

# Weighted Clustering: Towards Solving the User's Dilemma

Margareta Ackerman<sup>1a</sup>, Shai Ben-David<sup>b</sup>, Simina Branzei<sup>c</sup>, David Loker<sup>d</sup>

<sup>a</sup>*Department of Computer Science and Engineering  
Santa Clara University  
Santa Clara, CA*

<sup>b</sup>*Cheriton School of Computer Science  
University of Waterloo  
Waterloo, ON*

<sup>c</sup>*Perdue University  
Department of Computer Science  
West Lafayette, IN*

<sup>d</sup>*WaveAI  
Sunnyvale, CA*

---

## Abstract

This paper makes a major step towards addressing a long-standing challenge in cluster analysis, known as *the user's dilemma*, which is the problem of selecting an appropriate clustering algorithm for a specific task. A formal approach for addressing this challenge relies on the identification of succinct, user-friendly properties that capture formal differences amongst clustering techniques. While helpful for gaining insight into the nature of clustering paradigms, there is a theory-practice gap that has so far limited the utility of this approach: Formal properties typically highlight advantages of classical linkage-based algorithms, while practical experience shows that center-based methods are preferable for many applications. We present simple new properties that delineate core differences between common clustering paradigms and overcome this theory-practice gap. The properties we present give a formal understanding of the advantages of center-based approaches for some applications and insight into when different clustering paradigms should be used. These properties address how sensitive algorithms are to changes in element frequencies, which we capture in a generalized setting where every element is associated with a real-valued weight. To complement extensive formal analysis, we discuss how these properties can be applied in practice.

---

---

<sup>1</sup>Corresponding author

## 1. Introduction

Clustering is one of the most useful data mining tools, utilized in a wide range of applications, from astronomy to zoology. Yet, its theoretical underpinnings lag behind its wild application. Even the fairly basic problem of which algorithm to select for a given application, known as “the user’s dilemma,” is left to ad hoc solutions. Issues of running time complexity and space usage remain the primary considerations when choosing clustering techniques. Yet, for clustering, such considerations are insufficient: Different clustering algorithms often produce radically different results on the same input, and as such, differences in their input-output behavior should take precedence over computational concerns.

The user’s dilemma has been tackled since the 1970s ([23, 48]). A formal approach to this problem proposes that we rely on succinct mathematical properties that reveal fundamental differences in the input-output behaviour of different clustering algorithms (see, for example, [23, 17, 5]). However there is an important theory-practice disconnect: Virtually all the properties proposed in this framework highlight the advantages of linkage-based methods, most of which are satisfied by single-linkage – an algorithm that often performs poorly in practice. While prior properties give important insight into clustering algorithms, relying on them for selecting an algorithm inevitably results in a linkage-based technique. According to these properties, there is never a reason to choose, say, algorithms based on the  $k$ -means objective ([42]), which often performs well in practice.

Of course, practitioners of clustering have known for a long time that, for many applications, variations of the  $k$ -means method outperform classical linkage-based techniques. Unfortunately, a lack of clarity as to why this is the case leaves the “the user’s dilemma” largely unsolved. Despite continued efforts to find better clustering methods, the ambiguous nature of clustering precludes the existence of a single algorithm that will be suited for all applications. As such, generally successful methods, such as popular algorithms for the  $k$ -means objective, are nevertheless ill-suited for some applications. To this end, it is necessary for users of clustering to understand how clustering paradigms differ in their input-output behavior.

Unfortunately, informal recommendations are not sufficient. Many such recommendations advise to use  $k$ -means when the true clusters are spherical and to apply single-linkage when they may possess arbitrary shape. Such advice can be insufficient, as clustering users know that single-linkage can fail to detect arbitrary-shaped clusters, and  $k$ -means does not always succeed when clusters are spherical. Further insight comes from viewing data as a mixture model (when variations of  $k$ -means, particularly EM, are known to perform well), but most users don’t have information on how their data is generated when applying clustering to real data. Another common way to differentiate clustering methods is to partition them into partitional and hierarchical and to imply that users should choose algorithms based on this consideration. Although the format of the output is important, it does not address fundamental differences, as most clustering approaches can be expressed in both frameworks.<sup>2</sup>

---

<sup>2</sup>For example,  $k$ -means can be reconfigured to output a hierarchy using Ward’s method and Bisecting

The lack of formal understanding of the key differences between clustering paradigms leaves users at a loss when selecting algorithms. In practice, many users give up on clustering altogether when a single algorithm that had been successful on a different data set fails to attain a satisfactory clustering on new data. Without a clear understanding as to the manner in which algorithms differ can prevent user from selecting appropriate methods, or even sampling a few diverse techniques.

This paper identifies the first set of properties that differentiates between some of the most popular clustering methods, while highlighting advantages of  $k$ -means (and similar) methods. The properties go to the heart of the difference between some clustering methods. In an effort to propose properties that are useful in practice, we have distilled our analysis of clustering methods to several elegant properties, which are easy to understand and apply in practice. We believe this to be a substantial step forward towards solving the user’s dilemma.

The properties proposed in this paper distill to the rather basic concept of how different clustering methods react to element duplication. This leads to three surprisingly simple categories, each highlighting when some clustering paradigms should be used over others. To this end, we consider a generalization of the notion of element duplication by casting the clustering problem in the weighted setting, where each element is associated with a real valued weight.

This generalized setting enables more accurate representation of some clustering instances. Consider, for instance, vector quantization, which aims to find a compact encoding of signals that has low expected distortion. The accuracy of the encoding is most important for signals that occur frequently. With weighted data, such a consideration is easily captured by having the weights of the points represent signal frequencies. When applying clustering to facility allocation, such as the placement of police stations in a new district, the distribution of the stations should enable quick access to most areas in the district. However, the accessibility of different landmarks to a station may have varying importance. The weighted setting enables a convenient method for prioritizing certain landmarks over others.

It is important to note that while the weighted setting has independent interest, the classical un-weighted clustering model can be readily mapped to the weighted framework by replacing duplicates with integer weights representing the number of occurrences of each data point. As such, all of the results presented here also apply in the traditional setting, which does not require points to be weighted.

We formulate intuitive properties that may allow a user to select an algorithm based on how it treats weighted data (or, element duplicates). These surprisingly simple properties are able to distinguish between classes of clustering techniques and clearly delineate instances in which some methods are preferred over others, without having to resort to assumptions about how the data may have been generated. As such, they may aid in the clustering selection process for clustering users at all levels of expertise.

Based on these properties we obtain a classification of clustering algorithms into three

---

$k$ -mean, and classical hierarchical methods can be terminated using a variety of termination conditions ([30]) to obtain a single partition instead of a hierarchy.

categories: those that are affected by weights on all data sets, those that ignore weights, and those methods that respond to weights on some configurations of the data but not on others. Among the methods that always respond to weights are several well-known algorithms, such as  $k$ -means and  $k$ -median. On the other hand, algorithms such as single-linkage, complete-linkage, and min-diameter ignore weights.

From a theoretical perspective, perhaps the most notable is the last category. We find that methods belonging to that category are robust to weights when data is sufficiently clusterable, and respond to weights otherwise. Average-linkage as well as the well-known spectral objective function, ratio cut, both fall into this category. We characterize the precise conditions under which these methods are influenced by weights.

Finally, we include a discussion of how the properties introduced in this work can help to find suitable clustering algorithms in practice, as well as the relation between the framework proposed here to clustering constructs such as validity indices and clusterability measures. While not intended to solve the user’s dilemma in all instances, the properties that we present offer clear guidance on an important dimension differentiating clustering techniques, and, in some cases, are sufficient for identifying a suitable algorithm. We hope that this line of research will continue to grow to include other properties that, when considered together, will provide a comprehensive solution to the user’s dilemma.

### 1.1. Related Work

Clustering algorithms are usually analyzed in the context of unweighted data. The weighted clustering framework was briefly considered in the early 70s, but wasn’t developed further until now. Fisher and Van Ness [23] introduced several properties of clustering algorithms. Among these, they include “point proportion admissibility,” which requires that the output of an algorithm should not change if any points are duplicated. They then observe that a few algorithms are point proportion admissible. However, clustering algorithms can display a much wider range of behaviours on weighted data than merely satisfying or failing to satisfy point proportion admissibility. We carry out the first extensive analysis of clustering on weighted data, characterizing the precise conditions under which algorithms respond to weight.

In addition, Wright [48] proposed a formalization of cluster analysis consisting of eleven axioms. In two of these axioms, the notion of mass is mentioned. Namely, that points with zero mass can be treated as non-existent, and that multiple points with mass at the same location are equivalent to one point with weight the sum of the masses. The idea of mass has not been developed beyond stating these axioms in their work.

After these early works on foundations of clustering, interest in axioms and properties of clustering saw a notable resurgence since the early 2000s following the publication of Kleinberg’s [30] famous impossibility theorem, where he proved that three seemingly natural properties cannot be simultaneously satisfied by any clustering method. Follow up work often sought to overcome the impossibility result by either considering alternate clustering settings or considering modifications to Kleinberg’s axioms. For example, it has been shown that Kleinberg’s axioms are consistent in the context of clustering quality measures [4], and

variations of his original “consistency” axiom have been shown to resolve the impossibility [20, 33]. These axioms have also been studied in the context of the  $k$ -means algorithm, often showing that variations of some of the axioms hold for this popular method [31, 47, 32].

Following Kleinberg’s work, most property analysis of clustering techniques, including research that does not focus on this axioms, has been in the unweighted clustering settings for both partitional [2, 17, 5, 16] and hierarchical clustering [1]. Previous work in this line of research centers on classical linkage-based methods and their advantages. Particularly well-studied is the single-linkage algorithm, for which there are multiple property-based characterizations, showing that single-linkage is the unique algorithm that satisfies several sets of properties [27, 17, 19]. The entire family of linkage-based algorithms was characterized [2, 1], differentiating those algorithms from other clustering paradigms by presenting some of the advantages of those methods. In addition, previous property-based taxonomies in this line of work highlight the advantages of linkage-based methods [5, 23]. Despite the emphasis on linkage-based methods in the theory literature, empirical studies and user experience have shown that, in many cases, other techniques produce more useful clusterings than those obtained by classical linkage-based methods. Here we propose categories that distinguish between clustering paradigms while also showing when other techniques, such as popular center-based methods, may be more appropriate. Our work falls under the broader framework of aiming to bridge theory and practice in clustering. See [12] for further discussion on the disconnect between theory and practice as it relates to the clustering user dilemma.

Another related direction is the study of robustness criteria, which aims to identify whether clustering techniques are reliable by studying whether they remain (fairly) consistent under various conditions (ex. data perturbation or the presence of outliers). Robustness criteria were studied for  $k$ -means [34],  $k$ -center [10] density-based techniques [28] and spectral clustering [38], amongst other methods [36]. This type of analysis is often approached from the perspective that robustness is inherently desirable. We emphasize that our analysis of weight sensitivity and robustness is approached from a different standpoint. We analyse how algorithms react to changes in weights in order to help users determine which clustering technique is best applied to their specific application, as different weight responses are appropriate under different circumstances. In particular, weight robustness is only appropriate under certain circumstances.

Finally, another related area of research is the study of clusterability, which aims to evaluate the degree of inherent cluster structure in data. While many formal methods for clusterability evaluation have been proposed, a formal analysis reveals that they are often pairwise inconsistent [3]. As such, a variation of the user’s dilemma arises, whereby the user needs to choose which clusterability method they are to apply for the application at hand. Recently, there has been progress towards the clusterability version of the user’s dilemma [6], with an extensive analysis of clusterability methods and recommendations for how to select a clusterability technique. In this paper, we rely on notions of clusterability to elucidate the behaviour of several weight-considering techniques.

## 2. Preliminaries

A *weight function*  $w$  over  $X$  is a function  $w : X \rightarrow R^+$ , mapping elements of  $X$  to positive real numbers. Given a domain set  $X$ , denote the corresponding weighted domain by  $w[X]$ , thereby associating each element  $x \in X$  with weight  $w(x)$ . A *dissimilarity function* is a symmetric function  $d : X \times X \rightarrow R^+ \cup \{0\}$ , such that  $d(x, y) = 0$  if and only if  $x = y$ .<sup>3</sup> We consider *weighted data sets* of the form  $(w[X], d)$ , where  $X$  is some finite domain set,  $d$  is a dissimilarity function over  $X$ , and  $w$  is a weight function over  $X$ .

A  $k$ -*clustering*  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$  of a domain set  $X$  is a partition of  $X$  into  $1 < k < |X|$  disjoint, non-empty subsets of  $X$  where  $\cup_i C_i = X$ . A *clustering* of  $X$  is a  $k$ -clustering for some  $1 < k < |X|$ . To avoid trivial partitions, clusterings that consist of a single cluster, or where every cluster has a unique element, are not permitted.

Denote the *weight of a cluster*  $C_i \in \mathcal{C}$  by  $w(C_i) = \sum_{x \in C_i} w(x)$ . For a clustering  $\mathcal{C}$ , let  $|\mathcal{C}|$  denote the number of clusters in  $\mathcal{C}$ . For  $x, y \in X$  and clustering  $\mathcal{C}$  of  $X$ , write  $x \sim_{\mathcal{C}} y$  if  $x$  and  $y$  belong to the same cluster in  $\mathcal{C}$  and  $x \not\sim_{\mathcal{C}} y$ , otherwise.

A *partitional weighted clustering algorithm* is a function that maps a data set  $(w[X], d)$  and an integer  $1 < k < |X|$  to a  $k$ -clustering of  $X$ .

A *binary hierarchy*  $\mathcal{H}$  of  $X$  is a pair  $(T, M)$  where  $T$  is a strictly binary rooted tree and  $M : \text{leaves}(T) \rightarrow X$  is a bijection. Since the current paper is only concerned with binary hierarchies, for brevity, we refer to a “binary hierarchy” as a “hierarchy.” A *hierarchical weighted clustering algorithm* is a function that maps a data set  $(w[X], d)$  to a hierarchy of  $X$ . A set  $C_0 \subseteq X$  is a cluster in a hierarchy  $\mathcal{H} = (T, M)$  of  $X$  if there exists a node  $x$  in  $T$  so that  $C_0 = \{M(y) \mid y \text{ is a leaf and a descendent of } x\}$ . Two hierarchies of  $X$  are *equivalent* if they contain the same clusters, and  $[\mathcal{H}]$  denotes the equivalence class of hierarchy  $\mathcal{H}$ .

For a hierarchical weighted clustering algorithm  $\mathcal{A}$ , a clustering  $\mathcal{C} = \{C_1, \dots, C_k\}$  *appears in*  $\mathcal{A}(w[X], d)$  if  $C_i$  is a cluster in  $\mathcal{A}(w[X], d)$  for all  $1 \leq i \leq k$ . A partitional algorithm  $\mathcal{A}$  outputs clustering  $\mathcal{C}$  on  $(w[X], d)$  if  $\mathcal{A}(w[X], d, |\mathcal{C}|) = \mathcal{C}$ .

For the remainder of this paper, unless otherwise stated, we will use the term “clustering algorithm” for “weighted clustering algorithm”.

The range of a partitional algorithm on a data set is the set of clusterings it outputs on that data over all weight functions.

**Definition 1** (Range (Partitional)). *Given a partitional clustering algorithm  $\mathcal{A}$ , a data set  $(X, d)$ , and  $1 \leq k \leq |X|$ , let  $\text{range}(\mathcal{A}(X, d, k)) = \{\mathcal{C} \mid \exists w \text{ such that } \mathcal{C} = \mathcal{A}(w[X], d)\}$ , i.e. the set of  $k$ -clusterings that  $\mathcal{A}$  outputs on  $(X, d)$  over all possible weight functions.*

The range of a hierarchical algorithm on a data set is the set of hierarchies it outputs on that data over all weight functions.

---

<sup>3</sup>We note that the requirement that  $d(x, y) = 0 \implies x = y$  is not always included in the definition of dissimilarity functions. When this condition holds, then the dissimilarity function can be referred to as *proper*. We utilize proper dissimilarity functions throughout this paper.

**Definition 2** (Range (Hierarchical)). *Given a hierarchical clustering algorithm  $\mathcal{A}$  and a data set  $(X, d)$ , let  $\text{range}(\mathcal{A}(X, d)) = \{\mathcal{H} \mid \exists w \text{ such that } \mathcal{H} = \mathcal{A}(w[X], d)\}$ , i.e. the set of hierarchies that  $\mathcal{A}$  outputs on  $(X, d)$  over all possible weight functions.*

### 3. Basic Categories

Different clustering algorithms exhibit radically different response to weighted data. In this section we introduce a formal categorization of clustering algorithms based on their response to weights. This categorization identifies fundamental differences between clustering paradigms, while highlighting when some of the more empirically successful methods should be used. These properties can assist clustering users in selecting a suitable method by simply considering how an algorithm should react to element duplication for a given application. After introducing the three categories, we present a classification of some well-known clustering methods according to their response to weight, summarized in Table 1.

#### 3.1. Weight Robust Algorithms

We first introduce the notion of “weight robust” algorithms. Weight robustness requires that the output of the algorithm be unaffected by changes of element weights (or, the number of occurrences of each point in the unweighted setting). This category is closely related to “point proportion admissibility” by [23].

**Definition 3** (Weight Robust (Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-robust if for all  $(X, d)$  and  $1 < k < |X|$ ,  $|\text{range}(\mathcal{A}(X, d, k))| = 1$ .*

The definition in the hierarchical setting is analogous.

**Definition 4** (Weight Robust (Hierarchical)). *A hierarchical algorithm  $\mathcal{A}$  is weight-robust if for all  $(X, d)$ ,  $|\text{range}(\mathcal{A}(X, d))| = 1$ .*

At first glance, this appears to be a desirable property. A weight robust algorithm is able to keep sight on the geometry of the data without being “distracted” by weights, or element duplicates. Indeed, when a similar property was proposed by [23], it was presented as a desirable characteristic.

Yet, notably, few algorithms possess it (particularly single-linkage, complete-linkage, and min-diameter), while most techniques, including those with a long history of empirical success, fail this property. This brings into question how often is weight-robustness a desirable characteristic, and suggests, that at least for some applications, sensitivity to weights may be an advantage. Significantly, the popular  $k$ -means and similar methods fail weight robustness in a strong sense, being “weight sensitive.”

#### 3.2. Weight Sensitive Algorithms

We now introduce the definition of “weight sensitive” algorithms.

**Definition 5** (Weight Sensitive (Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-sensitive if for all  $(X, d)$  and  $1 < k < |X|$ ,  $|\text{range}(\mathcal{A}(X, d, k))| > 1$ .*

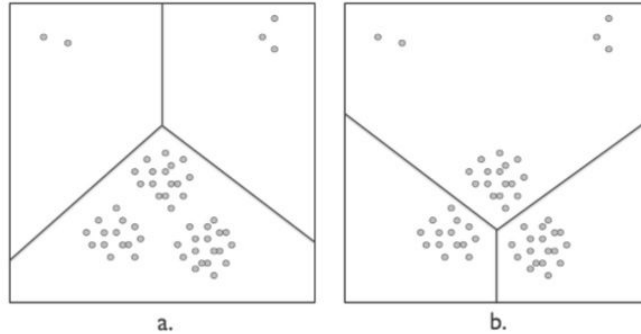


Figure 1: An example of different cluster structures in the same data, illustrating inherent tradeoffs between separation and cluster balance. The clustering on the left finds inherent structure in the data by identifying well-separated partitions, while the clustering on the right discovers structure in the data by focusing on the dense region, achieving more balanced cluster sizes. The correct partitioning depends on the application at hand.

The definition is analogous for hierarchical algorithms.

**Definition 6** (Weight Sensitive (Hierarchical)). *A hierarchical algorithm  $\mathcal{A}$  is weight-sensitive if for all  $(X, d)$  where  $|X| > 2$ ,  $|\text{range}(\mathcal{A}(X, d))| > 1$ .*

Note that this definition is quite extreme. It means that *no matter how well-separated* the clusters are, the output of a weight-sensitive algorithm can be altered by modifying some of the weights. That is, a weight-sensitive algorithm will miss arbitrarily well-separated clusters, for some weighting of its elements. In practice, weight sensitive algorithms tend to aim for balanced cluster sizes, and so prioritize a balance in cluster sizes (or, sum of cluster weights) over separation between clusters.

While weight-robust algorithms are interested exclusively in the geometry of the data, weight-sensitive techniques have two potentially conflicting considerations: The weight of the points and the geometry of the data. For instance, consider the data in Figure 1, which has two distinct 3-clusterings, one which provides superior separation between clusters, and another in which clusters sizes are balanced. Note how different are the two clusterings from each other. All the weight-sensitive methods we consider select the clustering on the right, as it offers more balanced clusters. On the other hand, the weight-robust methods we study pick the clustering on the left hand side, as it offers better cluster separation.

Another way we could think of weight sensitive algorithm is that, unlike weight-robust methods, weight sensitive algorithms allow the weights to alter the geometry of the data. In contrast, weight robust techniques do not allow the weights of the points to “interfere” with the underlying geometry.

It is important to note that there appear to be implications of these categories that apply to data that is neither weighted nor contains element duplicates. Considering the algorithms we analyzed (summarized in Table 1), the behaviour we observe on element duplicates extends to “near-duplicates,” which are closely positioned elements. Furthermore,



the weight response of an algorithm sheds light on how it treats dense regions. In particular, weight sensitive algorithms have a tendency to “zoom in” on areas of high density, effectively ignoring sparse regions, as shown on the right-hand side of Figure 1.

Finally, the last category considered here offers a compromise between weight-robustness and weight-sensitivity. We refer to this category as *weight considering*.

### 3.3. Weight Considering Algorithms

**Definition 7** (Weight Considering (Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-considering if*

- *There exist  $(X, d)$  and  $1 < k < |X|$  so that  $|\text{range}(\mathcal{A}(X, d, k))| = 1$ , and*
- *There exist  $(X, d)$  and  $1 < k < |X|$  so that  $|\text{range}(\mathcal{A}(X, d, k))| > 1$ .*

The definition carries over to the hierarchical setting as follows.

**Definition 8** (Weight Considering (Hierarchical)). *A hierarchical algorithm  $\mathcal{A}$  is weight-considering if*

- *There exist  $(X, d)$  with  $|X| > 2$  so that  $|\text{range}(\mathcal{A}(X, d))| = 1$ , and*
- *There exist  $(X, d)$  with  $|X| > 2$  so that  $|\text{range}(\mathcal{A}(X, d))| > 1$ .*

Weight considering methods appear to have the best of both worlds. The weight-considering algorithms that we analyzed (average-linkage and ratio-cut), ignore weights when clusters are sufficiently well-separated and otherwise takes them into consideration. Yet, it is important to note that this is only desirable in some instances. For example, when cluster balance is critical, as may be the case for market segmentation, weight-sensitive methods may be preferable over weight considering ones. On the other hand, when the distribution may be highly biased, as is often the case for phylogenetic analysis, weight-considering methods may offer a satisfactory compromise between weight-sensitivity and weight-robustness. Weight-considering algorithms can detect well-separated clusters, possibly of radically different sizes, without entirely disregarding weights. Notably, of all the classical clustering algorithms studied here, average-linkage, a weight-considering technique, is the only one that is commonly applied to phylogenetic analysis.

The following table presents a classification of classical clustering methods based on these three categories. Sections 4 and 5 provide the proof for the results summarized below. In addition, these sections also characterize precisely when the weight-considering techniques studied here respond to weights. An expanded table that includes heuristics, with a corresponding analysis, is included in Section 6.

To formulate clustering algorithms in the weighted setting, we consider their behaviour on data that allows duplicates. If we allow for semi-definite dissimilarity functions, given a data set  $(X, d)$ , elements  $x, y \in X$  are duplicates if  $d(x, y) = 0$  and  $d(x, z) = d(y, z)$  for all  $z \in X$ . In a Euclidean space, duplicates correspond to elements that occur at the same location. We obtain the weighted version of a data set by de-duplicating the data, and

	<b>Partitional</b>	<b>Hierarchical</b>
<b>Weight Sensitive</b>	$k$ -means, $k$ -medoids $k$ -median, Min-sum	Ward’s method Bisecting $k$ -means
<b>Weight Considering</b>	Ratio-cut	Average-linkage
<b>Weight Robust</b>	Min-diameter $k$ -center	Single-linkage Complete-linkage

Table 1: A classification of clustering algorithms based on their response to weighted data.

associating every element with a weight equaling the number of duplicates of that element in the original data. The weighted version of an algorithm partitions the resulting weighted data in the same manner that the unweighted version partitions the original data. As shown throughout the paper, this translation leads to natural formulations of weighted algorithms. Please note that since we utilize proper dissimilarity functions, where  $d(x, y) = 0 \implies x = y$ , all duplicates in our framework must be captured through weights.

#### 4. Partitional Methods

In this section, we show that partitional clustering algorithms respond to weights in a variety of ways. Many popular partitional clustering paradigms, including  $k$ -means,  $k$ -median, and min-sum, are weight sensitive. It is easy to see that methods such as min-diameter and  $k$ -center are weight-robust. We begin by analysing the behaviour of a spectral objective function ratio cut, which exhibits interesting behaviour on weighted data by responding to weight unless data is highly structured.

##### 4.1. Ratio-Cut Clustering

We investigate the behaviour of a spectral objective function, ratio-cut ([45]), on weighted data. Instead of a dissimilarity function, spectral clustering relies on a *similarity function*, which maps pairs of domain elements to non-negative real numbers that represent how alike the elements are. The ratio-cut of a clustering  $\mathcal{C}$  is:

$$\text{cost}_{rcut}(\mathcal{C}, w[X], s) = \frac{1}{2} \sum_{C_i \in \mathcal{C}} \frac{\sum_{x \in C_i, y \in X \setminus C_i} s(x, y) \cdot w(x) \cdot w(y)}{\sum_{x \in C_i} w(x)}.$$

The ratio-cut clustering function is:

$$\text{rcut}(w[X], s, k) = \arg \min_{\mathcal{C}; |\mathcal{C}|=k} \text{cost}_{rcut}(\mathcal{C}, w[X], s).$$

We prove that this function ignores data weights only when the data satisfies a very strict notion of clusterability. To characterize precisely when ratio-cut responds to weights, we first present a few definitions.

A clustering  $\mathcal{C}$  of  $(w[X], s)$  is *perfect* if for all  $x_1, x_2, x_3, x_4 \in X$  where  $x_1 \sim_{\mathcal{C}} x_2$  and  $x_3 \not\sim_{\mathcal{C}} x_4$ ,  $s(x_1, x_2) > s(x_3, x_4)$ .  $\mathcal{C}$  is *separation-uniform* if there exists  $\lambda$  so that for all  $x, y \in X$  where  $x \not\sim_{\mathcal{C}} y$ ,  $s(x, y) = \lambda$ . Note that neither condition depends on the weight function.

We show that whenever a data set has a clustering that is both perfect and separation-uniform, then ratio-cut uncovers that clustering, which implies that ratio-cut is not weight-sensitive. Note that, in particular, these conditions are satisfied when all within-cluster similarities are set to zero. On the other hand, we show that ratio-cut does respond to weights when either condition fails.

**Lemma 1.** *If, given a data set  $(X, d)$ ,  $1 < k < |X|$  and some weight function  $w$ , ratio-cut outputs a  $k$ -clustering  $\mathcal{C}$  that is not separation-uniform and where every cluster has more than a single point, then  $|\text{range}(\text{ratio-cut}(X, d))| > 1$ .*

*Proof.* We consider two cases.

**Case 1:** There is a pair of clusters with different similarities between them. Then there exist  $C_1, C_2 \in \mathcal{C}$ ,  $x \in C_1$ , and  $y \in C_2$  so that  $s(x, y) \geq s(x, z)$  for all  $z \in C_2$ , and there exists  $a \in C_2$  so that  $s(x, y) > s(x, a)$ .

Let  $w$  be a weight function such that  $w(x) = W$  for some sufficiently large  $W$  and weight 1 is assigned to all other points in  $X$ . Since we can set  $W$  to be arbitrarily large, when looking at the cost of a cluster, it suffices to consider the dominant term in terms of  $W$ . We will show that we can improve the cost of  $\mathcal{C}$  by moving a point from  $C_2$  to  $C_1$ . Note that moving a point from  $C_2$  to  $C_1$  does not affect the dominant term of clusters other than  $C_1$  and  $C_2$ . Therefore, we consider the cost of these two clusters before and after rearranging points between these clusters.

Let  $A = \sum_{a \in C_2} s(x, a)$  and let  $m = |C_2|$ . Then the dominant term, in terms of  $W$ , of the cost of  $C_2$  is  $W \left( \frac{A}{m} \right)$ . The cost of  $C_1$  approaches a constant as  $W \rightarrow \infty$ .

Now consider clustering  $\mathcal{C}'$  obtained from  $\mathcal{C}$  by moving  $y$  from cluster  $C_2$  to cluster  $C_1$ . The dominant term in the cost of  $C_2$  becomes  $W \left( \frac{A-s(x,y)}{m-1} \right)$ , and the cost of  $C_1$  approaches a constant as  $W \rightarrow \infty$ . By choice of  $x$  and by choice of  $y$ , if  $\frac{A-s(x,y)}{m-1} < \frac{A}{m}$  then  $\mathcal{C}'$  has lower loss than  $\mathcal{C}$  when  $W$  is large enough. The inequality  $\frac{A-s(x,y)}{m-1} < \frac{A}{m}$  holds when  $\frac{A}{m} < s(x, y)$ , and the latter holds by choice of  $x$  and by choice of  $y$ .

**Case 2:** The similarities between every pair of clusters are the same. However, there are clusters  $C_1, C_2, C_3 \in \mathcal{C}$ , so that the similarities between  $C_1$  and  $C_2$  are greater than the ones between  $C_1$  and  $C_3$ . Let  $a$  and  $b$  denote the similarities between  $C_1, C_2$  and  $C_1, C_3$ , respectively.

Let  $x \in C_1$  and  $w$  a weight function, such that  $w(x) = W$  for large  $W$ , and weight 1 is assigned to all other points in  $X$ . The dominant term comes from clusters going into  $C_1$ , specifically edges that include point  $x$ . The dominant term of the contribution of cluster  $C_3$  is  $Wb$  and the dominant term of the contribution of  $C_2$  is  $Wa$ , totaling  $Wa + Wb$ .

Now consider clustering  $\mathcal{C}'$  obtained from clustering  $\mathcal{C}$  by merging  $C_1$  with  $C_2$ , and splitting  $C_3$  into two clusters (arbitrarily). The dominant term of the clustering comes from clusters other than  $C_1 \cup C_2$ , and the cost of clusters outside  $C_1 \cup C_2 \cup C_3$  is unaffected. The dominant term of the cost of the two clusters obtained by splitting  $C_3$  is  $Wb$  for each, for a total of  $2Wb$ . However, the factor of  $Wa$  that  $C_2$  previously contributed is no longer present. This replaces the coefficient of the dominant term from  $a + b$  to  $2b$ , which improved the cost of the clustering because  $b < a$ .  $\square$

**Lemma 2.** *If, given a data set  $(X, d)$ ,  $1 < k < |X|$ , and some weight function  $w$ , ratio-cut outputs a clustering  $\mathcal{C}$  that is not perfect and where every cluster has more than a single point, then  $|\text{range}(\text{ratio-cut}(X, d, k))| > 1$ .*

*Proof.* If  $\mathcal{C}$  is also not separation-uniform, then Lemma 1 can be applied, and so we can assume that  $\mathcal{C}$  is separation-uniform. Then there exists a within-cluster similarity in  $\mathcal{C}$  that is smaller than some between-cluster similarity, and all between cluster similarities are the same. Specifically, there exist clusters  $C_1$  and  $C_2$ , such that all the similarities between  $C_1$  and  $C_2$  are  $a$ , and there exist  $x, y \in C_1$  such that  $s(x, y) < a$ . Let  $b = s(x, y)$ .

Let  $w$  be a weight function such that  $w(x) = W$  for large  $W$ , and weight 1 is assigned to all other points in  $X$ . Then the dominant term in the cost of  $C_2$  is  $Wa$ , which comes from the cost of points in cluster  $C_2$  going to point  $x$ . The cost of  $C_1$  approaches a constant as  $W \rightarrow \infty$ .

Consider the clustering  $\mathcal{C}'$  obtained from  $\mathcal{C}$  by completely re-arranging all points in  $C_1, C_2 \in \mathcal{C}$  as follows:

1. Let  $\{y, z\} = C'_1$  be one cluster for any  $z \in C_2$ .
2. Let  $C'_2 = (C_1 \cup C_2) \setminus C'_1$  be the new second cluster.

The dominant term in  $C'_1$ , which comes from the cost of points in cluster  $C'_1$  going to point  $x$ , is  $\left(\frac{a+b}{2}\right)W$ , which is smaller than  $Wa$  since  $a > b$ . Note that the cost of each cluster outside of  $C'_1 \cup C'_2$  remains unchanged. Since  $x \in C'_1$ , the cost of  $C'_1$  approaches a constant as  $W \rightarrow \infty$ . Therefore,  $\text{cost}_{\text{rcut}}(C', w[X], s)$  is smaller than  $\text{cost}_{\text{rcut}}(\mathcal{C}, w[X], s)$  when  $W$  is sufficiently large.  $\square$

**Lemma 3.** *Given any data set  $(w[X], s)$  and  $1 < k < |X|$  that has a perfect, separation-uniform  $k$ -clustering  $\mathcal{C}$ ,  $\text{ratio-cut}(w[X], s, k) = \mathcal{C}$ .*

*Proof.* Let  $(w[X], s)$  be a weighted data set, with a perfect, separation-uniform clustering  $\mathcal{C} = \{C_1, \dots, C_k\}$ . Recall that for any  $Y \subseteq X$ ,  $w(Y) = \sum_{y \in Y} w(y)$ . Then:

$$\begin{aligned}
\text{cost}_{\text{rcut}}(\mathcal{C}, w[X], s) &= \frac{1}{2} \sum_{i=1}^k \frac{\sum_{x \in C_i} \sum_{y \in \overline{C_i}} s(x, y) w(x) w(y)}{\sum_{x \in C_i} w(x)} \\
&= \frac{1}{2} \sum_{i=1}^k \frac{\sum_{x \in C_i} \sum_{y \in \overline{C_i}} \lambda w(x) w(y)}{\sum_{x \in C_i} w(x)} \\
&= \frac{\lambda}{2} \sum_{i=1}^k \frac{\sum_{y \in \overline{C_i}} w(y) \sum_{x \in C_i} w(x)}{\sum_{x \in C_i} w(x)} = \frac{\lambda}{2} \sum_{i=1}^k \sum_{y \in \overline{C_i}} w(y) \\
&= \frac{\lambda}{2} \sum_{i=1}^k w(\overline{C_i}) = \frac{\lambda}{2} \sum_{i=1}^k [w(X) - w(C_i)] \\
&= \frac{\lambda}{2} \left( kw(X) - \sum_{i=1}^k w(C_i) \right) = \frac{\lambda}{2} (k-1)w(X).
\end{aligned}$$

Consider any other clustering,  $\mathcal{C}' = \{C'_1, \dots, C'_k\} \neq \mathcal{C}$ . Since  $\mathcal{C}$  is both perfect and separation-uniform, all between-cluster similarities in  $\mathcal{C}$  are  $\lambda$ , and all within-cluster similarities are greater than  $\lambda$ . From here it follows that all pair-wise similarities in the data are at least  $\lambda$ . Since  $\mathcal{C}'$  is a  $k$ -clustering different from  $\mathcal{C}$ , it must differ from  $\mathcal{C}$  on at least one between-cluster edge, so that edge must be greater than  $\lambda$ . Thus the cost of  $\mathcal{C}'$  is:

$$\begin{aligned}
\text{cost}_{\text{rcut}}(\mathcal{C}', w[X], s) &= \frac{1}{2} \sum_{i=1}^k \frac{\sum_{x \in C'_i} \sum_{y \in \overline{C'_i}} s(x, y) w(x) w(y)}{\sum_{x \in C'_i} w(x)} \\
&> \frac{1}{2} \sum_{i=1}^k \frac{\sum_{x \in C'_i} \sum_{y \in \overline{C'_i}} \lambda w(x) w(y)}{\sum_{x \in C'_i} w(x)} \\
&= \frac{\lambda}{2} (k-1)w(X) = \text{cost}_{\text{rcut}}(\mathcal{C}).
\end{aligned}$$

Thus clustering  $\mathcal{C}'$  has a higher cost than  $\mathcal{C}$ . □

It follows that ratio-cut responds to weights on all data sets except those where it is possible to obtain cluster separation that is both very large and highly uniform. This implies that ratio cut is highly unlikely to be unresponsive to weights in practice.

Formally, we have the following theorem, which gives sufficient conditions for when ratio-cut ignores weights, as well conditions that make this function respond to weights.

**Theorem 4.1.** *Given any  $(X, d)$  and  $1 < k < |X|$ ,*

1. *if  $(X, d)$  has a clustering that is both perfect and separation uniform, then*

$$|\text{range}(\text{Ratio-cut}(X, s, k))| = 1,$$

*and*

2. if  $\text{range}(\text{Ratio-cut}(X, s, k))$  includes a clustering  $\mathcal{C}$  that is not perfect or not separation-uniform, and has no singleton clusters, then  $|\text{range}(\text{Ratio-cut}(X, s, k))| > 1$ .

*Proof.* The result follows by Lemma 1, Lemma 2, and Lemma 3.  $\square$

#### 4.2. *K-Means*

Many popular partitional clustering paradigms, including  $k$ -means (see [42] for a detailed exposition of this popular objective function and related algorithms),  $k$ -median, and the min-sum objective ([40]), are weight sensitive. Moreover, these algorithms satisfy a stronger condition. By modifying weights, we can make these algorithms separate any set of points. We call such algorithms *weight-separable*.

**Definition 9** (Weight Separable). *A partitional clustering algorithm  $\mathcal{A}$  is weight-separable if for any data set  $(X, d)$  and any  $S \subset X$ , where  $2 \leq |S| \leq k$ , there exists a weight function  $w$  so that  $x \not\sim_{\mathcal{A}(w[X], d, k)} y$  for all distinct  $x, y \in S$ .*

Note that every weight-separable algorithm is also weight-sensitive.

**Lemma 4.** *If a clustering algorithm  $\mathcal{A}$  is weight-separable, then  $\mathcal{A}$  is weight-sensitive.*

*Proof.* Given any  $(X, d)$  and weight function  $w$  over  $X$ , let  $\mathcal{C} = \mathcal{A}(w[X], d, k)$ . Select points  $x$  and  $y$  where  $x \sim_{\mathcal{C}} y$ . Since  $\mathcal{A}$  is weight-separable, there exists  $w'$  so that  $x \not\sim_{\mathcal{A}(w'[X], d, k)} y$ , and so  $\mathcal{A}(w'[X], d, k) \neq \mathcal{C}$ . It follows that for any  $(X, d)$ ,  $|\text{range}(\mathcal{A}(X, d))| > 1$ .  $\square$

$K$ -means is perhaps the most popular clustering objective function, with cost:

$$k\text{-means}(\mathcal{C}, w[X], d) = \sum_{C_i \in \mathcal{C}} \sum_{x \in C_i} d(x, \text{cnt}(C_i))^2 \cdot w(x),$$

where  $\text{cnt}(C_i)$  denotes the center of mass of cluster  $C_i$ . The  $k$ -means objective function finds a clustering with minimal  $k$ -means cost. We show that  $k$ -means is weight-separable, and thus also weight-sensitive.

**Theorem 4.2.** *The  $k$ -means objective function is weight-separable.*

*Proof.* Consider any  $S \subseteq X$ . Let  $w$  be a weight function over  $X$  where  $w(x) = W$  if  $x \in S$ , for large  $W$ , and  $w(x) = 1$  otherwise. As shown by [37], the  $k$ -means objective function is equivalent to

$$\sum_{C_i \in \mathcal{C}} \frac{\sum_{x, y \in C_i} d(x, y)^2 \cdot w(x) \cdot w(y)}{w(C_i)}.$$

Let  $m_1 = \min_{x, y \in X} d(x, y)^2 > 0$ ,  $m_2 = \max_{x, y \in X} d(x, y)^2$ , and  $n = |X|$ . Consider any  $k$ -clustering  $\mathcal{C}$  where all the elements in  $S$  belong to distinct clusters. Then we have:

$$k\text{-means}(\mathcal{C}, w[X], d) < km_2 \left( n + \frac{n^2}{W} \right).$$

On the other hand, given any  $k$ -clustering  $\mathcal{C}'$  where at least two elements of  $S$  appear in the same cluster,  $k\text{-means}(\mathcal{C}', w[X], d) \geq \frac{W^2 m_1}{W+n}$ . Since

$$\lim_{W \rightarrow \infty} \frac{k\text{-means}(\mathcal{C}', w[X], d)}{k\text{-means}(\mathcal{C}, w[X], d)} = \infty,$$

$k$ -means separates all the elements in  $S$  for large enough  $W$ . □

Min-sum is another well known objective function and it minimizes the expression:

$$\sum_{C_i \in \mathcal{C}} \sum_{x, y \in C_i} d(x, y) \cdot w(x) \cdot w(y).$$

**Theorem 4.3.** *Min-sum is weight-separable.*

*Proof.* Let  $(X, d)$  be any data set and  $1 < k < |X|$ . Consider any  $S \subseteq X$  where  $1 < |S| \leq k$ . Let  $w$  be a weight function over  $X$  where  $w(x) = W$  if  $x \in S$ , for large  $W$ , and  $w(x) = 1$  otherwise. Let  $m_1 = \min_{x, y \in X} d(x, y)$  be the minimum dissimilarity in  $(X, d)$ , and let  $m_2 = \max_{x, y \in X} d(x, y)$  be the maximum dissimilarity in  $(X, d)$ .

Then the cost of any cluster that includes two elements of  $S$  is at least  $m_1 W^2$ , while the cost of a cluster that includes at most one element of  $S$  is less than  $m_2 |X|(|X| + W)$ . So when  $W$  is large enough, min-sum selects a partition where no two elements of  $S$  appear in the same cluster. □

Several other objective functions similar to  $k$ -means are also weight-separable. We show that  $k$ -median and  $k$ -medoids are weight sensitive by analysing center-based approaches that use exemplars from the data as cluster centers (as opposed to any elements in the underlying space). Given a set  $T \subseteq X$ , define  $C(T)$  to be the clustering obtained by assigning every element in  $X$  to the closest element (“center”) in  $T$ .

Exemplar-based clustering is defined as follows.

**Definition 10** (Exemplar-based). *An algorithm  $\mathcal{A}$  is exemplar-based if there exists a function  $f : \mathbb{R}^+ \cup \{0\} \rightarrow \mathbb{R}^+ \cup \{0\}$  such that for all  $(w[X], d)$ ,  $\mathcal{A}(w[X], d, k) = C(T)$  where*

$$T = \arg \min_{T \subseteq X; |T|=k} \sum_{x \in X} w(x) f(\min_{y \in T} d(x, y)).$$

Note that when  $f$  is the identity function, we obtain  $k$ -median, and when  $f(x) = x^2$  we obtain  $k$ -medoids.

**Theorem 4.4.** *Every exemplar-based clustering function is weight-separable.*

*Proof.* Consider any  $S \subseteq X$  with  $|S| \leq k$ . For all  $x \in S$ , set  $w(x) = W$  for some large value  $W$  and set all other weights to 1. Recall that a clustering has between 2 and  $|X| - 1$  clusters. Consider any clustering  $\mathcal{C}$  where some distinct elements  $x, y \in S$  belong to the same cluster  $C_i \in \mathcal{C}$ . Since at most one of  $x$  or  $y$  can be the center of  $C_i$ , the cost of  $\mathcal{C}$  is at least  $W \cdot \min_{x_1, x_2 \in C_i} f(d(x_1, x_2))$ . Observe that  $f(d(x_1, x_2)) > 0$ .

Consider any clustering  $\mathcal{C}'$  where all the elements in  $S$  belong to distinct clusters. If a cluster contains a unique element  $x$  of  $S$ , then its cost is constant in  $W$  if  $x$  is the cluster center, and at least  $W \cdot \min_{x_1, x_2 \in X} f(d(x_1, x_2))$  if  $x$  is not the center of the cluster. This shows that if  $W$  is large enough, then every element of  $S$  will be a cluster center, and so the cost of  $\mathcal{C}'$  would be independent of  $W$ . So when  $W$  is large, clusterings that separate elements of  $S$  have lower cost than those that merge any points in  $S$ .  $\square$

We have the following corollary.

**Corollary 1.** *The  $k$ -median and  $k$ -medoids objective functions are weight-separable and weight-sensitive.*

## 5. Hierarchical Algorithms

Similar to partitional methods, hierarchical algorithms also exhibit a wide range of responses to weights. We show that Ward’s method [46], a successful linkage-based algorithm, as well as popular divisive hierarchical methods, are weight sensitive. On the other hand, it is easy to see that the linkage-based algorithms single-linkage and complete-linkage are both weight robust, as was observed in [23].

Average-linkage, another popular linkage-based method, exhibits more nuanced behaviour on weighted data. When a clustering satisfies a reasonable notion of clusterability, average-linkage detects that clustering irrespective of weights. On the other hand, this algorithm responds to weights on all other clusterings.

### 5.1. Average Linkage

Linkage-based algorithms start by placing each element in its own cluster, and proceed by repeatedly merging the “closest” pair of clusters until the entire hierarchy is constructed. To identify the closest clusters, these algorithms use a linkage function that maps pairs of clusters to a real number. Formally, a *linkage function* is a function  $\ell : \{(X_1, X_2, d, w) \mid d, w \text{ over } X_1 \cup X_2\} \rightarrow \mathbb{R}^+$ .

Average-linkage is one of the most popular linkage-based algorithms (commonly applied in bioinformatics under the name Unweighted Pair Group Method with Arithmetic Mean). Recall that  $w(X) = \sum_{x \in X} w(x)$ . The average-linkage linkage function is

$$\ell_{AL}(X_1, X_2, d, w) = \frac{\sum_{x \in X_1, y \in X_2} d(x, y) \cdot w(x) \cdot w(y)}{w(X_1) \cdot w(X_2)}.$$

To study how average-linkage responds to weights, we give a relaxation of the notion of a perfect clustering.

**Definition 11** (Nice). *A clustering  $\mathcal{C}$  of  $(w[X], d)$  is nice if for all  $x_1, x_2, x_3 \in X$  where  $x_1 \sim_{\mathcal{C}} x_2$  and  $x_1 \not\sim_{\mathcal{C}} x_3$ ,  $d(x_1, x_2) < d(x_1, x_3)$ .*



The concept of a “nice” clustering is closely related to the notion of a “strong” or “Apresjan” cluster [7, 13, 15] and previously appeared under the name “strict separation” [11].

As for a perfect clustering, being a nice clustering is independent of weights. Note that all perfect clusterings are nice, but not all nice clusterings are perfect. A hierarchy is *nice* if all clusterings that appear in it are nice.

We present a complete characterisation of the way that average-linkage (AL) responds to weights.

**Theorem 5.1.** *Given  $(X, d)$ ,  $|\text{range}(AL(X, d))| = 1$  if and only if  $(X, d)$  has a nice hierarchy.*

*Proof.* We first show that if a data set has a nice hierarchy, then this is the hierarchy that average-linkage outputs. Note that the property of being nice is independent of the weight function. So, the set of nice clusterings of any data set  $(w[X], d)$  is invariant to the weight function  $w$ . Lemma 5 shows that, for every  $(w[X], d)$ , every nice clustering in  $(w[X], d)$  appears in the hierarchy produced by average-linkage.

Let  $(X, d)$  be a data set that has a nice hierarchy  $\mathcal{H}$ . We would like to show that average-linkage outputs that hierarchy. Let  $\mathbb{C}$  be the set of all nice clusterings of  $(X, d)$ . Let  $\mathcal{L} = \{c \mid \exists \mathcal{C} \in \mathbb{C} \text{ such that } c \in \mathcal{C}\}$ . That is,  $\mathcal{L}$  is the set of all clusters that appear in some nice clustering of  $(X, d)$ .

Since  $\mathcal{H}$  is a nice hierarchy of  $(X, d)$ , all clusterings that appear in it are nice, and so it contains all the clusters in  $\mathcal{L}$  and no additional clusters. In order to satisfy the condition that every nice clustering of  $(X, d)$  appears in the hierarchy,  $\mathcal{H}_{AL}$ , produced by average-linkage,  $\mathcal{H}_{AL}$  must have all clusters in  $\mathcal{L}$ .

Since a hierarchy is a strictly binary tree, any hierarchy of  $(X, d)$  has exactly  $|X| - 1$  inner nodes. In particular, all hierarchies of the same data set have exactly the same number of inner nodes. This implies that  $\mathcal{H}_{AL}$  has the same clusters as  $\mathcal{H}$ , so  $\mathcal{H}_{AL}$  is equivalent to  $\mathcal{H}$ .

Now, let  $(X, d)$  be a data set that does not have a nice hierarchy. Then, given any  $w$  over  $X$ , there is a clustering  $\mathcal{C}$  that is not nice that appears in  $AL(w[X], d)$ . Lemma 6 shows that if a clustering that is not nice appears in  $AL(w[X], d)$ , then  $|\text{range}(AL(X, d))| > 1$ .  $\square$

Theorem 5.1 follows from the two lemmas below. It is important to note that Lemma 5 below was previously proven by Bryant and Berry [15]. For completion, we include a short proof utilizing the notation used throughout this paper.

**Lemma 5 (Originally proved in [15]).** *Given any weighted data set  $(w[X], d)$ , if  $\mathcal{C}$  is a nice clustering of  $(X, d)$ , then  $\mathcal{C}$  is in the hierarchy produced by average-linkage on  $(w[X], d)$ .*

*Proof.* Consider a nice clustering  $\mathcal{C} = \{C_1, \dots, C_k\}$  over  $(w[X], d)$ . It suffices to show that for any  $1 \leq i < j \leq k$ ,  $X_1, X_2 \subseteq C_i$  where  $X_1 \cap X_2 = \emptyset$  and  $X_3 \subseteq C_j$ ,  $\ell_{AL}(X_1, X_2, d, w) < \ell_{AL}(X_1, X_3, d, w)$ . It can be shown that

$$\ell_{AL}(X_1, X_2, d, w) \leq \frac{\sum_{x_1 \in X_1} w(x_1) \cdot \max_{x_2 \in X_2} d(x_1, x_2)}{w(X_1)}$$

and

$$\ell_{AL}(X_1, X_3, d, w) \geq \frac{\sum_{x_1 \in X_1} w(x_1) \cdot \min_{x_3 \in X_3} d(x_1, x_3)}{w(X_1)}.$$

Since  $\mathcal{C}$  is nice, it follows that

$$\min_{x_3 \in X_3} d(x_1, x_3) > \max_{x_2 \in X_2} d(x_1, x_2)$$

Thus  $\ell_{AL}(X_1, X_3, d, w) > \ell_{AL}(X_1, X_2, d, w)$ , which completes the proof.  $\square$

**Lemma 6.** *For any data set  $(X, d)$  and any weight function  $w$  over  $X$ , if a clustering that is not nice appears in  $AL(w[X], d)$ , then  $|\text{range}(AL(X, d))| > 1$ .*

*Proof.* Let  $(X, d)$  be a data set so that a clustering  $\mathcal{C}$  that is not nice appears in  $AL(w[X], d)$  for some weight function  $w$  over  $X$ . We construct  $w'$  so that  $\mathcal{C} \notin AL(w'[X], d)$ , which would show that  $|\text{range}(AL(X, d))| > 1$ .

Since  $\mathcal{C}$  is not nice, there exist  $1 \leq i, j \leq k$ ,  $i \neq j$ , and  $x_1, x_2 \in C_i$ ,  $x_1 \neq x_2$ , and  $x_3 \in C_j$ , so that  $d(x_1, x_2) > d(x_1, x_3)$ .

Now, define weight function  $w'$  as follows:  $w'(x) = 1$  for all  $x \in X \setminus \{x_1, x_2, x_3\}$ , and  $w'(x_1) = w'(x_2) = w'(x_3) = W$ , for some large value  $W$ . We argue that when  $W$  is sufficiently large,  $\mathcal{C}$  is not a clustering in  $AL(w'[X], d)$ .

By way of contradiction, assume that  $\mathcal{C}$  is a clustering in  $AL(w'[X], d)$  for any setting of  $W$ . Then there is a step in the algorithm where clusters  $X_1$  and  $X_2$  merge, where  $X_1, X_2 \subset C_i$ ,  $x_1 \in X_1$ , and  $x_2 \in X_2$ . At this point, there is some cluster  $X_3 \subseteq C_j$  so that  $x_3 \in X_3$ .

We compare  $\ell_{AL}(X_1, X_2, d, w')$  and  $\ell_{AL}(X_1, X_3, d, w')$ . First, note that

$$\ell_{AL}(X_1, X_2, d, w') = \frac{W^2 d(x_1, x_2) + \alpha_1 W + \alpha_2}{W^2 + \alpha_3 W + \alpha_4}$$

for some non-negative real valued  $\alpha_i$ s. Similarly, we have that for some non-negative real-valued  $\beta_i$ :

$$\ell_{AL}(X_1, X_3, d, w') = \frac{W^2 d(x_1, x_3) + \beta_1 W + \beta_2}{W^2 + \beta_3 W + \beta_4}$$

Dividing by  $W^2$ , we see that  $\ell_{AL}(X_1, X_3, d, w') \rightarrow d(x_1, x_3)$  and  $\ell_{AL}(X_1, X_2, d, w') \rightarrow d(x_1, x_2)$  as  $W \rightarrow \infty$ , and so the result holds since  $d(x_1, x_3) < d(x_1, x_2)$ . Therefore average linkage merges  $X_1$  with  $X_3$ , thus cluster  $C_i$  is never formed, and so  $\mathcal{C}$  is not a clustering in  $AL(w'[X], d)$ . It follows that  $|\text{range}(AL(X, d))| > 1$ ,  $\square$

## 5.2. Ward's Method

Ward's method is a highly effective clustering algorithm ([46]), which, at every step, merges the clusters that will yield the minimal increase to the sum-of-squares error (the  $k$ -means objective function). Recall that  $\text{cnt}(C_i)$  denotes the center of mass of cluster  $C_i$ .

Then, the linkage function for Ward's method is

$$\ell_{Ward}(X_1, X_2, d, w) = \frac{w(X_1) \cdot w(X_2) \cdot d(\text{cnt}(X_1), \text{cnt}(X_2))^2}{w(X_1) + w(X_2)},$$

where  $X_1$  and  $X_2$  are disjoint subsets (clusters) of  $X$ .

**Theorem 5.2.** *Ward’s method is weight sensitive.*

*Proof.* Consider any data set  $(X, d)$  and any clustering  $\mathcal{C}$  output by Ward’s method on  $(X, d)$ . Let  $x, y \in X$  be any distinct points that belong to the same cluster in  $\mathcal{C}$ . Let  $w$  be the weight function that assigns a large weight  $W$  to points  $x$  and  $y$ , and weight 1 to all other elements.

Since Ward’s method is linkage-based, it starts off by placing every element in its own cluster. We will show that when  $W$  is large enough, it is prohibitively expensive to merge a cluster that contains  $x$  with a cluster that contains point  $y$ . Therefore, there is no cluster in the hierarchy produced by Ward’s method that contains both points  $x$  and  $y$ , other than the root, and so  $\mathcal{C}$  is not a clustering in that hierarchy. This would imply that  $|\text{range}(\text{Ward}(X, d))| > 1$ .

At some point in the execution of Ward’s method,  $x$  and  $y$  must belong to different clusters. Let  $C_i$  be a cluster that contains  $x$ , and cluster  $C_j$  a cluster that contains point  $y$ . Then  $\ell_{\text{Ward}}(C_i, C_j, d, w) \rightarrow \infty$  as  $W \rightarrow \infty$ . On the other hand, whenever at most one of  $C_i$  or  $C_j$  contains an element of  $\{x, y\}$ ,  $\ell_{\text{Ward}}(C_i, C_j, d, w)$  approaches some constant as  $W \rightarrow \infty$ . This shows that when  $W$  is sufficiently large, a cluster containing  $x$  is merged with a cluster containing  $y$  only at the last step of the algorithm, when forming the root of the hierarchy. □

### 5.3. Divisive Algorithms

The class of divisive clustering algorithms is a well-known family of hierarchical algorithms, which construct the hierarchy by using a top-down approach. This family of algorithms includes the popular bisecting  $k$ -means algorithm. We show that a class of algorithms that includes bisecting  $k$ -means consists of weight-sensitive methods.

Given a node  $x$  in hierarchy  $(T, M)$ , let  $C(x)$  denote the cluster represented by node  $x$ . That is,  $C(x) = \{M(y) \mid y \text{ is a leaf and a descendent of } x\}$ .

Informally, a  $\mathcal{P}$ -Divisive algorithm is a hierarchical clustering algorithm that uses a partitional clustering algorithm  $\mathcal{P}$  to recursively divide the data set into two clusters until only single elements remain. Formally, a  $\mathcal{P}$ -divisive algorithm is defined as follows.

**Definition 12** ( $\mathcal{P}$ -Divisive). *A hierarchical clustering algorithm  $\mathcal{A}$  is  $\mathcal{P}$ -Divisive with respect to a partitional clustering algorithm  $\mathcal{P}$ , if for all  $(X, d)$ , we have  $\mathcal{A}(w[X], d) = (T, M)$ , such that for all non-leaf nodes  $x$  in  $T$  with children  $x_1$  and  $x_2$ ,  $\mathcal{P}(w[C(x)], d, 2) = \{C(x_1), C(x_2)\}$ .*

We obtain bisecting  $k$ -means by setting  $\mathcal{P}$  to  $k$ -means. Other natural choices for  $\mathcal{P}$  include min-sum, and exemplar-based algorithms such as  $k$ -median. As shown above, many of these partitional algorithms are weight-separable. We show that whenever  $\mathcal{P}$  is weight-separable, then  $\mathcal{P}$ -Divisive is weight-sensitive.

**Theorem 5.3.** *If  $\mathcal{P}$  is weight-separable then the  $\mathcal{P}$ -Divisive algorithm is weight-sensitive.*

*Proof.* Given any non-trivial clustering  $\mathcal{C}$  output by the  $\mathcal{P}$ -Divisive algorithm, consider any pair of elements  $x$  and  $y$  that are placed within the same cluster of  $\mathcal{C}$ . Since  $\mathcal{P}$  is weight separating, there exists a weight function  $w$  so that  $\mathcal{P}$  separates points  $x$  and  $y$ . Then  $\mathcal{P}$ -Divisive splits  $x$  and  $y$  in the first step, directly below the root, and clustering  $\mathcal{C}$  is never formed. □

## 6. Heuristic Approaches

We have seen how weights affect various algorithms that optimize different clustering objectives. Since optimizing a clustering objective is usually NP-hard, heuristics are used in practice. In this section, we consider several common heuristical clustering approaches, and show how they respond to weights.

We note that there are many algorithms that aim to find high quality partitions for popular objective functions. For the  $k$ -means objective alone many different algorithms have been proposed, most of which provide different initializations for Lloyd’s method. For example, [43] studied a dozen different initializations. There are many other algorithms based on the  $k$ -means objective functions, some of the most notable being  $k$ -means++ ([9]) and the Hochbaum-Schmoys initialization ([24]) studied, for instance, by [18]. As such, this section is not intended as a comprehensive analysis of all available heuristics, but rather it shows how to analyze such heuristics, and provides a classification for some of the most popular approaches.

To define the categories in the randomized setting, we need to modify the definition of *range*. Given a randomized, partitional clustering algorithm  $\mathcal{A}$  and a data set  $(X, d)$ , the *randomized range* is

$$\text{randRange}(\mathcal{A}(X, d)) = \{\mathcal{C} \mid \forall \epsilon < 1 \exists w \text{ such that } P(\mathcal{A}(w[X], d) = \mathcal{C}) > 1 - \epsilon\}.$$

That is, the randomized range is the set of clusterings that are produced with arbitrarily high probability, when we can modify weights.

The categories describing an algorithm’s behaviour on weighted data are defined as previously, but using randomized range.

**Definition 13** (Weight Sensitive (Randomized Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-sensitive if for all  $(X, d)$  and  $1 < k < |X|$ ,  $|\text{randRange}(\mathcal{A}(X, d, k))| > 1$ .*

**Definition 14** (Weight Robust (Randomized Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-robust if for all  $(X, d)$  and  $1 < k < |X|$ ,  $|\text{randRange}(\mathcal{A}(X, d, k))| = 1$ .*

**Definition 15** (Weight Considering (Randomized Partitional)). *A partitional algorithm  $\mathcal{A}$  is weight-considering if*

- *There exist  $(X, d)$  and  $1 < k < |X|$  so that  $|\text{randRange}(\mathcal{A}(X, d, k))| = 1$ , and*
- *There exist  $(X, d)$  and  $1 < k < |X|$  so that  $|\text{randRange}(\mathcal{A}(X, d, k))| > 1$ .*

### 6.1. Partitioning Around Medoids (PAM)

In contrast with the Lloyd method and  $k$ -means, PAM is a heuristic for exemplar-based objective functions such as  $k$ -medoids, which chooses data points as centers (thus it is not required to compute centers of mass). As a result, this approach can be applied to arbitrary data, not only to normed vector spaces.

*Partitioning around medoids* (PAM) is given an initial set of  $k$  centers,  $T$ , and changes  $T$  iteratively to find a “better” set of  $k$  centers. This is done by swapping out centers for

other points in the data set and computing the cost function:  $\sum_{c \in T} \sum_{x \in X \setminus T, c \sim_{\mathcal{C}} x} d(x, c) \cdot w(x)$ . Each iteration performs a swap only if a better cost is possible, and it stops when no changes are made [29].

Note that our results in this section hold regardless of how the initial  $k$  centers are chosen from  $X$ .

**Theorem 6.1.** *PAM is weight-separable.*

*Proof.* Let  $T = \{x_1, \dots, x_l\}$  be  $l$  points that we want to separate, where  $2 \leq l \leq k$ . Let  $\{m_1, \dots, m_k\} \subset X$  be  $k$  centers chosen by the PAM initialization, and denote by  $\mathcal{C} = \{c_1, \dots, c_k\}$  the clustering induced by the corresponding centers. Set  $w(x_i) = W, \forall x_i \in T$  for some large  $W$ . We first note that any optimal clustering  $\mathcal{C}^*$  sets the points in  $T$  as the centers. The cost of  $\mathcal{C}^*$  is constant as a function of  $W$ , while every clustering with a different set of centers has a cost proportional to  $W$ , which can be made arbitrarily high by increasing  $W$ .

Assume, by contradiction, that the algorithm stops at a clustering  $\mathcal{C}$  that does not separate all the points in  $T$ . Then, there exists a cluster  $c_i \in \mathcal{C}$  such that  $|c_i \cap T| \geq 2$ . Thus,  $c_i$  contributes a factor of  $\alpha \cdot W$  to the cost, for some  $\alpha > 0$ . Further, there exists a cluster  $c_j \in \mathcal{C}$  such that  $c_j \cap T = \emptyset$ . Then the cost of  $\mathcal{C}$  can be further decreased, by a quantity proportional to  $W$ , by assigning one of the heavy non-medoid from  $c_i$  to be the center of  $c_j$ , which is a contradiction. Thus, the algorithm cannot stop before setting all the heavy points as cluster centers.  $\square$

## 6.2. Lloyd method

The Lloyd method is a heuristic commonly used for uncovering clusterings with low  $k$ -means objective cost. The Lloyd algorithm can be combined with different approaches for seeding the initial centers. In this section, we start by considering the following deterministic seeding methods.

**Definition 16** (Lloyd Method). *Given  $k$  points (centers)  $\{c_1, \dots, c_k\}$  in the space, assign every element of  $X$  to its closest center. Then compute the centers of mass of the resulting clusters by summing the elements in each cluster and dividing by the number of elements in that partition, and assign every element to its closest new center. Continue until no change is made in one iteration.*

With the Lloyd method, the dissimilarity (or, distance) to the center can be both the  $\ell_1$ -norm or squared.

First, we consider the case when the  $k$  initial centers are chosen in a deterministic fashion. For example, one deterministic seeding approach involves selecting the  $k$ -furthest centers (see, for example, [5]).

**Theorem 6.2.** *Let  $\mathcal{A}$  represent the Lloyd method with some deterministic seeding procedure. Consider any data set  $(X, d)$  and  $1 < k < |X|$ . If there exists a clustering  $\mathcal{C} \in \text{range}(\mathcal{A}(X, d, k))$  that is not nice, then  $|\text{range}(\mathcal{A}(X, d, k))| > 1$ .*

*Proof.* For any seeding procedure, since  $\mathcal{C}$  is in the range of  $\mathcal{A}(X, d)$ , there exists a weight function  $w$  so that  $\mathcal{C} = \mathcal{A}(w[X], d)$ .

Since  $\mathcal{C}$  is not nice, there exist points  $x_1, x_2, x_3 \in X$  where  $x_1 \sim_{\mathcal{C}} x_2$ ,  $x_1 \not\sim_{\mathcal{C}} x_3$ , but  $d(x_1, x_3) < d(x_1, x_2)$ . Construct weight function  $w'$  such that  $w'(x) = 1$  for all  $x \in X \setminus \{x_2, x_3\}$ , and  $w'(x_2) = w'(x_3) = W$ , for some constant  $W$ .

If for some value of  $W$ ,  $\mathcal{A}(w'[X], d, k) \neq \mathcal{C}$ , then we're done. Otherwise,  $\mathcal{A}(w'[X], d, k) = \mathcal{C}$  for all values of  $W$ . But if  $W$  is large enough, the center of mass of the cluster containing  $x_1$  and  $x_2$  is arbitrarily close to  $x_2$ , and the center of mass of the cluster containing  $x_3$  is arbitrarily close to  $x_3$ . But since  $d(x_1, x_3) < d(x_1, x_2)$ , the Lloyd method would assign  $x_1$  and  $x_3$  to the same cluster. Thus, when  $W$  is sufficiently large,  $\mathcal{A}(w'[X], d, k) \neq \mathcal{C}$ .  $\square$

We also show that for a deterministic, weight-independent initialization, if the Lloyd method outputs a nice clustering  $\mathcal{C}$ , then this algorithm is robust to weights on that data.

**Theorem 6.3.** *Let  $\mathcal{A}$  represent the Lloyd method with some weight-independent deterministic seeding procedure. Given  $(X, d)$ , if there exists a nice clustering in the range( $\mathcal{A}(X, d)$ ), then  $\mathcal{A}$  is weight robust on  $(X, d)$ .*

*Proof.* Since the initialization is weight-independent,  $\mathcal{A}$  will find the same initial centers on any weight function. Given a nice clustering, the Lloyd method does not modify the clustering. If the seeding method were not weight-independent, it may seed in a way that may prevent the Lloyd method from finding  $\mathcal{C}$  for some weight function.  $\square$

**Corollary 2.** *Let  $\mathcal{A}$  represent the Lloyd method initialized with furthest centroids. For any  $(X, d)$  and  $1 < k < |X|$ ,  $|\text{range}(\mathcal{A}(X, d, k))| = 1$  if and only if there exists a nice  $k$ -clustering of  $(X, d)$ .*

### 6.2.1. $k$ -means++

The  $k$ -means++ algorithm, introduced by Arthur and Vassilvitskii ([9]) is the Lloyd algorithm with a randomized initialization method that aims to place the initial centers far apart from each other. This algorithm has been demonstrated to perform very well in practice.

Let  $D(x)$  denote the shortest dissimilarity from a point  $x$  to the closest center already chosen. The  $k$ -means++ algorithm chooses the initial center uniformly at random, and then  $x$  is selected as the next center with probability  $\frac{D(x)^2 w(x)}{\sum_y D(y)^2 w(y)}$  until  $k$  centers have been chosen.

**Theorem 6.4.**  *$k$ -means++ is weight-separable.*

*Proof.* If any  $k$  points  $\{x_1, \dots, x_k\}$  are assigned sufficiently high weight  $W$ , then the first center will be one of these points with arbitrarily high probability. The next center will also be selected with arbitrarily high probability if  $W$  is large enough, since for all  $y \notin \{x_1, \dots, x_k\}$ , the probability of selecting  $y$  can be made arbitrarily small when  $W$  is large enough.  $\square$

The same argument works for showing that the Lloyd method is weight-separable when the initial centers are selected uniformly at random (Randomized Lloyd). An expanded classification of clustering algorithms that includes heuristics is given in Table 1 below.

	<b>Partitional</b>	<b>Hierarchical</b>	<b>Heuristics</b>
<b>Weight Sensitive</b>	$k$ -means, $k$ -medoids $k$ -median, Min-sum	Ward’s method Bisecting $k$ -means	Randomized Lloyd, PAM, $k$ -means++
<b>Weight Considering</b>	Ratio-cut	Average-linkage	Lloyd with Furthest centroids
<b>Weight Robust</b>	Min-diameter $k$ -center	Single-linkage Complete-linkage	

Table 2: A classification of clustering algorithms based on their response to weighted data expanded to include several popular heuristic methods.

## 7. Discussion and Future Work

### 7.1. Applying The Properties in Practice

We now delve deeper into how the properties presented in this paper can be applied to select clustering methods in practice. Application of the properties to the user’s dilemma reduces to the following simple question: “Should element duplicates impact the clustering outcome?” It may be helpful to consider extreme cases, with elements replicated hundreds or thousands of times. This can make the answer clearer than if one considers duplicating elements a small number of times.

Clustering typically takes place in the traditional unweighted framework, rather than the weighed setting. As discussed in Section 3, our properties apply both in the weighted and standard, unweighted models. Recall that to convert a method from the unweighted to the weighted setting, assign each element a weight based on the number of times that it occurs. That is, a non-replicated element is given a weight of 1, while elements with multiple occurrences are assigned a weight corresponding to their number of occurrences.

While the weighted properties directly map to precisely replicated elements, the concept extends more broadly. Intuitively, the weighted categories capture how algorithms treat similar items and handle dense regions. Algorithms often treat points in close proximity in a similar manner to perfect replicates. As such, when applying these properties in practice, it is worth considering not only how perfect replicates should be handled with respect to weight sensitivity, but also whether near-duplicates, or the presence of a large number of highly similar items, should impact the clustering.

If the user believes that high sensitivity to element duplicates fits the needs of the application, then a weight-sensitive method should be used. On the other hand, low sensitivity to weight should lead the user to consider either weight robust or weight considering techniques. In practice, weight considering methods tend to sufficiently prioritize geometry over density (or, weight) to justify their use when the user does not wish duplicates to have substantial impact on the resultant clustering.

We now examine several examples, demonstrating how clustering methods can be selected for these applications based on the desired type of weight sensitivity.

- **Phylogenetic analysis.** Phylogeny aims to understand the evolutionary relationship between organisms. It has been applied to a wide range of organisms, from mammals to the evolution of the HIV virus. Given the focus on evolutionary relationships, phylogenetic analysis often takes place in the hierarchical clustering setting. However, as discussed in the introduction, choosing hierarchical over partitional techniques is itself not sufficient, as clustering techniques can be converted between these two settings.

The ground truth for a phylogenetic application is independent of element duplicates. That is, whether a sample contains one instance of each element, or many replicates of some of the elements, has no impact on the true evolutionary relationship between the data points. As such, sensitivity to weight is not desirable for phylogenetic methods.

Indeed, phylogenetic analysis relies heavily on average-linkage, referred to as UPGMA in the phylogenetic literature [41]. This pattern persists despite the overwhelming popularity of  $k$ -means and related techniques.

- **Collaborative filtering.** Collaborative filtering seeks to provide a user with item recommendations (such as songs, TV shows, or physical products), based on their own past purchasing behaviour and/or rankings and, critically, those of others [35, 44]. Clustering can be applied to collaborative filtering in order to identify users with similar shopping behaviour, and as such recommend users products prevalent in their cluster.

What kind of weight response makes sense for a collaborative filtering application of cluster analysis? If a single shopper has a particular shopping pattern, that shouldn't have much impact on the recommendations made to others. On the other hand, many shoppers exhibiting the same, or similar, behaviour is an important factor to consider. As such, weight sensitive methods are appropriate for this application. Other aspects, such as an algorithm's ability to handle sparse data, should also be considered when selecting algorithms for collaborative filtering.

- **Market segmentation.** Market segmentation looks to identify similar groups of customers or "personas." If we consider a classical variation, with a large number of customers with similar purchasing power, then a weight sensitive approach may be appropriate in order to identify common customer types. Indeed, an analysis of clustering methods applied to market segmentation reveals that Ward's method was the most frequently applied hierarchical technique, while the classical  $k$ -means algorithm was most popular amongst partitional methods, both of which are weight sensitive [21].

On the other hand, in applications such a fundraising, where potential donors may bring radically different value to the organization, it may be essential that a small cluster of high value donors not be overshadowed by dense regions. Weight sensitive methods, such as  $k$ -means, run the risk of occluding a low-weight cluster, treating it as an outlier in favour of partitioning dense regions. As such, weight considering methods could be explored. This example highlights that even within the same broad application, it is important to consider the specific needs of a given use case.



The applications above illustrate how the properties introduced here apply to common applications. Viewing the user’s dilemma from a weighted properties perspective often explains common trends in the application of cluster analysis in specific domains, such as the popularity of average-linkage in phylogenetic analysis, and dominance of Ward’s method and  $k$ -means in market segmentation.

However, clustering applications are highly diverse, and may not fit neatly into an overarching, common application. To further illustrate the utility of our approach, we mention an application of cluster analysis where the weighted approach was used to select an algorithm for clustering the echolocation clicks of toothed whale species [39], in a project involving one of the co-authors of this paper. The aim of the work was to separate the data into clusters corresponding to distinct whale species, where shortage of labeled data made cluster analysis a promising alternative to supervised techniques. To select a clustering method, recommendations made earlier in this section were utilized, considering whether data duplicates should impact the solution. Since the true clusters, corresponding to whale species, are independent of the number of occurrences of any echolocation click, low weight sensitivity was desired, and average-linkage was utilized.

We make one final recommendation, which applies when there is insufficient information to address the simple question regarding desired behaviour on data duplicates. One of the primary points of failure when selecting clustering techniques is reliance on a single technique. For example, a researcher may choose to use Lloyd’s method or a related algorithm as a single clustering approach. When it fails, the researcher may conclude that clustering is not an appropriate data analysis method for the application at hand. A key takeaway from the analysis presented in the current paper is the partition of clustering techniques into three distinct categories: Weight robust, sensitive, and considering. In the absence of additional information, techniques from distinct weight categories should be explored.

## *7.2. Relation to other considerations relevant to the user’s dilemma*

The main contribution of this work is a simple property-based solution that assists users in the selection of a clustering algorithm suited to their application. The strength of this solution is that it both relies on solid theory and is applicable in practice.

Many other properties have been previously proposed in order to solve the user’s dilemma. These include many crisp, simple properties [23, 17, 5]. However, most such properties from previous work have a major shortcoming: They elevate single-linkage, an algorithm that fails for many applications. Several authors have demonstrated single-linkage to be the only method to satisfy sets of desirable properties [27, 17, 19].

Meanwhile, the literature offers no simple, usable properties that showcase the strengths of highly useful methods, such as  $k$ -means and related techniques. This presents a major theory-practice gap - the theory suggests that single-linkage is the algorithm of choice, while practice shows that methods such as  $k$ -means are more often appropriate.

Less drastic, but also important, is the fact that techniques like average-linkage are often preferred to single-linkage in practice (this is the case, for example, in Phylogeny, where average-linkage goes under the name of UPGMA [41]), and yet the property-based literature provides little to no insight as to when to apply average-linkage over single-linkage.

The properties proposed here offer clear, direct guidance on how to select amongst clustering algorithms, in a manner that clearly demonstrates when some popular approaches are more appropriate than others. This is a major step forward for the property-based approach to the user’s dilemma.

One of the main advantages of the current approach is that it is theoretically founded. This offers a level of clarity on fundamental differences amongst clustering techniques that was not previously available, and as such stands to serve as a foundation for further fruitful exploration into core differences between clustering paradigms. Our approach also stands in stark contrast to less rigorous guidance that relies on considerations such as the shape of the data.

It is important to stress that while the weight-based classification is a major step forward in the user’s dilemma, it is not intended to offer a complete solution. In order to have a more comprehensive solution to the user’s dilemma we need additional properties that identify other essential differences amongst clustering techniques. The ultimate goal of the property-based approach to the user’s dilemma is to have a small set of complementary properties that together aid in the selection of clustering techniques for a wide range of applications.

Considerations outside of the property-based approach can be combined with the methodology proposed here to give the user further guidance on how to select a suitable clustering method for a given problem. Such considerations include, for example, issues of data presentation and identifying the number of clusters (see Jain [26] for a discussion). These considerations are compatible with the weight-based properties proposed here, and can be used in conjunction with them. In particular, representational issues come into consideration before selecting a clustering method, while the number of clusters can be found after identifying a suitable clustering techniques using the methodology proposed here.

The following summarizes the advantages of our approach:

- The properties we propose are theoretically founded. Clustering algorithms provably fall into one of three categories: Weight-robust, weight-considering, or weight-sensitive. This offers a level of rigor and accuracy that is often lacking from recommendations to the user’s dilemma.
- Real world data is often difficult to understand or visualize, particular when it is large and high dimensional. Users often need to utilize cluster analysis without insight into the shape of the data or threshold at which data should be marked as noise. Our approach can be applied without any special insight into the shape or structure of the data, but rather focuses on the needs of the application.
- Unlike prior property-based approaches to the user’s dilemma, the weight-based properties proposed here clearly elucidate differences between clustering methods without elevating any method over another. By contrast, prior properties in this line of research tend to highlight the advantages of single-linkage, while failing to elucidate the benefits of other popular techniques.
- Our approach is complementary to and can be used in conjunction with other important

considerations for identifying a useful clustering, such as representational considerations and identifying the right number of clusters [26].

### 7.3. Comparison to other clustering paradigms

Due to the high applicability of clustering, there are numerous variations of this data mining method. Our analysis focuses on two of the most popular clustering frameworks, a partitional setting where the number of clusters ( $k$ ) is provided, and the hierarchical clustering model. The weighted framework presented here is not limited to the frameworks on which we focus, and we discuss how it applies to several other paradigms.

#### 7.3.1. Density-based approaches

Density based clustering methods, such as, for instance the popular DBSCAN [22], focus specially on the identification of dense regions. A variety of density based approaches have been proposed (see a survey by Bhattacharjee and Mitra [14]). Density based clustering methods extract compact regions in the data space from noise, identifying clusters as areas of higher density than the remainder of the data [14].

There is an important distinction between the methodology proposed here and the density-based view of clustering. A critical aspect of the weighted properties framework is to choose a clustering technique based on how sparse regions should be treated. In stark contrast to eliminating noise and outliers, some applications, such as phylogeny, call for methods that prioritize the identification of clusters formed from a small number of elements.

On the other hand, when it is determined that for a given application it is desirable that noise and outliers not have excessive influence over the resultant clustering, then density based approaches can offer additional viable options. This is particularly relevant when users have sufficient insight into the nature of the data in order to effectively set parameters required by many density-based algorithms.

#### 7.3.2. Validity criteria and clusterability measures

In addition to clustering algorithms, other clustering tools are available throughout the cluster analysis process. The clustering pipeline [6] begins with clusterability evaluation [3], where notions of clusterability can determine whether data possesses inherent cluster structure. Then clustering algorithms are applied, followed by validity indices [8] (also known as “clustering quality measures” [4]), which assess whether a successful clustering has been discovered by any given clustering algorithm. Validity indices can further be internal (determining the quality of a clustering based on its structure) or external (comparing a clustering to a ground truth, when such is available). While cluster analysis is flexible and is often applied without all of these steps, clusterability analysis and validity indices offer additional tools to improve the chances of identifying useful partitions.

The techniques developed in this work apply not only to algorithms, but also to clusterability measures and validity indices, particularly internal validity indices. If, for instance, a user determines that their application calls for valuing data present in sparse regions (as opposed to treating it as noise or outliers), it would be important not only to select a clustering

algorithm that exhibits such behaviour, but also to ensure that any clusterability method or validity index that is applied in the analysis exhibits the corresponding treatment of weights.

If any component of the clustering pipeline is selected without regard for how weights should be treated for the given application, the analysis may yield misleading results. A clustering that is better suited for the application at hand may be discarded in favour of one with higher quality according to a validity index that treats weights differently. Similarly, a poorly chosen clusterability measure may deem data unclusterable when in fact it possesses cluster structure when accounting for the desired treatment of sparse regions (for example, a data set may only be clusterable if small clusters are permissible). While the current paper focuses on partitional and hierarchical clustering algorithms, categorization of clusterability and validity indices based on their weight sensitivity is left for future work. Future work may also involve the development of novel validity indices and notions of clusterability with varied behaviour with respect to how they treat weighted data.

#### 7.4. Conclusions

A primary limitation in the application of properties to the user’s dilemma is a theory-practice disconnect, whereby properties that formally distinguish between clustering techniques are often not relevant to the practitioner. A central contribution of this work is the introduction of properties that simultaneously allow for formal differentiation between clustering techniques, while being helpful in practice.

We studied the behaviour of clustering techniques in a weighted framework, presenting three fundamental categories that describe how algorithms respond to weights and classifying several well-known techniques according to these categories. Our results are summarized in Table 1. We note that all of our results readily translate to the standard setting, by mapping each point with integer weight to the same number of unweighted duplicates. We discussed how the weighted clustering categories can be applied to help identify a suitable clustering algorithm in practice, with illustrations from specific clustering applications, and discussed how the weighted clustering approach developed here relates to other clustering constructs.

This paper presents a significant step forward in the property-based approach for selecting clustering algorithms. Unlike previous properties, which focused on advantages of linkage-based algorithms, these properties show when applications call for popular center-based approaches, such as  $k$ -means. Furthermore, the simplicity of these properties makes them widely applicable, requiring only that the user decide whether duplicating elements should impact the output of the algorithm. We hope that this work will trigger additional research on formal properties that are applicable to solving the user’s dilemma in the practical arena and lead to the study of the weighted behaviour of other clustering constructs.

## References

- [1] M. Ackerman and S. Ben-David. Discerning linkage-based algorithms among hierarchical clustering methods. In *IJCAI*, 2011.
- [2] M. Ackerman, S. Ben-David, and D. Loker. Characterization of linkage-based clustering. In *COLT*, 2010.

- [3] M. Ackerman and S. Ben-David. Clusterability: A theoretical study. *Artificial intelligence and statistics*. 2009.
- [4] M. Ackerman and S. Ben-David. Measures of clustering quality: A working set of axioms for clustering. *Advances in neural information processing systems* 21 (2008): 121-128.
- [5] M. Ackerman, S. Ben-David, and D. Loker. Towards property-based classification of clustering paradigms In *NIPS*, 2010.
- [6] Andreas Adolfsson, M. Ackerman, and Naomi C. Brownstein. To cluster, or not to cluster: An analysis of clusterability methods. *Pattern Recognition* 88 (2019): 13-26
- [7] Apresjan, Ju D. An algorithm for constructing clusters from a distance matrix, Mashinnyi perevod: prikladnaja lingvistika: 9 (1966): 3-18
- [8] Arbelaitz, Olatz, et al. An extensive comparative study of cluster validity indices. *Pattern Recognition* 46.1 (2013): 243-256.
- [9] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *SODA*, 2007.
- [10] Balcan, Maria-Florina, Nika Haghtalab, and Colin White.  $k$ -center Clustering under Perturbation Resilience. *ACM Transactions on Algorithms (TALG)* 16.2 (2020): 1-39.
- [11] M. F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *STOC*, 2008.
- [12] Ben-David, Shai. Clustering – What both theoreticians and practitioners are doing wrong. *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1. 2018.
- [13] Bandelt, H-J., and Andreas WM Dress. Weak hierarchies associated with similarity measures—an additive clustering technique. *Bulletin of mathematical biology* 51.1 (1989): 133-166.
- [14] Bhattacharjee, P., and P. Mitra. A survey of density based clustering algorithms. *Frontiers of Computer Science* 15.1 (2020): 1-27.
- [15] Bryant, D., and V. Berry. A structured family of clustering and tree construction methods. *Advances in Applied Mathematics* 27.4 (2001): 705-732.
- [16] Cohen-Addad, Vincent, Varun Kanade, and Frederik Mallmann-Trenn. Clustering Redemption – Beyond the Impossibility of Kleinberg’s Axioms. *Advances in Neural Information Processing Systems*. 2018.
- [17] R. Bosagh-Zadeh and S. Ben-David. A uniqueness theorem for clustering. In *UAI*, 2009.

- [18] Sébastien Bubeck, Marina Meila, and Ulrike von Luxburg. How the initialization affects the stability of the k-means algorithm. *arXiv preprint arXiv:0907.5494*, 2009.
- [19] Gunnar Carlsson and Facundo Mémoli. Characterization, stability and convergence of hierarchical clustering methods. *The Journal of Machine Learning Research*, 11:1425–1470, 2010.
- [20] Cohen-Addad, Vincent, Varun Kanade, and Frederik Mallmann-Trenn. Clustering Redemption-Beyond the Impossibility of Kleinberg’s Axioms. *NeurIPS*, 2018.
- [21] Sara Dolnicar. A Review of Unquestioned Standards in Using Cluster Analysis for Data-driven Market Segmentation. *Conference Proceedings of the Australian and New Zealand Marketing Academy Conference 2002 (ANZMAC)*, Deakin University, Melbourne, 2-4. December 2002.
- [22] Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, vol. 96, no. 34, pp. 226-231. 1996.
- [23] L. Fisher and J. Van Ness. Admissible clustering procedures. *Biometrika*, 58:91–104, 1971.
- [24] Dorit S Hochbaum and David B Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985.
- [25] Lawrence Hubert and James Schultz. Hierarchical clustering and the concept of space distortion. *British Journal of Mathematical and Statistical Psychology*, 28(2):121–133, 1975.
- [26] Jain, Anil K. Data clustering: 50 years beyond K-means. *Pattern recognition letters*, 31.8 (2010): 651-666.
- [27] Nicholas Jardine and Robin Sibson. The construction of hierarchic and non-hierarchic classifications. *The Computer Journal*, 11(2):177–184, 1968.
- [28] Jiang, Heinrich, Jennifer Jang, and Ofir Nachum. Robustness guarantees for density clustering. *The 22nd International Conference on Artificial Intelligence and Statistics. PMLR*, 2019.
- [29] L. Kaufman and P. J. Rousseeuw. *Partitioning Around Medoids (Program PAM)*, pages 68–125. John Wiley & Sons, Inc., 2008.
- [30] J. Kleinberg. An impossibility theorem for clustering. *Proceedings of International Conferences on Advances in Neural Information Processing Systems*, pages 463–470, 2003.

- [31] Kłopotek, Robert A., and Mieczysław A. Kłopotek. On Probabilistic k-Richness of the k-Means Algorithms. *International Conference on Machine Learning, Optimization, and Data Science*. Springer, Cham, 2019.
- [32] Kłopotek, Mieczysław Alojzy and Robert Albert Kłopotek. Clustering algorithm consistency in fixed dimensional spaces. *International Symposium on Methodologies for Intelligent Systems*. Springer, Cham, 2020.
- [33] Kłopotek, Mieczysław, and Robert Kłopotek. In-The-Limit Clustering Axioms. *International Conference on Artificial Intelligence and Soft Computing*. Springer, Cham, 2020.
- [34] Meilă, Marina. Good (K-means) clusterings are unique (up to small perturbations). *Journal of Multivariate Analysis* 173 (2019): 1-17.
- [35] Arnd Kohrs-Bernard Merialdo. Clustering for collaborative filtering applications. *Intelligent Image Processing, Data Analysis & Information Retrieval* 3 (1999): 199.
- [36] Moore, Jarrod, and Margareta Ackerman. Foundations of perturbation robust clustering. *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016.
- [37] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *FOCS*, 2006.
- [38] Peng, Pan, and Yuichi Yoshida. Average sensitivity of spectral clustering. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*. 2020.
- [39] Yun Trinh, Scott Lindeneau, Margareta Ackerman, Simone Baumann-Pickering, and Marie Roch. Unsupervised clustering of toothed whale species from echolocation clicks. In *The Journal of the Acoustical Society of America* 140, 3302 (2016).
- [40] Sartaj Sahni and Teofilo Gonzalez. P-complete approximation problems. *Journal of the ACM (JACM)*, 23(3):555–565, 1976.
- [41] Peter HA Sneath and R Robert Sokal Numerical taxonomy. The principles and practice of numerical classification. 1973.
- [42] D. Steinley. K-means clustering: a half-century synthesis. In *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34, 2006.
- [43] Douglas Steinley and Michael J Brusco. Initializing k-means batch clustering: a critical evaluation of several techniques. In *Journal of Classification*, 24(1):99–121, 2007.
- [44] Lyle H. Ungar and Dean P. Foster. Clustering methods for collaborative filtering. In *AAAI workshop on recommendation systems*. Vol. 1. 1998.

- [45] U. Von Luxburg. A tutorial on spectral clustering. In *J. Stat. Comput.*, 17(4):395–416, 2007.
- [46] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. In *Journal of the American statistical association*, 58(301):236–244, 1963.
- [47] Wei, Jia-heng. Two examples to show how k-means reaches richness and consistency. DEStech Transactions on Computer Science and Engineering (2017).
- [48] W. E. Wright. A formalization of cluster analysis. In *J. Pattern Recognition*, 5(3):273–282, 1973.