Indian Statistical Institute, Kolkata

Semi-Supervised Clustering of stable instances

by

Deepayan Sanyal(CS1607)

A report submitted in partial fulfillment for the
degree of Master of Technology in Computer Science

Guided by
Prof. Swagatam Das
Electronics and Communication Sciences Unit

July 2018

Declaration of Authorship

I, Deepayan Sanyal, declare that this thesis titled, '*Semi-Supervised Clustering of stable instances*' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the Indian Statistical Institute.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

CERTIFICATE OF COMPLETION

This is to certify that the dissertation entitled **“Semi-Supervised Clustering of stable instances”** submitted by **Deepayan Sanyal** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Swagatam Das
Associate Professor,
Electronics and Communication Sciences Unit,
Indian Statistical Institute,
Kolkata-700108, INDIA.

"Millions saw the apple fall but Newton was the one who asked why."

-Bernard Baruch

Abstract

Clustering is one of the fundamental problems in data mining. Various objective functions, like k-means, k-median, k-mediod have been applied to solve clustering problems. While all these objectives are *NP-Hard* in the worst case, practitioners have found remarkable success in applying heuristics like Lloyd's algorithm for this problem. The following question then becomes important: *what properties of real-world instances will enable us to design efficient algorithms and prove guarantees for finding the optimal clustering*? We consider the case of multiplicative perturbation stability. Stable instances have an optimal solution that does not change when the distances are perturbed. This captures the notion that optimal solution is tolerant to measurement errors and uncertainty in the points. Semi-supervision allows us to have an oracle $\mathcal{O}$ which answers pairwise queries. We design efficient algorithms which solve problems of multiplicative perturbation stability using a noisy oracle model.

Acknowledgements

I would like to show the highest gratitude to my advisor, Dr. Swagatam Das, Associate Professor, Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, for his guidance and continuous support and encouragement. He has taught me how to do good research, and motivated me with great insights and innovative ideas.

I would also extend my deepest respect towards Prof. CA Murthy for reviewing my mid-term presentation and providing valuable inputs for the future work.

My deepest thanks to all the teachers of Indian Statistical Institute, for their valuable suggestions and discussions which added an important dimension to my research work.

Finally, I am very much thankful to my parents and family for their everlasting supports.

Last but not the least, I would like to thank all of my friends for their help and support. I thank all those, whom I have missed out from the above list.

Contents

Declaration of Authorship i

Abstract iv

Acknowledgements v

1 Introduction 1

2 Preliminaries 6
- 2.1 α-Multiplicative Perturbation Stability . . . 6
- 2.2 Semi-supervised Clustering . . . 7
- 2.3 Overview of our methods . . . 8

3 Related Work and Our Contribution 9
- 3.1 Related Work . . . 9
- 3.2 Our Contribution . . . 11

4 Our Algorithms 12
- 4.1 Structural Properties . . . 12
- 4.2 Algorithm for Ideal Oracle . . . 13
- 4.3 Algorithm for Noisy Oracle . . . 15

5 Conclusion 22

Bibliography 23

Dedicated to my parents and family...

Chapter 1

Introduction

Clustering is at once a very basic and useful task in data mining. The main aim of clustering is to partition the dataset into meaningful partitions. An operational definition of clustering can be defined as: Given n objects, find k groups based on a measure of similarity such that objects in the same group are more similar than objects in different groups. Problems of clustering data arise in a variety of areas like document clustering, protein clustering, image segmentation, cosegmentation among a few. However, despite its vitality in various tasks, there is severe dearth of theoretical foundations of clustering. Without doubt, it is difficult to develop a universal theoretical basis for clustering at a level of generality so as to make it relevant across multiple applications and uses[1]. However, in recent times, a lot of new work has been published, which aim to provide a rigorous mathematical basis for clustering and to further the theory behind this vital and provably difficult task.

The k-means is possibly the most widely used clustering technique. Given a dataset and k, the number of clusters, k-means seeks to minimize the average squared distance for points in the same cluster. It achieves this through a greedy approach. First, a number of random points corresponding to the centers of the k clusters is generated and each point in the dataset is assigned to the nearest center. Next, from the assignment of the data points to different clusters, the cluster centers are recomputed through averaging the points in a particular cluster. This process is repeated iteratively until there is no change in cluster assignment of the datapoints. This solution, proposed by Lloyd[2] is still very widely used and has often been regarded as the most popularly used clustering technique in data mining. However, the greedy approach taken by k-means can only converge to a local minima. So, to get better results, it is customary to run k-means several times with different initialization of centers and then choose the one with the minimum objective value. Various modifications to k-means have been proposed, including k-means++[3] by Arthur et al. which improves upon k-means by choosing the centers in a more intelligent manner. Other than that, Pelleg et al.[4] used a kd-tree to speed up the computation of the closest cluster center for each datapoint. Bradley et al.[5] presented a fast, scalable and single-pass version of k-means, which does not require the entire data to be fit into the memory at the same time. It is quite customary to use k-means with Euclidean distances to compute the distances between points and cluster centers. As a result, k-means usually finds spherical or ball-shaped clusters in the data. *Kernel k-means*[6] was proposed to detect arbitrarily shaped clusters, with an appropriate choice of the kernel similarity function. Another modification to the k-means worth mentioning is the fuzzy c-means[7]. While k-means assigns each point to one cluster, the fuzzy c-means allows the points to belong to different clusters at the same time with a certain membership value.

However, in most clustering algorithms, the number of clusters, k is taken as an input from the user. Determining the number of clusters automatically presents a hard algorithmic problem which does not have a '*one size fits all*' solution. ISODATA[8] seeks to find the number of clusters by iteratively merging and splitting clusters. The algorithm uses a user-defined threshold and splits clusters when the intra-cluster standard deviation is greater than the threshold and merges clusters when the distance between their centers is less than the threshold. Pelleg et al[9] proposed X-means, which automatically estimates the number of clusters, while it also improves scalability and avoids local minima problems of the k-means. They determined the number of clusters by using the Bayesian Information Criteria(BIC). Subsequently, Hamerly et al.[10] proposed G-means based on the Anderson-Darling statistic[11]. This method works on the hypothesis that a subset of the data follows a Gaussian distribution. They argued that the BIC does not penalize strongly enough for model complexity and hence does not work well as a scoring function. Bischoff et al. use a minimum description length(MDL) framework. Their algorithm starts with a very high estimate of the number of centers, k and reduces k when removing a center reduces the description length. In between the steps, they use k-means to optimize the data fit to the model. Other methods of calculating the number of clusters includes the *elbow*[12] method which increases the number of clusters as long as they explain a significant portion of the data variance. At some point, the drop in variance given by adding a cluster drops and the number of clusters is chosen at this point. However, such points can often not be determined unambiguously. There is also the *silhouette*[12] method, where the silhouette of a data instance is a measure of how closely it is matched to data within its cluster and how loosely it is matched to data of a neighbouring cluster. The neighbouring cluster is the cluster which is closest to the data point. A silhouette close to +1 denotes the data is in the correct cluster and a value close to -1 denotes that the data is in the wrong cluster. Genetic algorithm based optimization techniques are useful to determine the maximum silhouette value.

Clustering in itself is a difficult task. However, in many cases, domain knowledge is available to the experimenter which could make the clustering task more feasible. To integrate this background information, Wagstaff et al.[13] focused on two types of constraints, namely *Must-Link* constraints and *Cannot-Link* constraints. *Must-Link* constraints specify that two data instances belong in the same cluster while *Cannot-Link* constraints specify that two data instances belong to different clusters. They only consider hard constraints, i.e. the constraints cannot be violated. Wagstaff et al.[13] combined this with k-means to create the COP-KMEANS algorithm which, like the k-means algorithm, assigns datapoints to the nearest cluster center. The difference is that COP-KMEANS ensures that no constraint is violated during cluster assignment. They reported significant improvement in clustering accuracy with introduction of only small number of constraints. However, the major limitation of this algorithm is that the greedy approach taken by this algorithm can often fail to find a cluster assignment which satisfies all constraints. To counter this problem Basu et al.[14] proposed PCKMeans which is a *soft constraint* version of COP-KMEANS. PCKMeans modifies the objective function of k-means and adds penalties whenever constraints are violated. A very high penalty would make it work like COP-KMEANS while a low penalty would make it similar to the normal k-means. An *active* version of PCKMeans is also available where the algorithm searches through the data to get k pairwise disjoint non-null neighbourhoods. It is assumed that the neighbourhoods would present a skeleton of the underlying cluster structure. After the initial cluster structure is ready, new queries are used to consolidate the skeleton and then initial cluster centers are estimated using means of the k non-null partitions. Basu et al.[14] showed that the *active* PCKMeans outperforms the generic PCKMeans for many datasets. The model that PCKMeans assumes is that there exists an oracle which answers pairwise queries and hence the algorithm can modify the queries it makes based on the the results of the queries it has made before. This introduces an *active* paradigm of semi-supervised clustering, which has a significant advantage over methods where arbitrary constraints are available to the algorithm. This concept of an oracle which answers pairwise queries made by the algorithm will be used by us. The oracle model which will be used by us is the *noisy* oracle model which answers pairwise queries correctly with probability $p > \frac{1}{2}$. It is important to understand that when the queries are erronous with probabilty p, repeating the same query multiple times does not change the answer. This model follows efforts made recently to use human in the loop. In this setting, humans are asked simple pairwise queries adaptively, ‘do

u and v represent the same entity?', and these queries are used to improve the algorithm performance. Proliferation of crowdsourcing platforms like Amazon Mechanical Turk and CrowdFlower allows for easy implementation of such models. However, data collected from non-expert workers on crowdsourcing networks are essentially noisy and query results can be influenced by previous experience or subject assumptions regarding the domain. A simple scheme to remedy this is to take a majority vote after presenting the same query multiple times to multiple subjects. But even in that case, some error in query output still remains. Let $p, \frac{1}{2} < p < 1$ be the probability with which the aggregated answer is correct. This forms the basis of the *noisy* oracle model we will be using in our algorithm. Clustering with the *noisy* oracle is basically reducing the number of queries made to the crowdsourcing platform to reduce cost and time while also simultaneously recovering the underlying cluster structure from the queries made.

The major challenge in the theory of clustering is to bridge the large disconnect between our theoretical and practical understanding of clustering. Most common clustering objectives like the k-means or k-median are NP-hard in the worst case even for $k = 2$. However, variants of these algorithms like the k-means++ work as good heuristics to solve many clustering problems. This has led to the following hypothesis: Clustering is difficult only when it does not matter, known commonly as the CDNM hypothesis[15, 16]. The natural question that arises is the following: *Why are real-world instanes of clustering easy? Can we identify properties of real-world instances that make them tractable?[17]* One way to model real-world instances of clustering problems is through *instance stability*, which is an implicit structural assumption about the instance. Practical instances of clustering usually have a clear optimal solution and this solution remains the same even after slight perturbation of the data instances, i.e. the instances are stable. Balcan et al.[18] argues that clustering objectives are often proxies for recovering an unknown *ground truth* clustering. Additionally, the distance measures we use are almost always heuristics used to find an optimal clustering close to the *ground truth*. Such distance measures do not necessarily constitute the correct 'semantic distance' in the data instance. Thus, finding the *correct* clustering using distances and objective functions only make sense when the data instances are resilient to perturbations introduced by these proxies and heuristics. The notion of stability introduced first and formalized was the multiplicative perturbation stability[19]. This notion assumed that the optimal solution is resilient to multiplicative perturbations of the distances. For $\alpha \geq 1$, a metric clustering instance (X, d) on point set $X \subset \mathbb{R}^d$ and metric $d : X \times X \to \mathbb{R}^+$ is said to be α-multiplication perturbation resilient iff the unique optimal clustering $X_1, X_2, \ldots, X_k$ of X remains the optimal solution for any instance (X, d') where any subset of the distances are increased by upto a factor of α factor, i.e. $d(x, y) \leq d'(x, y) \leq \alpha d(x, y)$ for any $x, y \in X$. However, multiplicative perturbation stability has strong structural conditions which are unlikely to be followed by real-world datasets. Also, presence of outliers in the dataset would further affect datasets adhering to this notion of stability. To counter this, another notion of stability is the additive perturbation stability. This requires that for a given δ, the optimal k-means clustering remains optimal even if the points are perturbed by an Euclidean distance of at most δ. δ that is taken for additive perturbation stability is not scale invariant. Ackerman et al.[20] took $\delta = \epsilon diam(X)$, where $diam(X)$ is the diameter of the whole dataset. However, diameter is a very non-robust quantity and presence of a few outliers can affect it considerably. As an alternative, Dutta et al.[17] came up with a notion of $\delta = \epsilon \text{max}_{i,j}|\mu_i - \mu_j|$, where μ_i, μ_j are any two cluster centers and provided polynomial time algorithms to recover the optimal clusterings.

Chapter 2

Preliminaries

In this section, we introduce some definitions and notions relevant to this thesis.

A *k-clustering* of data set X is a *k-partition* of X, that is a set of k non-empty, disjoint subsets of X. A *clustering* of X is a k-clustering of X for some $k \geq 1$. For $x, y \in X$ and clustering C of X, $x \sim_C y$ whenever x and y are in the same cluster with respect to C, and $x \nsim_C y$, otherwise.

2.1 α-Multiplicative Perturbation Stability

Definition: Given a metric space(X, d), a clustering objective is *center-based* if the optimal solution can be defined by k points $c_1, c_2, \ldots, c_k$ in the metric space X called *centers* such that every data point is assigned to its nearest center. Such a clustering objective is *separable* if it furthermore satisfies the following two conditions:

- The objective function value of a given clustering is either a (weighted) sum or the maximum of the indivuidual cluster values.
- Given a proposed single cluster, its score can be computed in polynomial time.

The set of such points $c_1, c_2, \ldots, c_k$ is called a set of centers for the clustering C. The clustering partition can be found by the Voronoi partition induced by such centers. The problem is to find such centers. The basic notion of stable clustering is that the *ground-truth* clustering induced by these points remains the same even after certain perturbations given to the data points. We now go on to describe our notion of perturbations.

Definition: Given a metric (X,d) and $\alpha > 1$, we say a function $d' : X \times X \to \mathcal{R}_{\geq 0}$ is an α-perturbation of d, if for any $x, y \in X$, it holds that

$$d(x,y) \leq d'(x,y) \leq \alpha d(x,y)$$

It is to be noted that d' need not be a metric. Any non-negative function would suffice.

Definition: Suppose we have a clustering instance composed of n points residing in a metric (X, d) and an objective function ϕ we wish to optimize. We call the clustering instance *α-perturbation resilient* for ϕ if for any d' which is an α-perturbation of d, the (only) optimal clustering of (S, d') under ϕ is identical, as a partition of points into subsets, to the optimal clustering of (S, d) under ϕ.

We will work with only separable center-based objectives (s.c.b.o) ϕ. Many common clustering objectives such as k-means, k-median and k-center are s.c.b.o.

The method we follow to arrive at our results is to first derive structural properties necessary to maintain the α-perturbation resilience property. In the second step, we use semi-supervised methods to find the optimal clustering in the α-perturbation resilient scenario.

2.2 Semi-supervised Clustering

Conventional clustering methods, which seek to identify homogenous subgroups in data are unsupervised in nature, that is, nothing is known about the *output* variable and no interrelationships among the observations are known. Unsupervised clustering is a difficult task, which has led people to turn to obtaining some form of *background knowledge* to make it more feasible.

The kind of *background information* that we will deal with is in the form of two sets of instance level constraints:

- **Must-Link Constraints** which specify that two instances have to be in the same cluster.
- **Cannot-Link Constraints** which specify that two instances have to be in different clusters.

The formal model that we will be using in this thesis for semi-supervised clustering is the *noisy oracle* model. Suppose we are given n points to cluster. We will assume that there exists an oracle, which when given two points u and v, tells us whether they belong in the same cluster or not. For the *noisy oracle*, we will assume that the answers given by the oracle are correct with probabilty better than random, i.e. with probability $p = \frac{1}{2} + \lambda$ where $\lambda > 0$. It must be noted that if we repeat the queries to the oracle, we will always get the same answer. Formally, clustering with the noisy oracle model is defined as[29]:

Consider a set of points $X \equiv [n]$ containing k *latent* clusters X_i, $i = 1, ..., k$ such that for $i \neq j$, $X_i \cap X_j = \phi$ where k and X_i's are unknown. There is an oracle $\mathcal{O} : X \times X \to \{\pm 1\}$ with two error parameters p, q: $0 < p < q < 1$. The oracle takes as input two points u, $v \in X \times X$ and if u, v belong to the same cluster, $\mathcal{O}_{p,q}(u, v) = +1$ with probability $1 - p$ and $\mathcal{O}_{p,q}(u, v) = -1$ with probability p. On the other hand, if u, v do not belong to the same cluster, $\mathcal{O}_{p,q}(u, v) = +1$ with probability $1 - q$ and $\mathcal{O}_{p,q}(u, v) = -1$ with probability q. Note that the oracle returns the *same* answer on *repitition.* Now, given X, find $Y \subseteq X \times X$ such that $|Y|$ is minimum, and from the oracle answers, it is possible to find k and the X_i's with high probability. Note that the queries Y made to oracle $\mathcal{O}$ can be made adaptively based on the outputs of previous queries.

If $p = 1 - q = 0$, we will say that the oracle $\mathcal{O}$ is ideal. Thus, an ideal oracle is one which always answers the queries correctly with probability 1. Clustering with both noisy and noiseless queries form a part of current research trends.

2.3 Overview of our methods

For α-multiplicative perturbation resilient instances, we will derive several structural properties which these notions of stability impose on the data instances. Then we will exploit these structural properties using noisy and ideal oracle to output the optimal clusters. One fact to note is that our method using semi-supervision allows us to find the optimal clusters even without knowing the correct number of clusters. Choosing the number of clusters for an instance is a difficult task in clustering and is often left as choices made by domain experts or assumed to be an input along with the data instance. This is, to our knowledge, the first attempt to use stability notions and semi-supervision to provide guarantees for clustering instances. It must be noted that the presence of noisy queries ensures that that we can never be absolutely sure about our output. We can only say that our output is correct with very high probability.

Chapter 3

Related Work and Our Contribution

3.1 Related Work

Kleinberg [21] defined a set of three properties, *Richness, Scale-Invariance and Consistency* and showed there is no clustering function which satisfies all three properties simultaneously. Following up, Ben-David et al.[22] defined a *clustering-quality* measure as a function that maps the pair {*dataset, clustering*} to a certain set(say, the set of non-negative real numbers) such that these values reflect how 'good' or 'cogent' the clustering output is. They provided a set of axioms and proposed several clustering quality measures which satisfy those axioms. These works represent an attempt to prove or disprove the existence of an unified framework for a problem as difficult as clustering.

Balcan et al.[18] argue that in most uses of clustering, we assume that there exists some 'correct' clustering and we hope that optimising the objective functions will help us get closer to the *truth*. They showed that if there exists any c-approximation that is ϵ-close to the target, then it is possible to produce clusterings which are $O(\epsilon)$ close to the target, even for values of c for which obtaining the c-approximation is *NP-hard*. Epster et al.[23] showed a means to measure clustering tendency or *clusterability* of a dataset. They defined a perfectly *clusterable* dataset as one in which all intracluster edges are smaller than any intercluster edge. Ostrovsky et al.[24] introduced separability as a notion of clusterability. They defined separability to measure the drop in the loss function when moving from $(k-1)$ clusters to k clusters for the k-means objective(though it can be defined for other objectives as well). They said that a dataset X is (k, ϵ)-separable if the k-means objective is ϵ times smaller for k than for $(k-1)$. Zhang et al.[25] introduced the ratio of *between-cluster* variance and *within-cluster* variance as a measure of *clustered-ness* of the data. Ackerman et al.[1] defined *clusterability* as a measure of clustered structure in a dataset. They intitiated a theoretical study of the various notions of *clusterability* and introduced a new notion of *clusterability*. For these notions, they showed that it is possible to obtain the optimal cluster for well clusterable data feasibly. They further showed that it is often *NP-hard* to calculate the clusterablility values of datasets.

Most clustering objectives like k-means, k-median, min-sum are *NP-hard* to optimise. Bilu et al.[15] reasoned that the optimal clustering to a desired objective must be resilient to multiplicative perturbations up to a factor $\alpha > 1$ of the distances between points. They show that for instances of max-cut clustering which are stable to $\alpha = O(\sqrt{n})$ perturbations, the optimal clusters can be computed using efficient algorithms. Awasthi et al.[19] considered the case of α-multiplicative perturbation resilient instances of center-based clustering objectives and showed polynomial time algorithms for $\alpha \geq 3$. Balcan and Liang[26] improved upon this result to provided algorithms for $alpha \geq 1 + \sqrt{2}$. Ben-David and Reyzin[20] show that the problems of finding the optimal clusters in $(2-\epsilon)$-multiplicative perturbation stable instances of k-median is *NP-Hard*. They also proved *NP-Hardness* of $(2-\epsilon)$-stable instances of min-sum clustering. Balcan, Haghtalab and White[17] provided algorithms for solving 2-multiplicative perturbation stable version of both symmetric and assymetric k-center. They also prove that finding the optimal cluster for the k-center problem is *NP-Hard* for $\alpha = 2 - \epsilon$.

Ackerman et al.[20] first initiated the study of additive perturbation stability and they used the diameter of the dataset $diam(X)$ as the normalization element. Following this, Dutta et al.[17] used the maximum distance between a pair of means as the normalizing factor and provided polynomial time algorithms to find out the optimal clusters when the dataset X and the number of clusters k are known.

Ashtiani et al.[27] assumed that the optimal clusters can be approximated by first mapping the dataset into a new space and then performing k-means clustering. The user provided the clustering for a random small sample of the data and their algorithm found out an approximation for the optimal clustering by learning the representation of the datapoints. Ashtiani et al.[28] showed that under some margin assumptions, instances of k-means which are generally *NP-Hard* to compute can be computed easily with a small number of same-cluster queries. Recently, Mazumdar et al.[29] provided lower bounds for the number of queries which must be made to a noisy oracle in order to obtain the clustering correctly. The also provided polynomial-time algorithms which can compute the correct clustering with high probability in both the adaptive and non-adaptive scenario.

3.2 Our Contribution

In this thesis, we have studied the α-multiplicative perturbation stability and structural properties that these instances must satisfy. We have then taken advantage of these properties using semi-supervision to obtain the optimal clustering. The advantage of our method is that we do not need to have the number of clusters k as an input. Our algorithms do the following:

- For the ideal oracle, we present a polynomial time algorithm which obtains the optimal clusters for $\alpha \geq 2$ with $O(n)$ queries.
- For the noisy oracle, we present a polynomial time algorithm which obtains the optimal clusters for $\alpha \geq 3$ with $O(n \log^2 n)$ queries and with probability at least $1 - O(\frac{n}{n^{\log n}})$

There are scopes of improving the bound for α for the noisy oracle scenario, but for this dissertation, we will stick to $\alpha \geq 3$.

Chapter 4

Our Algorithms

4.1 Structural Properties

In this section we will provide proofs for some properties which are relevant to our algorithms.

Definition 4.1.1: Let $p \in X$ be any arbitrary point, let c_i be the center that p is assigned to in the *ground-truth* clustering and let $c_j \neq c_i$ be any other center in the *ground-truth* clustering. We say X satisfies *α-center proximity* property if for any $p \in X$, it holds that

$d(p, c_j) > \alpha d(p, c_i)$

This property implies that in the *ground-truth* clustering of instances with *α-center proximity* property, every point p is at least α times closer to the center of the cluster it is assigned to than any other cluster center.

Fact 4.1.2:[19] If a clustering instance satisfies the *α-perturbation resilient* property, then it also saisfies the *α-center proximity* property.

Proof: The entire proof appears in the original paper. We will discuss the basic rational here. We take any cluster C_i with center c_i and blow up all distances in it by a factor of α. Now consider any other cluster C_j with center c_j. Since even after the perturbation, any point $p \in C_i$ is still assigned to the same cluster, the following inequality results:
$d(p, c_j) > \alpha d(p, c_i)$

We now go on to prove the property which is of importance to our algorithm.

Theorem 4.1.3:[19] Let X be a given *α-perturbation resilient* clustering instance. For every point $p \in C_i$ with center c_i and any other point p' belonging to some other cluster $C_j(j \neq i)$ with center c_j, it follows that $d(p,p') > (\alpha-1)d(p,c_i)$.

Proof: By the triangle inequality, $d(p,p') \geq d(p,c_j) - d(p',c_j)$. Without loss of generality, assume that $d(p,c_i) > d(p',c_j)$. Since the data instance is stable to α-multiplicative perturbations, $d(p,c_j) > \alpha d(p,c_i)$. Combined with the triangle inequality, this gives $d(p,p') > \alpha d(p,c_i) - d(p',c_j)$. Taking our assumption into account, $d(p,p') > \alpha d(p,c_i) - d(p,c_i) = (\alpha-1)d(p,c_i)$. This proves the theorem.

Corollary 4.1.4: For $\alpha > 2, d(p,p') > d(p,c_i)$.
This corollary is trivial for $\alpha > 2$ in the above theorem.

4.2 Algorithm for Ideal Oracle

We assume that the data instance has the weak center proximity property. Our algorithm takes the input of set of points X and outputs a clustering of all the points.

Algorithm 1 Clustering with a perfect oracle

```
    input: the set of input points $X$
    output: the clusters obtained $X_1, X_2, \dots, X_k$ after determining the correct number
    of clusters $k$
    procedure BuildTree($X$)
4:      Create a complete graph($X', E$), with $X$ as the vertices and the distance between
    points in $X$ as the edge weights.
        Build the Minimum Spanning Tree $T$ on $X'$ using Prim's algorithm
    end procedure
    procedure BuildClustersIdeal($T$)
8:      for every edge $e = (u, v) \in T$ do Query the ideal oracle whether $u$ and $v$ belong
    to the same cluster
            if $u$ and $v$ belong to different clusters then
                remove edge $e = (u, v)$ from $T$
            end if
12:     end for
    end procedure
    return $T = \{T_1, T_2, \dots, T_k\}$        ▷ $T_1, T_2, \dots, T_k$ form the optimal clusters.
```

Definition 4.2.1(Weak center proximity): A data instance X with optimal clusters $X_1, X_2, \ldots, X_k$ and cluster centers $\mu_1, \mu_2, \ldots, \mu_k$ is said to have the **weak center proximity** property if for all $i, 1 \leq i \leq 1, x \in X_i, y \in X_j$ and $j \neq i$, $d(x, \mu_i) < d(x, y)$.

Based on Corollary 4.1.4, the following fact follows trivially.
Fact 4.2.2: For $\alpha \geq 2$, data instances with α-multiplicative perturbation stabilty possess the **weak center proximity** property.

We will now prove the correctness of Algorithms 1 and 2. For algorithm 1, we show that for data instances with weak center proximity, the BUILDTREE procedure produces a tree with $(k-1)$ inter-cluster edges and $(n-k)$ intra-cluster edges. We show the proof with respect to the working of Prim's algorithm.
Prim's algorithm is a greedy technique to compute the minimum spanning tree(MST) in a given graph. It starts with an empty spanning tree. The idea is to maintain two sets of vertices, one of those already included in the MST and the others not included. At all steps, Prim's algorithm looks at all the edges which connect the two sets and chooses the edge with the minimum weight among these edges.

Lemma 4.2.3: For data instance X with the weak center property and optimal clusters $X_1, X_2, \ldots, X_k$, let $x \in X_i$ be the first point from X_i that is added to the MST. Let μ_i be the center of X_i. Then, no edge of the form $(u, v), u \in X_i$ and $v \in X_j, j \neq i$ is added to the MST before an edge of the form $(u, \mu_i), u \in X_i$ is added.

Proof. Suppose the opposite and we arrive at a contradiction. The fact that $(u, v), u \in X_i, v \in X_j, j \neq i$ was added implies that u is already in the MST. However, $d(u, v) > d(u, \mu_i)$, which is a contradiction. □

Lemma 4.2.4: Let us consider a point $x \in C_i$ which had not been added when μ_i was added. Then, x is added to the MST through the addition of an edge either of the form $(y, x), y \in X_i$ or (μ_i, x).

Proof. Suppose x was added through the addition of an edge of the form $(y, x), y \in X_j, j \neq i$. However $d(\mu_i, x) < d(y, x)$ form the weak center proximity property. So, Prim's algorithm should add (μ_i, x) to the MST before (y, x). This is a contradiction. □

Theorem 4.2.5: For data instances X with the weak center property, procedure **BUILDTREE** of Algorithms 1 and 2 produces a Minimum Spanning tree in which there are $(k-1)$ inter-cluster edges and $(n-k)$ intracluster edges.

Proof. Let us consider the above two lemmas and their implications. We see that whenever any point in any cluster is added to the MST, except for the first point, all other points are added through the addition of an intra-cluster edge. Thus, inter-cluster edges are added only when the first point from any cluster is added to the MST. Since there are k clusters, only $(k-1)$ inter-cluster edges were added. Thus, the MST consists of $(n-k)$ intra-cluster edges. □

Theorem 4.2.6: For an instance X with weak center proximity property, procedure **BUILDCLUSTERSIDEAL** of Algorithm 1 produces the correct optimal clustering assuming we have an ideal oracle $\mathcal{O}$(as defined in the Chapter 2) to answer pairwise queries.

Proof. The previous theorem shows that the MST T formed at the end of the BUILDTREE procedure has $(k-1)$ inter-cluster edges. Identifying and removing those edges from T would do the dual task of identifying the number of clusters and finding the optimal clusters as well. Since we have an ideal oracle $\mathcal{O}$ to use, we query the oracle for every edge in the tree T and remove the edges for which $\mathcal{O}(u,v) = +1$. Thus, T now consists of k sub-trees which represent the individual optimal clusters. □

4.3 Algorithm for Noisy Oracle

Definition 4.3.1(Min-stability): A data instance X with optimal clusters $X_1, X_2, \ldots, X_k$ and cluster centers $\mu_1, \mu_2, \ldots, \mu_k$ is said to have the **min-stability** property if for all $i, 1 \leq i \leq 1$, any subset $X' \subsetneq X_i$ and any other cluster X_j, $d_{min}(X', X_i \setminus X') < d_{min}(X', X_j)$.

We will now show that for $\alpha > 3$, α-multiplicative perturbation resilient data instances satisfy the **min-stability** property.

Lemma 4.3.2: Let X be any α-multiplicative perturbation resilient clustering instance. Let $X_1, X_2, \ldots, X_k$ be the optimal clusterings with centers $\mu_1, \mu_2, \ldots, \mu_k$ respectively. Let $X' \subsetneq X_i$ and $x \in X'$ for some $i, 1 \leq i \leq k$ be any point in X' and $y \in X_j, 1 \leq j \leq k, j \neq i$ be any point in some other cluster X_j which obtain the minimum distance $d_{min}(X', X_j)$. Then, for $\alpha \geq 3$, there exists $x' \in X_i \setminus X'$ such that $d(x, x') < d(x, y)$.

Proof. Let $x' \in X_i \setminus X'$ be the point which minimizes the distance $d(x, x')$. There are two cases: $\mu_i \in X'$ and $\mu_i \notin X'$.

Case(a) $\mu_i \notin X'$. This follows trivially because of the **weak center proximity** property.

Case(b) $\mu_i \in X'$. Suppose for the sake of contradiction that $d_{min}(X', X_i \setminus X') \geq d_{min}(X', X_j)$. Then $d(x', \mu_i) \geq d(x, x') > (\alpha - 1)d(x, \mu_i) = (3 - 1)d(x, \mu_i) = 2d(x, \mu_i)$. So $d(x, \mu_i) < \frac{1}{2}d(x', \mu_i)$. We also see that $d(y, \mu_i) > \alpha d(y, \mu_j) = 3d(y, \mu_j)$. Therefore, $d(y, \mu_j) < \frac{1}{3}d(y, \mu_i) \leq \frac{1}{3}[d(y, x) + d(x, \mu_i)] < \frac{1}{3}[d(x', \mu_i) + \frac{1}{2}d(x', \mu_i)]$. Then, $d(y, \mu_j) < \frac{1}{2}d(x', \mu_i)$. Now, $d(x', \mu_j) \leq d(x', \mu_i) + d(\mu_i, x) + d(x, y) + d(y, \mu_j) < 3d(x', \mu_i)$. This violates the center proximity property of α- multiplicative perturbation resilient instances. Thus, we see that $d(x, y) > d(x, x')$. Hence, for $\alpha \geq 3$, α-multiplicative perturbation resilient instances follow the *min-stability* property. □

Theorem 4.3.3: For an instance X with min-stability property, procedure **BUILDTREEMODIFIED** of Algorithm 2 produces an MST in which there are $(k - 1)$ intercluster edges and $(n - k)$ intracluster edges. Also, in the array *arrayOrder*, every cluster X_i with size $|X_i|$ occupies $|X_i|$ adjacent positions. We prove the above theorem with the help of the following lemma.

Lemma 4.3.4: Let $X = \{X_1, X_2, .., X_k\}, X_i \cap X_j = \phi$ be any data instance for which the min-stability property holds. Let $x \in X_i$ be the first point from any cluster X_i which is added to the MST. Then no point $y \in X_j$, $j \neq i$ is added to the MST until X_i is completely added to the MST.

Proof. Let $x \in X_i$ be the first point which is added to the MST. By virtue of the min-stability property, there exists a point $x' \in X_i$ which is added by Prim's algorithm next. Continuing is this way, the entire cluster X_i is added before any $y \in X_j$ is added. When X_i has been completely added, an intercluster edge (x', y'), $x' \in X_i$, $y' \in X_j$, $j \neq i$ is added and the MST enters the cluster X_j. Now, the MST can grow in two ways, either through the addition of intercluster edges or through addition of intracluster edges. Min-stability ensures that no intercluster edge from X_j is added before X_j is completely added to the MST. Let (x'', y'') be the minimum weight intercluster edge from X_i to $X_{i'}, i' \neq j$. Now, $d(x'', y'') \geq d(x', y')$ since (x', y') is added to the MST first. Again min-stability ensures that until X_j has been completely added to the MST, there exists points (z_1, z_2) with $z_1, z_2 \in X_j$ and z_1 in the MST such that $d(z_1, z_2) < d(x', y') \leq d(x'', y'')$. Thus X_j is added to the MST before any othe intercluster edge is added. By induction, we see that, Prim's algorithm ensures the lemma. □

Algorithm 2 Clustering with a noisy oracle

```
input: the set of input points $X$
output: the clusters obtained $X_1, X_2, \ldots, X_k$ after estimating the correct number of clusters, $k$
procedure BuildTreeModified($X$)
    Create a complete graph($X', E$), with $X$ as the vertices and the distance between points in $X$ as the edge weights.
    Build the Minimum Spanning Tree $T$ on $X'$ using Prim's algorithm and build an array *arrayOrder* of size $n$ to store the order in which all points were added to T. Also, maintain an array *arrayParents* which stores the parents of all points in $T$
end procedure
procedure BuildClustersNoisy($T$)
    for every edge $e = (u, v) \in T$ do
        Find the $\ell$ points appearing before $u$ in *arrayOrder*, $U = \{u_1, u_2, \ldots, u_\ell\}$
        if no such $\ell$ points exist before $u$ then
            continue
        end if
        Find the $\ell$ points appearing after $v$ in *arrayOrder*, $V = \{v_1, v_2, \ldots, v_\ell\}$
        if no such $\ell$ points exist after $v$ then
            continue
        end if
        count $\leftarrow 0$
        for every pair $(x, y)$ where $x \in U$ and $y \in V$ do
            Query the noisy oracle $\mathcal{O}$ whether $x$ and $y$ belong to the same cluster
            if answer is 'no' then
                count $\leftarrow$ count $+1$
            end if
        end for
        if $\frac{\text{count}}{\ell^2} \geq \frac{1}{2}$ then
            Starting from $v$, start building an MST using Prim's algorithm on $X'$.
            Similarly, starting from $u$, start building an MST on $X'$.
            Let $V' = \{v'_1, v'_2, \ldots, v'_\ell\}$ be the first $\ell$ points added to the MST starting from $v$ in this way.
            Let $U' = \{u'_1, u'_2, \ldots, u'_\ell\}$ be the first $\ell$ points added to the MST starting from $u$ in this way.
            if every $v' \in V'$ lies to the right of $v$ AND every $u' \in U'$ lies to the left of $u$ then
                remove edge $e = (u, v)$ from $T$
            end if
        end if
    end for
end procedure
return $T = \{T_1, T_2, .., T_k\}$          ▷ $T_1, T_2, .., T_k$ form the optimal clusters.
```

The above theorem follows from the previous lemma. Since every cluster is completely added to the MST one at a time, we observe that an intracluster edge is added every time a cluster has been completely included. Thus the MST has $(k-1)$ intercluster edges and $(n-k)$ intracluster edges. Also since for any cluster X_i, all $|X_i|$ points are added sequentially, they must occupy $|X_i|$ adjacent positions in array *arrayOrder* since it maintains the order in which the points are added to the MST.

Like in Algorithm 1, we also need to identify and remove the intercluster edges from the MST T to obtain the optimal clusters. However in this case, since the oracle $\mathcal{O}$ gives noisy answers, we cannot remove the edges as easily as we could with the ideal noiseless oracle.

Theorem 4.3.5: Let us consider a clustering instance X with optimal clusters $X_1, X_2, \ldots, X_k$ which follows the *min-stability* property. Let there exist an oracle $\mathcal{O} : X \times X \rightarrow \{\pm 1\}$ with an error parameter $q, 0 < q < \frac{1}{2}.\mathcal{O}$ takes as input two vertices $(u, v) \in X \times X$ and if u and v belong to the same cluster, answers 'yes' with probability $p = 1 - q$. Similarly, if u and v belong to different clusters, $\mathcal{O}$ answers 'no' with probability p. If the size of every clusters exceeds $c \log n$, for some suitable constant c, then there exists an algorithm with query complexity $O(n \log^2 n)$ and runtime $O(n^2 \log n)$ which obtains the optimal clusters with probability 1 - $O(\frac{n}{n^{\log n}})$.

We will now show that Algorithm 2 satisfies the requirements of the above theorem.

Lemma 4.3.6: Let at any point of time $(u, v) \in T$ be the edge of interest to Algorithm 2. If u and v belong to differnt clusters in the optimal clustering, then the ℓ points before u in *arrayOrder* and the ℓ points after v in *arrayOrder* belong to different clusters, assuming that all clusters have at least 2ℓ points.

Proof. Prim's algorithm and *min-stability* ensure that all points in a cluster are added to *arrayOrder* sequentially. Thus, if all clusters have at least ℓ points, then it must be true that ℓ points before u and ℓ points after v in *arrayOrder* belong to different clusters since $u \nsim_C v$. □

Lemma(Hoeffding): If $X_1, X_2, \ldots, X_n$ are independent random variables and $a_i \leq X_i \leq b_i$ for all $i \in [n]$, then

$$\Pr(\frac{1}{n}\sum_{i=1}^{n}(X_i - \mathcal{E}X_i) \geq t) \leq \exp(-\frac{2n^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2})$$

$$\Pr(\frac{1}{n}\sum_{i=1}^{n}(X_i - \mathcal{E}X_i) \leq -t) \leq \exp(-\frac{2n^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2})$$

Lemma 4.3.7: Let $c = \frac{1}{(\frac{1}{2}-p)^2}$ and $\ell = \frac{c\log n}{2}$. When (u, v) is being processed, and u and v belong to different clusters, Algorithm 2 can identify that $u \nsim_C v$ with probability at least $(1 - \frac{1}{n^{\log n}})$.

Proof. Let Y be a random variable denoting how many queries for the edges $(x, y), x \in U, y \in V$ receive an answer 'no' from oracle $\mathcal{O}$. Then $Y \sim \text{Binomial}(\ell^2, p)$, where p is the probability with which the oracle gives a wrong answer to a query. Hence, $\mathcal{E}[Y] = \ell^2 p$. The probability with which an intercluster edge is classified as an intracluster edge is given by $\Pr[Y < \frac{\ell^2}{2}] = \Pr[Y - \ell^2 p < \ell^2/2 - \ell^2 p] = \Pr[Y - \ell^2 p < \ell^2(\frac{1}{2} - p)] = \Pr[Y - \ell^2 p < \delta\ell^2]$ where $\delta = \frac{1}{2} - p$. By Hoeffding's inequality for binomial random variables, the probability of wrong classification is given by $\Pr[Y - \ell^2 p < \delta\ell^2] \leq e^{-2\delta^2\ell^2}$. Since $\delta = \frac{1}{2} - p$, this probability $= e^{-2(1/2-p)^2\ell^2}$. Plugging in $\ell = \frac{1}{2(1/2-p)^2}\log n$, we get $\Pr[\text{wrong classification}] \leq e^{-\log^2 n} = n^{-\log n} = \frac{1}{n^{\log n}}$. Hence probability that intercluster edge is classified correctly is $\geq 1 - \frac{1}{n^{\log n}}$. □

Lemma 4.3.8: Let $c = \frac{1}{(\frac{1}{2}-p)^2}$ and $\ell = c/2$. When (u, v) is being processed, and u and v belong to the same cluster, Algorithm 2 can identify that $u \sim_C v$ with probability $(1 - \frac{1}{n^{\log n}})$ if the sets U and V belong to the same cluster.

Proof. Let Y' be a random variable denoting how many queries for the edges $(x, y), x \in U, y \in V$ receive an answer 'no' from oracle $\mathcal{O}$. Then $Y \sim \text{Binomial}(\ell^2, p)$, where p is the probability with which the oracle gives a wrong answer to a query. Hence, $\mathcal{E}[Y'] = \ell^2(1 - p)$. The probability with which an intracluster edge is classified as an intercluster edge is given by $\Pr[Y' > \frac{\ell^2}{2}]$. The rest of the proof is similar to that of the last lemma. □

Lemma 4.3.9: Let $(u, v) \in T$ and $u \nsim_c v$. Compute V' where V' is the set of the ℓ points which are added first when we build an MST starting from v. Then V' lies entirely to the right of v. Similarly, the ℓ points added first when building an MST starting from u lie to the left of u.

Proof. u and v belong to different clusters and let $v \in X_j$. We have already shown that the cluster corresponding to the first point added to the MST is added before any other cluster is touched. Since each cluster has at least 2ℓ points, the first ℓ points added when starting from v, $V' = \{v'_1, v'_2, \ldots, v'_\ell\}$ belong to X_j. Since X_j lies to the right of v, V' must also lie to the right of v. $\square$

Lemma 4.3.10: Let $(u, v) \in T$ and $u \sim_c v$. Compute V' where V' is the set of the ℓ points which are added first when we build an MST starting from v. Compute U' where U' is the set of the ℓ points which are added first when we build an MST starting from u. If $V \not\subset X_i$, then there exists either a point $v' \in V'$ which lies to the left of v or a point $u' \in U'$ which lies to the right of u.

Proof. The first ℓ points added to the MST starting from v must belong to X_i since $v \in X_i$. Now, because $V \not\subset X_i$, there exists a point $x \in V$ such that $x \in X_j$ and there are less than ℓ points to the right of v which belong to X_i. Now, if we build an MST starting from v, then at least one of the first ℓ points must lie to the left of v. $\square$

Combining the above two lemmas, we see that for the case when u and v belong to the same cluster, if $V \not\subset X_i$, then there exists a point $v' \in V'$ where V' is the set of $\ell + 1$ closest points to v such that v' appears before v in *arrayOrder*. However, if u and v do not belong to the same cluster, then the V' lies entirely to the right of v.

Lemma 4.3.11: If we cannot find ℓ points before u and ℓ points after v, then the edge (u, v) is an intracluster edge.

Proof. Suppose, for the sake of contradiction that (u, v) is an intercluster edge. Then, there exists a cluster X_j whose first point in the MST is v or a cluster $X_{j'}$ whose last point in the MST is u. Now, since all points from a cluster are added sequentially to the MST and every cluster has 2ℓ points, there must exist ℓ points to the right of v and ℓ points to the left of u, which is a contradiction. $\square$

We now go on to prove the theorem. If u and v belong to different clusters, then lemma shows that we can detect the edge with probability $(1 - \frac{1}{n^{\log n}})$. When u and v belong to the same cluster and U and V belong to the same cluster, lemma proves that we can correctly classify the edge with probability $(1 - \frac{1}{n^{\log n}})$. The problem arises when U and V do not belong to the same cluster, i.e. $V \not\subset X_i$. In that case, classifying the edge based on whether most of the edges (u, v) where $u \in U, v \in V$ evoke 'no' answer from oracle $\mathcal{O}$ can be erronous. Two cases are possible: (a) Most of the edges evoke 'yes' answer from $\mathcal{O}$. In that case, the algorithm classifies the edge as intracluster and we have no problem. (b) However, if most edges (u, v) evoke 'no' from $\mathcal{O}$, the algorithm would classify it as an intercluster edge. To prevent this, we add an extra check. Lemma 4.2.9 ensures that when $(u', v') \in T$ belong to different clusters, V' lies to the right of v. Lemma 4.2.10 ensures that this is always false when u and v belong to the same cluster and $V \not\subset X_i$. Hence, adding a check based on this would allow us to correctly classify this case. Let $\ell = \frac{c \log n}{2}$. Then for each edge in T, we make $O(\log^2 n)$ queries. Since we query $(n-1)$ edges at most, the total query complexity is $O(n \log^2 n)$. Also, creating the MST takes $O(n^2 \log n)$ time. The rest of the algorithm is assymptotically bounded by this. Hence the overall time complexity of the algorithm is $O(n^2 \log n)$.

Now we have to prove that we find the optimal clusters with high probability. Lemmas 4.2.6 and 4.2.7 prove that we can classify each edge in T correctly with probability at least $(1 - \frac{1}{n^{\log n}})$. Thus probability of failure to classify correctly for each edge is bounded by $\frac{1}{n^{\log n}}$. The probability that at least one query fails to give the correct answer is bounded by the union bound to $\frac{n}{n^{\log n}}$. Thus, our algorithm finds the correct answer with probability $(1 - \frac{n}{n^{\log n}})$. This proves the theorem.

Chapter 5

Conclusion

In this dissertation, we have done the following:

- We studied one particular notion of stability, namely the α-multiplication perturbation stability.
- For this notion of perturbation resilience, we used semi-supervision via a noisy oracle and an ideal oracle to provide algorithms which obtain the optimal clusters.
- For the ideal oracle, we have presented polynomial time algorithms to find the optimal clusters.
- For the noisy oracle, we have presented a polynomial time algorithm which finds the optimal clusters with very high probability.
- It must be noted that our algorithms do not need the number of clusters k as an input. The stability notions allow us to derive structural properties in data instances, which we can exploit through semi-supervision to obtain the clusters.

There are numerous ways in which we would like to carry this line of research forward. Studying other notions of stability and solving their instances with semi-supervision is one such way. Different objective functions such as k-means, k-medians would help us to understand the structure of data which are stable to perturbations with these objective functions. They would enable us to impose more restrictions on the data and get better results.

Bibliography

[1] Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, 5, April 2009. URL `http://proceedings.mlr.press/v5/ackerman09a/ackerman09a.pdf`.

[2] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2): 129–137, September 2006. ISSN 0018-9448. doi: 10.1109/TIT.1982.1056489. URL `http://dx.doi.org/10.1109/TIT.1982.1056489`.

[3] David Arthur and Sergei Vassilvitskii. kmeans++: the advantages of careful seeding. *Proceedings of the eighteenth annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, January 2007. URL `https://dl.acm.org/citation.cfm?id=1283383.1283494`.

[4] Dan Pelleg and Andrew Moore. Accelerating exact k-means algorithms with geometric reasoning. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 277–281, New York, NY, USA, 1999. ACM. ISBN 1-58113-143-7. doi: 10.1145/312129.312248. URL `http://doi.acm.org/10.1145/312129.312248`.

[5] P. S. Bradley, Usama Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, pages 9–15. AAAI Press, 1998. URL `http://dl.acm.org/citation.cfm?id=3000292.3000295`.

[6] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5):1299–1319, July 1998. ISSN 0899-7667. doi: 10.1162/089976698300017467. URL `http://dx.doi.org/10.1162/089976698300017467`.

[7] James C. Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 10(2):191 – 203, 1984. ISSN

0098-3004. doi: https://doi.org/10.1016/0098-3004(84)90020-7. URL http://www.sciencedirect.com/science/article/pii/0098300484900207.

[8] Nargess Memarsadeghi, David M. Mount, Nathan S. Netanyahu, and Jacqueline Le Moigne. A fast implementation of the isodata clustering algorithm. *International Journal of Computational Geometry and Applications*, 17, February 2007. URL https://doi.org/10.1142/S0218195907002252.

[9] Dan Pelleg and Andrew W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 727–734, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-707-2. URL http://dl.acm.org/citation.cfm?id=645529.657808.

[10] Greg Hamerly and Charles Elkan. Learning the k in k-means. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 281–288. MIT Press, 2004. URL http://papers.nips.cc/paper/2526-learning-the-k-in-k-means.pdf.

[11] M. A. Stephens. EDF statistics for goodness of fit and some comparisons. *Journal of American Statistical Association*, 69(347):730–737, 1974.

[12] URL https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set.

[13] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1. URL http://dl.acm.org/citation.cfm?id=645530.655669.

[14] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. pages 333–344, April 2004. URL http://www.cs.utexas.edu/users/ml/papers/semi-sdm-04.pdf.

[15] Yonatan Bilu and Nathan Linial. Are stable instances easy? *Comb. Probab. Comput.*, 21(5):643–660, September 2012. ISSN 0963-5483. doi: 10.1017/S0963548312000193. URL http://dx.doi.org/10.1017/S0963548312000193.

[16] Shai Ben-David. Computational feasibility of clustering under clusterability assumptions. *CoRR*, abs/1501.00437, 2015.

[17] Aravindan Vijayaraghavan, Abhratanu Dutta, and Alex Wang. Clustering stable instances of euclidean k-means. In I. Guyon, U. V. Luxburg,

S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6500–6509. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7228-clustering-stable-instances-of-euclidean-k-means.pdf`.

[18] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 1068–1077, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics. URL `http://dl.acm.org/citation.cfm?id=1496770.1496886`.

[19] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *CoRR*, abs/1009.3594, 2010.

[20] Shalev Ben-David and Lev Reyzin. Data stability in clustering: A closer look. *CoRR*, abs/1107.2379, 2011.

[21] Jon Kleinberg. An impossibility theorem for clustering. pages 446–453. MIT Press, 2002.

[22] Shai Ben-David and Margareta Ackerman. Measures of clustering quality: A working set of axioms for clustering. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 121–128. Curran Associates, Inc., 2009. URL `http://papers.nips.cc/paper/3491-measures-of-clustering-quality-a-working-set-of-axioms-for-clustering.pdf`.

[23] Scott Epter, Mukkai Krishnamoorthy, and Mohammed Zaki. Clusterability detection and initial seed selection in large data sets. Technical report, The International Conference on Knowledge Discovery in Databases, 1999.

[24] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. *J. ACM*, 59(6):28:1–28:22, January 2013. ISSN 0004-5411. doi: 10.1145/2395116.2395117. URL `http://doi.acm.org/10.1145/2395116.2395117`.

[25] B. Zhang. Dependence of clustering algorithm performance on clustered-ness of data. 2001.

[26] Maria-Florina Balcan and Yingyu Liang. Clustering under perturbation resilience. *CoRR*, abs/1112.0826, 2011.

[27] Hassan Ashtiani and Shai Ben-David. Representation learning for clustering: A statistical framework. In *UAI*, pages 82–91. AUAI Press, 2015.

[28] Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3216–3224. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6449-clustering-with-same-cluster-queries.pdf`.

[29] Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *NIPS*, pages 5790–5801, 2017.