computes distances and means. It is not, however, clear how it can be integrated into such an algorithm. In this section we describe our proposed method for doing so for the $k$-means clustering algorithm. For illustration purposes we assume that all the points are from $\mathbb{R}^2$. We propose three different versions for k-means. It is important to note that the first version is similar to the GMM algorithm described [6, 4], where each incomplete point is considered a single point. However, the other two versions are different and replace each incomplete point with a set of points according to the data distribution. As will be shown in our experiments, they outperform the first algorithm.

Given dataset $D$ that may contain points with missing values, the k-means algorithm has two basic steps, performed at each iteration: (1) It associates each point with its closest centroid. (2) It computes the new centroids. In the first step the $MD_E$ distance is used to compute the distances between the points and the $k$ centroids in order to associate each point with the closest centroid. There are several possible ways to then compute the new centroids of the clusters. These possibilities will be illustrated using Figure 1(a). According to this example, there are two clusters (i.e., $C1$ is assigned to the yellow cluster and $C2$ is assigned to the brown cluster). The goal is to compute the centroid of each cluster. For simplicity we will deal with $C1$. If none of the instances contain missing values, the centroid will be computed according to the Euclidian *mean* formula, resulting in the magenta star.

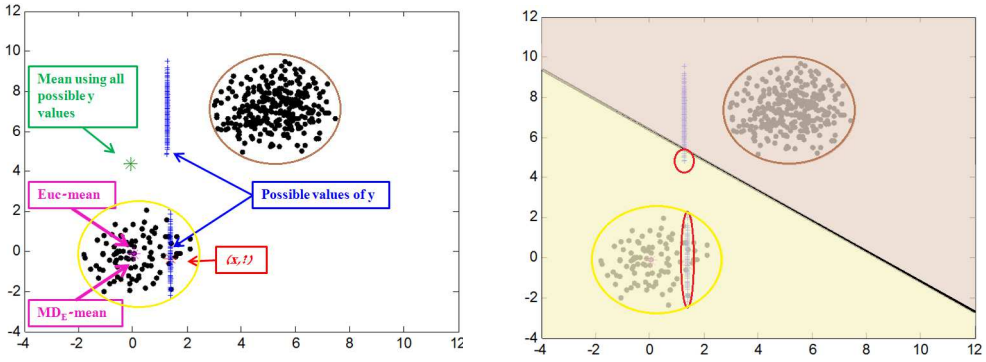
When the dataset contains points with missing values, it is not clear how to compute the mean. In the given example, let $(x_0, ?)$ (i.e., the red star) be a point with a missing $y$ value and $x = x_0$. Although the exact geometric location for this point is unknown, we still can associate it with $C1$'s cluster using the $MD_E$ distance as follows:

$$distance^2\big((x, ?), (x_c, y_c)\big) = MD_E^2\big((x, ?), (x_c, y_c)\big) = (x_c - x_0)^2 + (y_c - \mu_y)^2 + \sigma_y^2.$$

Using the MA-method, on the other hand, the point $(x_0, ?)$ will be replaced with $(x_0, \mu_y)$. It is clear that the difference between the two methods is only in $\sigma_y^2$, a fixed value that does not influence the association result.

Let the $Att_{possible}$ group denote all the possible values for each attribute. Thus, in our case:

$$Y_{possible} = \big\{y \in \mathbb{R} \big| \exists (x, y) \in D\big\}.$$




(a) (b)

**Fig. 1.** An example for computing the centroids for two clusters in a dataset with missing values. (a) Shows the results of the different methods of computing the *mean*. (b) Shows the Voronoi diagram.

Let

$$(x_0)_{possible} = \{(x_0, y_p) | y_p \in Y_{possible}\}$$

be the set of all the possible points that satisfy $x = x_0$, assume $y$ is missing (the blue "+"), and let

$$C1_{real} = \{(x, y) \in D | (x, y) \in C1\}$$

be the set of all the data points without missing values that are associated with the $C1$ cluster.

**The naïve method** to compute the new centroid is by replacing the point with the missing value with all the possible points $(x_0)_{possible}$, and then computing the *mean* according to these points ($C1_{real}$ and $(x_0)_{possible}$), where each point from $C1_{real}$ has weight one and each point from $(x_0)_{possible}$ has weight $\frac{1}{|Y_{possible}|}$. As a result, the weighted *mean* of $C1$ is:

$$mean(C1) = \frac{\sum_{(x,y) \in C1_{real}} (x, y) + (x_0, \mu_y)}{|C1_{real}| + \sum \frac{1}{|Y_{possible}|}}. \quad (5)$$

This is identical to the Euclidian *mean* when the missing point is replaced with $(x_0, \mu_y)$ and is equivalent to the MA method when $(x_0, \mu_y)$ is associated with $C1$. As a result, the real centroid of the cluster (the magenta star) moves to the green star. That is because this computation assumes that all the possible points $(x_0)_{possible}$ are represented as $(x_0, \mu_y)$, and ignores the other possible points, which in general is suboptimal, as can be seen in Figure 1 (b), where not all the blue "+" marks are associated with $C1$.

Thus, we must take into account the association of each possible point. There are two possible ways to do that. The first (which we name ***k*-mean-MD$_E$**) is

to include within the *mean* computation, in addition to the real points within the yellow circle, all the possible points $(x_0, y) \in (x_0)_{possible}$ such that their $y$ coordinates are the $y$ coordinates of the real data points from the yellow cluster (i.e., the blue "+" belonging to the yellow circle). Formally, the mean will be computed according to all the real points $C1_{real}$ and

$$C1_{(x_0)_{possible}} = \left\{(x_0, y_p) \in (x_0)_{possible} \middle| \exists (x, y) \in C1_{real} \wedge y = y_p\right\}.$$

The weight for each point from $C1_{real}$ is 1 and the weight for each point from $C1_{possible}$ is: $\frac{1}{|C1_{possible}|}$.

This means that all the weights are computed only according to the distribution of the points associated with $C1$. So, using only this set of points, the *mean* can be computed directly using (4). As a result, the centroid will be preserved, yielding the magenta star in the given example.

Computing the new centroid using (4) yields not only the same centroid as using the Euclidian distance, but also preserves the runtime of the standard k-means using the Euclidian distance.

Another method (which we name ***k-mean-HistMD***$_E$) is to use all the points from $(x_0)_{possible}$ that are associated with the $C1$ cluster and not only the points from $(x_0)_{possible}$ whose $y$ coordinates are from the real points associated with that cluster. Thus, we first associate each of the points from $(x_0)_{Ypossible}$ with its closest centroid (the blue "+" in the red circles in Figure 1 (b)), and then compute a weighted *mean*. Formally, the *mean* will be computed according to all the real points $C1_{real}$, and

$$PC1_{possible} = \left\{(x_0, y_p) \in (x_0)_{possible} \middle| (x_0, y_p) \in C1\right\}.$$

In this case we cannot use (4), because the weights are computed using the entire dataset. We therefore suggest three methods for implementing the *mean* computation:

1. Simple weighted *mean*: Simply replace each point with a missing value with the $\left|Y_{possible}\right|$ points, each with a weight $\frac{1}{\left|Y_{possible}\right|}$, and run weighted k-means on the new dataset. Thus, the weighted *mean* of $C1$ is:

$$mean_1(C1) = \frac{\sum_{(x,y)\in C1_{real}}(x, y) + \frac{1}{\left|Y_{possible}\right|} \cdot \sum_{(x_0,y_p)\in PC1_{possible}}(x_0, y_p)}{\left|C1_{real}\right| + \frac{1}{\left|Y_{possible}\right|} \cdot \left|PC1_{possible}\right|}. \tag{6}$$

   This method is simple to implement, but its runtime is high, since each point with, for example, a missing $y$ value will be replaced with $\left|Y_{possible}\right|$ points. As a result, the size of the dataset will be:

$$\left|D_{real}\right| + \left(\left|D\right| - \left|D_{real}\right|\right) \cdot \left|Att_{possible}\right|,$$

   where $D_{real}$ is the set of all the data points that do not contain missing values. We therefore chose to implement more efficient methods to compute the *mean* of the cluster.

2. Voronoi Diagram method: Using the Voronoi diagram, the data space is partitioned to $k$ subspaces (as can be seen in Figure 1 (b)). Each point is associated with the subspace of the cluster in which it lies. We can use the intersection between the Voronoi edges and the $x = x_0$ line to identify all the possible points associated with cluster $C1$, that is, the $PC1_{possible}$ points. Then all the possible points within this cluster will be represented by their Euclidian *mean* (assigned as $(x_{m_1}, y_{m_1})$). Its weight is:

$$w_{m_1} = \frac{|PC1_{possible}|}{\left|Y_{possible}\right|},$$

and the weight for each point within $C1_{real}$ is one. Thus, the *mean* formula is:

$$mean_2(C1) = \frac{\sum_{(x,y)\in C1_{real}}(x,y) + w_{m_1} \cdot (x_{m_1}, y_{m_1})}{|C1_{real}| + w_{m_1}}.$$

It is easy to see that this equation is identical to (6). However, this method is more efficient. Another possible and efficient approximation method is described below.

3. Histogram method: Instead of including each possible point in the computation procedure, we divide the $y$ value space to several disjoint intervals. Each interval will be represented by its mean, and the weight of each interval will be the ratio between the number of points in the interval to the number of all possible points. Formally, let $\Delta_i = \left\{y_p \in [y_{i-1}, y_i] \middle| y_p \in Y_{possible}\right\}$ be interval $i$. Then its representative point is: $r_i = (x_0, mean(\Delta_i))$ and its weight will be $w_i = \frac{\left|\Delta_i\right|}{\left|Y_{possible}\right|}$. The weight for each point within $C1_{real}$ is one. Thus, the formula for the *mean* is:

$$mean_3(C1) = \frac{\sum_{(x,y)\in C1_{real}}(x,y) + \sum_{r_i \in C1} w_i \cdot r_i}{\left|C1_{real}\right| + \sum_{r_i \in C1} w_i}. \tag{7}$$

This method approximates the method that computes the weighted *mean*; the only difference is for the intervals that intersect the Voronoi edges. For all the other intervals the two methods are identical. In the experiments we conclude that this method does not strongly depend on the number of intervals. This method is called ***k-mean-HistMD***$_E$. Consider the following two special cases. In the first case each interval contains only one point (i.e., $\Delta_i = \{y_p\}$, where $y_p \in Y_{possible}$). Then

$$r_i = (x_0, y_p) \in PC1_{possible} \quad , \quad w_i = \frac{1}{\left|Y_{possible}\right|},$$

and the *mean* will be:

$$mean(C1) = \frac{\sum_{(x,y)\in C1_{real}}(x,y) + \sum_{r_i \in C1} w_i \cdot r_i}{\left|C1_{real}\right| + \sum_{r_i \in C1} w_i}.$$

As a result, this $mean$ is identical to $mean_1$, described in (6). In the second case, when there is only one interval, then

$$\Delta_i = Y_{possible}, r_i = (x_0, mean(Y_{possible})) = (x_0, \mu_y), \text{and} w_i = \frac{|\Delta_i|}{|Y_{possible}|} = 1.$$

In this case the $mean$ will be:

$$mean(C1) = \frac{\sum_{(x,y)\in C1_{real}}(x,y) + (x_0, \mu_y)}{|C1_{real}| + 1}.$$

Then, this method will be equivalent to the MA imputation method described in (5).

**Discussion**

There are several differences between k-means-$MD_E$ and k-means-$HistMD_E$. They differ in their performance, efficiency and the way that they work. The k-means-$MD_E$ performs better when there are correlations between the attributes, while the k-means-$HistMD_E$ performs better when there are no strong correlations between the attributes. This is due to the different ways in which they associate points generated to represent the points with the missing values to the clusters.

Moreover, the k-means-$MD_E$ method is more efficient and its runtime is equivalent to the standard k-means as described above. In addition, the results of the two methods are different, as illustrated in Figure 1(b), where in the first method only the blue "+" signs in the yellow circle are included, while in the k-means-$HistMD_E$ method all the points within the red ellipses are included.

### 4.1 Algorithm Convergence

In the previous section we described our new suggested methods of k-means. Now we will prove that all these new methods converge to a local minimum of the cost function. This has been proven for the original k-means algorithm. In the first method (naïve k-means), as can be seen in the previous section, each missing value is replaced with the mean of all the known values of that coordinate. Then the standard k-means clustering algorithm is run on the modified data set. The algorithm therefore has the same properties as the standard k-means and it therefore also converges. A similar analysis is performed for the k-means-$HistMD_E$ algorithm. In this case each point that has a missing value is replaced with several points using the data distribution. Then again the algorithm can considered as if it were the standard weighted k-means algorithm which is run over this modified dataset. Thus, this method also has the standard k-means properties.

Since a simple reduction is not possible for the $k$-means-$MD_E$ algorithm we will now prove that this method also converges after a finite number of iterations. Let $c^{(k)}, k = 1..K$ be the centroids of each cluster $C_k$, where

$$C_k = \{x \in X : \textit{ the closest representative of } x \textit{ is } c^{(k)}\}.$$

The goal of the $k$- means algorithm is to find the $K$ centroids that minimize the cost function:

$$\{c^{(1)}, c^{(2)}, ..., c^{(K)}\} = \underset{\{c^{(1)},c^{(2)},...,c^{(K)}\}}{\arg\min} cost(C_1, C_2, ..., C_K, c^{(1)}, c^{(2)}, ..., c^{(K)})$$
$$= \underset{\{c^{(1)},c^{(2)},...,c^{(K)}\}}{\arg\min} \sum_{k=1}^{K} \sum_{x \in C_k} distance^2(x, c^{(k)}).$$

The standard $k$-means algorithm converges due to the fact that the value of the cost function monotonically decreases because in each iteration the new centroids $c'^{(i)}$ satisfy $c^{(i)} = \arg\min \sum_{x \in C_i} distance^2(x, c)$ and each point is associated with its closest centroid. We will now show that in the $k$-means-$MD_E$ algorithm the value of the cost function also decreases monotonically in the same manner.

In the previous section we showed that in the association step that using the $MD_E$ distance function each point will be associated to the closest centroid. This is true for regular points as well as for incomplete points. For each cluster, considering the points associated with it, the mean is computed using the derivation performed in Section 3, which satisfies:

$$c^{(k)} = \underset{c \in \mathbb{R}^d}{\arg\min} \sum_{x \in C_k} distance^2(x, c) = \underset{c \in \mathbb{R}^d}{\arg\min} \sum_{x \in C_i} \sum MD_E^2(x, c),$$

which is what is required.

As a result we conclude that the $k$-means-$MD_E$ clustering algorithm also converges to a local minimum like the standard $k$-means.

## 5 Experiments on Numerical Datasets

In order to measure the ability of k-means-$MD_E$ and k-means-$HistMD_E$ to cluster the incomplete datasets, we compare the performance of the k-means ($k$ is fixed for each dataset) clustering algorithm on complete data (i.e., without missing values) to its performance on data with missing values, using the $MD_E$ distance measure (k-means-$MD_E$ and k-means-$HistMD_E$) and then again using k-means-(MCA, MA, MI), where each missing value in each attribute is replaced using the MCA, MA or MI method respectively on which a standard k-means is run. In our experiments, k-means-$HistMD_E$ was run with 20 intervals. Later (in Section 5.1) we can see that the number of intervals is not critical for the algorithm's performance.

We use the Rand index [13], which is a measure of similarity between two data clusterings, to compare how similar the results of the standard k-means clustering algorithm were to the results of the other algorithms for datasets with missing values.

We ran our experiments on six standard numerical datasets from the Speech and Image Processing Unit [15] from different fields: the Flame dataset, the Jain dataset, the Pathbased dataset, the Spiral dataset, the Compound dataset, and the Aggregation dataset; dataset characteristics are shown in Table 1.

**Table 1.** Speech and Image Processing Unit Dataset Properties

| Dataset | Dataset size | Clusters |
|---|---|---|
| Flame | $240 \times 2$ | 2 |
| Jain | $373 \times 2$ | 2 |
| Pathbased | $300 \times 2$ | 3 |
| Spiral | $312 \times 2$ | 3 |
| Compound | $399 \times 2$ | 6 |
| Aggregation | $788 \times 2$ | 7 |

As can be seen in Figure 2, the two versions of k-means that use the $MD_E$ distance outperformed the other algorithm on all the datasets. Our intuitive explanation for the performance of our algorithms is as follows. In the MA MCA methods, the whole distribution of values is replaced by a single value (the mean or the mode of the distribution of known values). In our two algorithms we use the distribution of the observed values in all the computation stages. This additional information is probably the reason for the improved performance of our methods compared to the known heuristics. Moreover, we can see that, in most cases, k-means-$HistMD_E$ outperformed k-means-$MD_E$. The difference in performance is due to the data distribution as mentioned in discussion in Section 4.

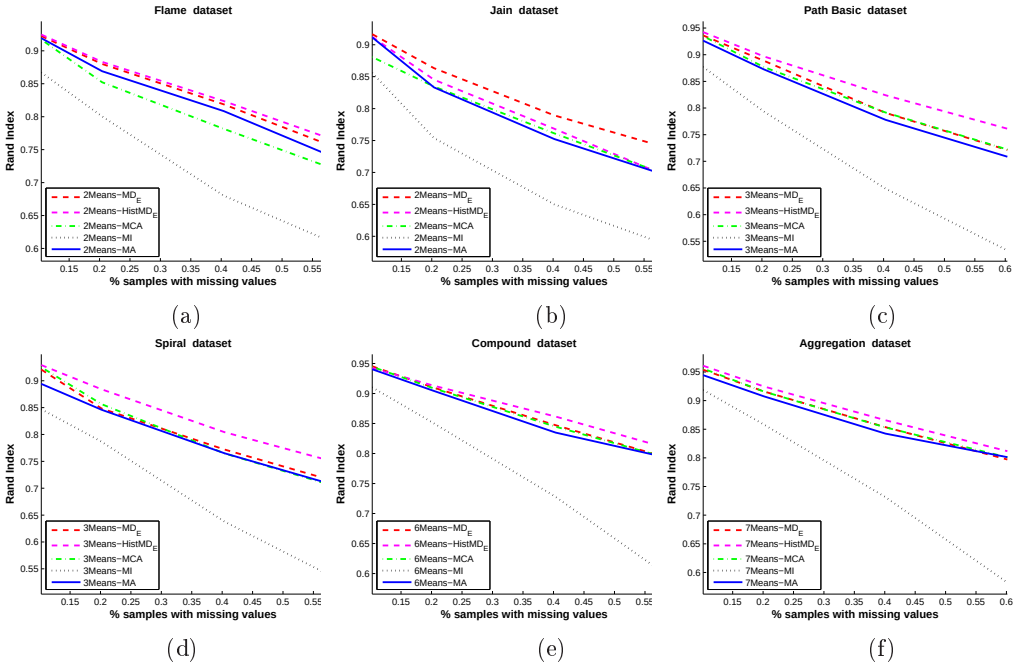



**Fig. 2.** Results of k-means clustering algorithm using the different distance functions on the six datasets from the Speech and Image Processing Unit.

### 5.1 The effect of the number of intervals on the performance of k-means-$HistMD_E$

We performed yet another experiment to determine the extent to which the algorithm depends on the number of intervals. In this experiment we evaluated the performance of the algorithm on the Spiral dataset. We ran k-means-$HistMD_E$ using six different numbers of intervals (1, 5, 10, 15, 20, 50, 100) and compared the performance to that of the standard k-means. As Figure 3 clearly shows, the performance of k-means-$HistMD_E$ does not critically depend on the number of intervals. When k-means-$HistMD_E$ was run with one interval, its performance was identical to that of k-means-MA, and when k-means-$HistMD_E$ was run with five intervals, its performance improved. At 10 intervals, its performance converged and did not change for larger numbers of intervals, as can be seen in the resulting curves. In our experiments we chose therefore to work with 20 disjoint intervals.

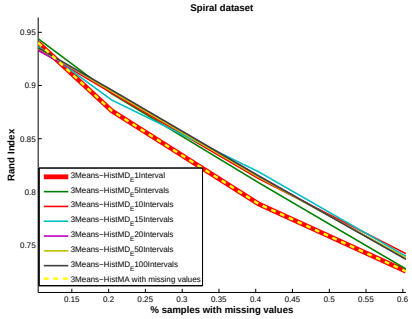



**Fig. 3.** Results of k-means-$HistMD_E$ clustering algorithm, when $k = 3$, using different numbers of intervals on the Spiral dataset.

## 6 Conclusions

Missing attribute values are very common in real-world datasets. In this work, we have proposed a new version of the k-means clustering algorithm that can deal with datasets with missing values.

We also derived a formula for the *mean* of a given dataset when it contains points with missing values. This formula provides the *mean* of the known values. The computational complexity for computing the *mean* using the $MD_E$ distance is the same as that of the standard *mean* using the Euclidian distance.

We integrated the $MD_E$ distance function and the *mean* computation within the framework of the k-means clustering algorithm proposed three different ways to compute the centroids while taking into account the associations between the data points and the centroids. This is in contrast to the other basic methods for dealing with missing values, which do not take into account these associations.

We conducted experiments using the k-means clustering algorithm on six standard datasets. Our k-means methods outperformed the standard methods for datasets with missing values.

These proposed methods are general and can be used as part of any algorithm that computes the distance between data points or *means*. Moreover, they can be used for different datasets in different application areas. So, our future vision is devoted to incorporating this distance function within other clustering and classification algorithms.

## References


1. L. Abedallah and I. Shimshoni. Mean Shift Clustering Algorithm for Data with Missing Values. In *Proc. of 14th Int. Conf. of DaWaK*, pages 426–438, 2014.
2. A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. Review: a gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.
3. Eirola Emil, Lendasse Amaury, Vandewalle Vincent, and Biernacki Christophe. Mixture of gaussians for distance estimation with missing data. *Neurocomputing*, 131:32–42, 2014.
4. Z. Ghahramani and M. Jordan. Learning from incomplete data. *Technical Report, MIT AI Lab Memo*, (1509), 1995.
5. Jerzy Grzymala-Busse and Ming Hu. A comparison of several approaches to missing attribute values in data mining. In *Proc. Rough Sets and Current Trends in Computing*, pages 378–385. Springer, 2001.
6. Lynette Hunt and Murray Jorgensen. Mixture model clustering for mixed data with missing information. *Computational Statistics & Data Analysis*, 41(3):429–440, 2003.
7. Joseph G Ibrahim, Ming-Hui Chen, Stuart R Lipsitz, and Amy H Herring. Missing-data methods for generalized linear models: A comparative review. *Journal of the American Statistical Association*, 100(469):332–346, 2005.
8. Roderick JA Little. Missing-data adjustments in large surveys. *Journal of Business & Economic Statistics*, 6(3):287–296, 1988.
9. Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.
10. S. Lloyd. Least squares quantization in PCM. *IEEE Trans. Information Theory*, 28:129–137, 1982.
11. J. B. MacQueen. Some methods for classification and analysis of multivariate observations. *Proc. of the 5th Symposium on Math, Statistics, and Probability*, pages 281–297, 1967.
12. Matteo Magnani. Techniques for dealing with missing data in knowledge discovery tasks. *Obtido http://magnanim.web.cs.unibo.it/index.html*, 15(01):2007, 2004.
13. W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
14. H. Steinhaus. Sur la division des corp materiels en parties. *Bull, Acad Polon Science*, 4(3):801–804, 1956.
15. Speech University of Eastern Finland and Image Processing Unit. Clustering dataset, http://cs.joensuu.fi/sipu/datasets/.
16. S. Zhang, Z. Qin, C.X. Ling, and S. Sheng. Missing is useful": missing values in cost-sensitive decision trees. *IEEE Trans. on KDE*, 17(12):1689–1693, 2005.