On Model-Based Semi-Supervised Clustering

by

Jordan Yoder

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

May, 2016

Abstract

We consider an extension of model-based clustering to the semi-supervised case, where some of the data are pre-labeled. We provide a derivation of the Bayesian Information Criterion (BIC) approximation to the Bayes factor in this setting. We then use the BIC to the select number of clusters and the variables useful for clustering. We discuss some considerations for $O(1)$ terms in information criteria when performing model-based clustering.

Next, we explore a novel method for the initialization of the EM algorithm for the semi-supervised case using modifications to the `k-means++` algorithm to account for the labels. Then, we derive an improved theoretical bound on expected cost and observe improved performance in simulated and real data examples. This analysis provides theoretical justification for a typically linear time semi-supervised clustering algorithm. We show how this algorithms outperforms related semi-supervised `k-means`-style algorithms on several datasets.

Finally, we demonstrate semi-supervised model based clustering with our improved `k-means++` initialization on two applications. First, we identify behaviotypes in a fly

ABSTRACT

larva dataset. Next, we nominate interesting vertices in graphs using two types of supervision.

Primary Reader: Carey E. Priebe

Secondary Reader: Vince Lyzinski

Acknowledgments

I would like to thank my advisor Carey Priebe for years of support and guidance. I would also like to thank Theodore Drivas for helpful discussions.

Dedication

This thesis is dedicated to Katherine Steinhardt, Mew Mew, Ling Ling, and Theodore Drivas.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

> Don't worry if your job is small and your rewards are few. Remember that the mighty oak was once a nut like you.
>
> Unknown

Clustering is the art of partitioning data into distinct and scientifically relevant classes by assigning labels to observations. Clustering is typically performed in an unsupervised setting, where none of the observed data initially have labels. There are a myriad of applications for clustering. Clustering is often performed in exploratory data analysis. For example, one particularly romantic graduate student clustered women on the popular online dating website OKCupid. [43] By finding groups of women, he posited that he could craft a targeted dating profile in order to succeed in

the exploitation task of finding true love.

In semi-supervised learning, some of the data have labels, but others do not; the goal is typically to classify the unlabeled data by assigning them to the same classes as the labeled data. For instance, Murphy *et al.* [41] verified the authenticity of extra virgin olive oil by a spectral analysis of the oil followed by semi-supervised classification.

When the goal is instead to group the data, there is semi-supervised clustering. This task may be further complicated by the fact that the total number of classes may be unknown. Just as in semi-supervised learning, leveraging the known labels allows for a more informed clustering. There are many real-life examples of applications of semi-supervised clustering. For instance, Basu *et al.* [9] use their semi-supervised clustering algorithms to detect newsgroups from raw messages. In Chapter 4, we detect flower species from measurements and terrain type from hyper-spectral images. In Chapter 5, we identify biological behaviotypes of lobotomized maggots from data derived from time series of their actions and also produce a nomination list of interesting vertices in a graph.

We will now introduce these topics more formally and explain their relatedness.

1.1 Supervised Learning (Classification)

Suppose that we observe labeled data

$$(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n),$$

where X_i represents some object living in $\mathcal{X}$ (typically $\mathbb{R}^d$) and Y_i is the label of the object living in $\mathcal{Y}$ (typically $\mathcal{Y} \subset \mathbb{Z}$). In classification, we use the data to learn a classification rule, or a mapping

$$g : \mathcal{X} \to \mathcal{Y}.$$

[16, 29] We can separate the two main branches of classification algorithms by observing that $P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$.

The first branch, **discriminative** algorithms, directly estimates the classification rule g, often by focusing on learning $P(Y|X)$. Examples of discriminative algorithms include support vector machines (SVN), logistic regression, and k-nearest neighbors.

The second branch, **generative** algorithms, seek to model the distribution of the conditional data and the classes together (i.e. learn both $P(X|Y)$ and $P(Y)$). Then, given this learned distribution, predict the y that would be most likely for the given instance x. Examples of generative algorithms are `k-means`, `Gaussian Mixture Modeling` (GMM), and `Naive Bayes`. When comparing discriminative and generative algorithms, there is an often cited principle attributed to Vapnik, which

roughly states that one should never do anything more complicated than you have to in order to solve your problem. This is the crux of why some prefer discriminative to generative algorithms, although generative algorithms can incorporate unlabeled data and benefit from some natural ways to answer some challenging (and important) questions, like how many clusters are there.

1.2 Unsupervised Learning (Clustering)

In the unsupervised learning paradigm, we have data $X_1, X_2, \ldots, X_n \in \mathcal{X}$ and we wish to group the data into meaningful categories, or clusters. Here, meaningful is somewhat vague, but generally a desirable cluster would have strong scientific significance. For instance, consider a clustering of patients with similar physiological traits where one cluster of patients responds very well to a treatment. Such a clustering may give rise to further study of those traits and how they are related to (or perhaps even cause) the positive treatment outcome. This is the most optimistic outcome of clustering.

Typically clustering in toy examples have "obvious" categories, such as species of Iris flower [19], and thus it is readily apparent how good a clustering is by comparing the partitioning to the true (and known) partition. In this way, we can view clustering as classification with unlabeled data. However, when the number of groups is unknown (and therefore must be estimated), such a correspondence is not so simple. Figure 1.1

represents a clustering problem, where the examples are colored and shaped according to their true (and unknown!) cluster, where cluster corresponds to which distribution the point was drawn from. The problem at hand would then be to partition the clusters into 1 or more groups, with the hope of recovering some meaningful clusters.

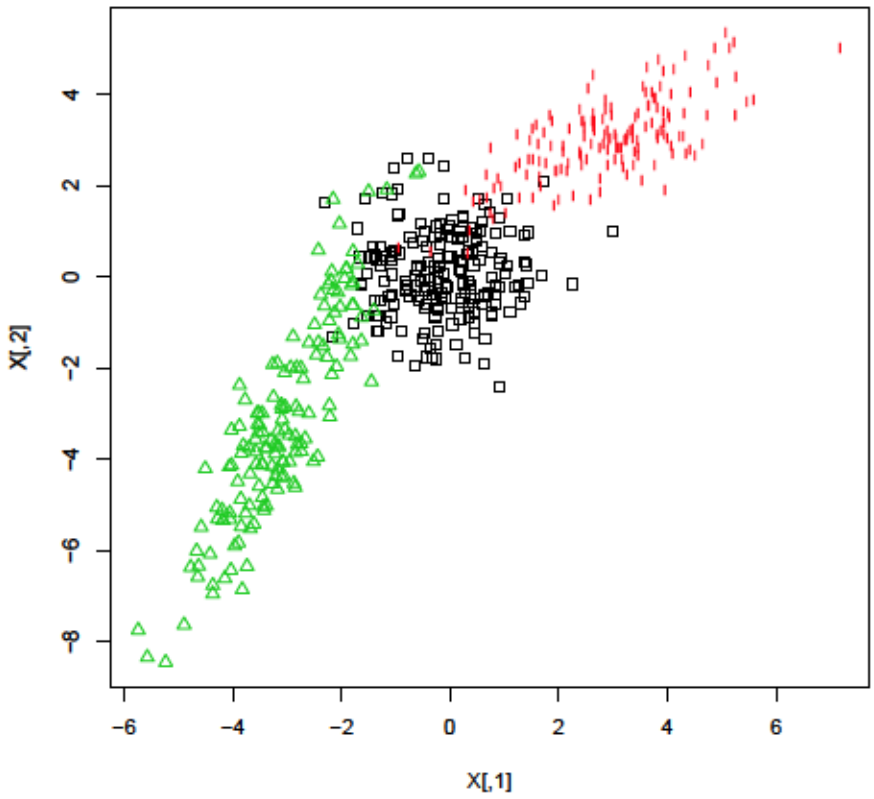


Figure 1.1: An example of a clustering problem. Colors represent the true clusters.

There are many strategies for clustering. Some of these are based on heuristics like "clusters are formed by similar objects," where similar might mean having small pairwise object distance. Others are model-based and use a learned generative distribution to model the data. Model-based methods allow for somewhat straightforward

incorporation of supervision type constraints (cf. Section 3.1).

1.3 Semi-supervised Learning

Semi-supervised learning problems have data with labels:

$$(X_1^s, Y_1^s), (X_2^s, Y_2^s), \ldots, (X_n^s, Y_n^s),$$

without labels:

$$X_1^u, X_2^u, \ldots, X_m^u,$$

and/or data with hard or soft constraints such as X_{i_1} and X_{i_2} must be (must not be) in the same cluster. The constraints are more general than labeled data, since particular pairwise constraints can transmit identical information as labels. Generally, when we refer to semi-supervised problems, we will use only the first two types of data. That is, we will only allow the data to be labeled and unlabeled, with no additional pairwise constraints.

1.4 Semi-supervised Clustering

Semi-supervised clustering is a semi-supervised learning problem where there are labeled and unlabeled data and the full set $\mathcal{Y}$ is probably unknown and there are possibly no exemplars for some of the groups. Figure 1.2 depicts a clustering problem,

where there are 40 points in $\mathbb{R}^2$, fifteen of which have labels already, as represented by the circle shape instead of the star. Note that one of the clusters (denoted by the color), has no labeled points–it is all stars. This is typical of semi-supervised clustering.

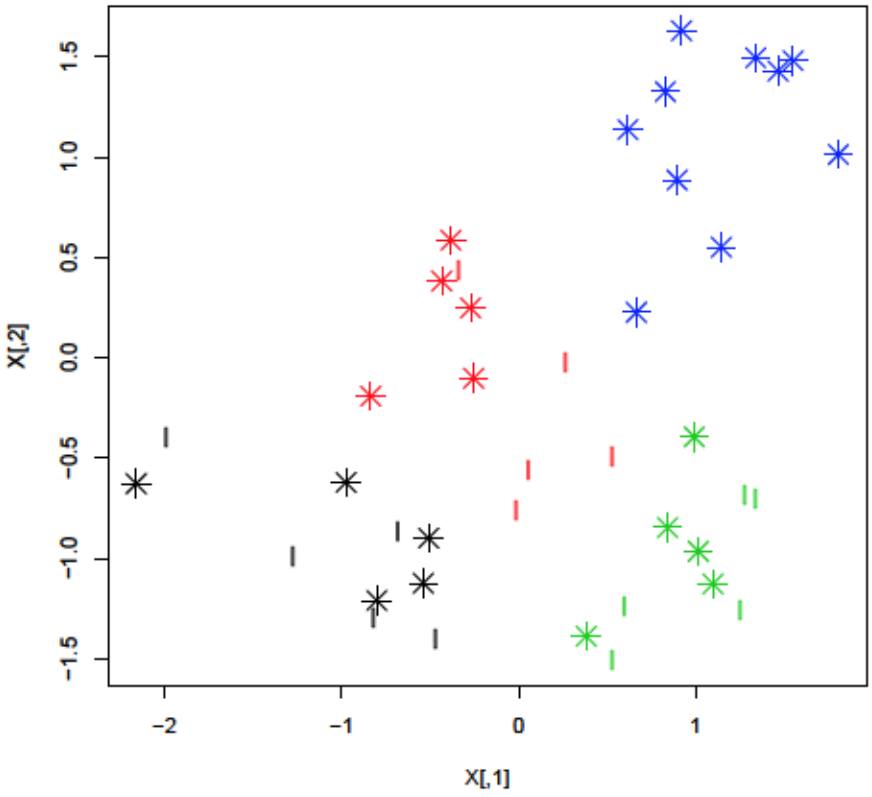


Figure 1.2: An example of a semi-supervised clustering problem. Colors represent the true clusters. Symbol represents unlabeled (star) or labeled (circle).

Chapter 2

Literature Review: Clustering

> Learn from the mistakes of others.
> You can never live long enough to
> make them all yourself.
>
> —— Groucho Marx

Some of this chapter appears in the introductions in [58, 59].

There are a wealth of strategies for solving learning problems. Due to the related nature of clustering and semi-supervised clustering, we first expound on some selected strategies for clustering, then detail the modifications involved for the semi-supervised case either in this or subsequent chapters.

2.1 Hierarchical

Hierarchical clustering strategies focus on building a dendrogram that can be cut at a specific level to produce a desired number of clusters. Clusters are built on top of the previous level's by either dividing (divisive hierarchical clustering) or combining clusters (agglomerative hierarchical clustering) using an optimization of an objective function. Hierarchical clustering has several pros: it can produce any desired number of clusters (between 1 and n, where n is the total number of points to be clustered); it is relatively simple to explain; and it is easy to implement (naively). Its main con is that it's slow. Most implementations are $\mathcal{O}(n^3)$ and "efficient" versions are still $\mathcal{O}(n^2)$. Thus, it is not well suited for big data.

The key difference between different hierarchical clustering methods is the choice of objective function as the clustering criterion. Let $\mathcal{X} \subset \mathbb{R}^d$ be the data. Suppose A and B are two clusters (consisting of elements of $\mathcal{X}$) at the current level. Often, we choose a **distance** function $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ acting on pairs of points. Some selected distance functions are

1. L_p distance: $d(x, y) = \|x - y\|_p = \left(\sum_{i=1}^{d}(x_i - y_i)^p\right)^{\frac{1}{p}}$ for $p \geq 1$ (particularly L_2 or Euclidean distance)

2. Ellipsoidal or Mahalanobis distance: $d(x, y) = \langle (x-y), A^{-1}(x-y) \rangle$ for positive definite matrix A

We still need to define the distance function $g : 2^{\mathcal{X}} \times 2^{\mathcal{X}} \to \mathbb{R}^+$ between A and

B, sets of points, since merging (dividing) clusters is based on the optimization over pairs of points. Generally, this is done by involving d on pairs of points in the two clusters. Some selected examples of distance functions between sets of points are

1. Maximum linkage: $g(A, B) = \max_{x \in A, y \in B} d(x, y)$
2. Minimum linkage: $g(A, B) = \min_{x \in A, y \in B} d(x, y)$
3. Average linkage: $g(A, B) = \frac{1}{card(A)card(B)} \sum_{x \in A, y \in B} d(x, y)$
4. Likelihood based where (a) mixture models are posited for the data (b) $g(A, B)$ is the change in overall (maximized) likelihood if a merge were to occur (cf. [23] for more details)

Once a distance function is defined, the hierarchical clustering generally starts with n or 1 clusters for agglomerative or divisive algorithms, respectively. Then, over n rounds, a single merge or split is performed based on what would be optimal according to the distance function applied to all current clusters. See Algorithm 1 for pseudocode of an agglomerative hierarchical scheme. That particular pseudocode is too general to incorporate some standard shortcuts that apply to particular link/distance functions. Figure 2.1 depicts a hierarchical clustering of US Arrests data by state from the USArrests dataset included in R. [38] This data is given as the number of arrests per 10^5 individuals for 3 violent crimes and population.

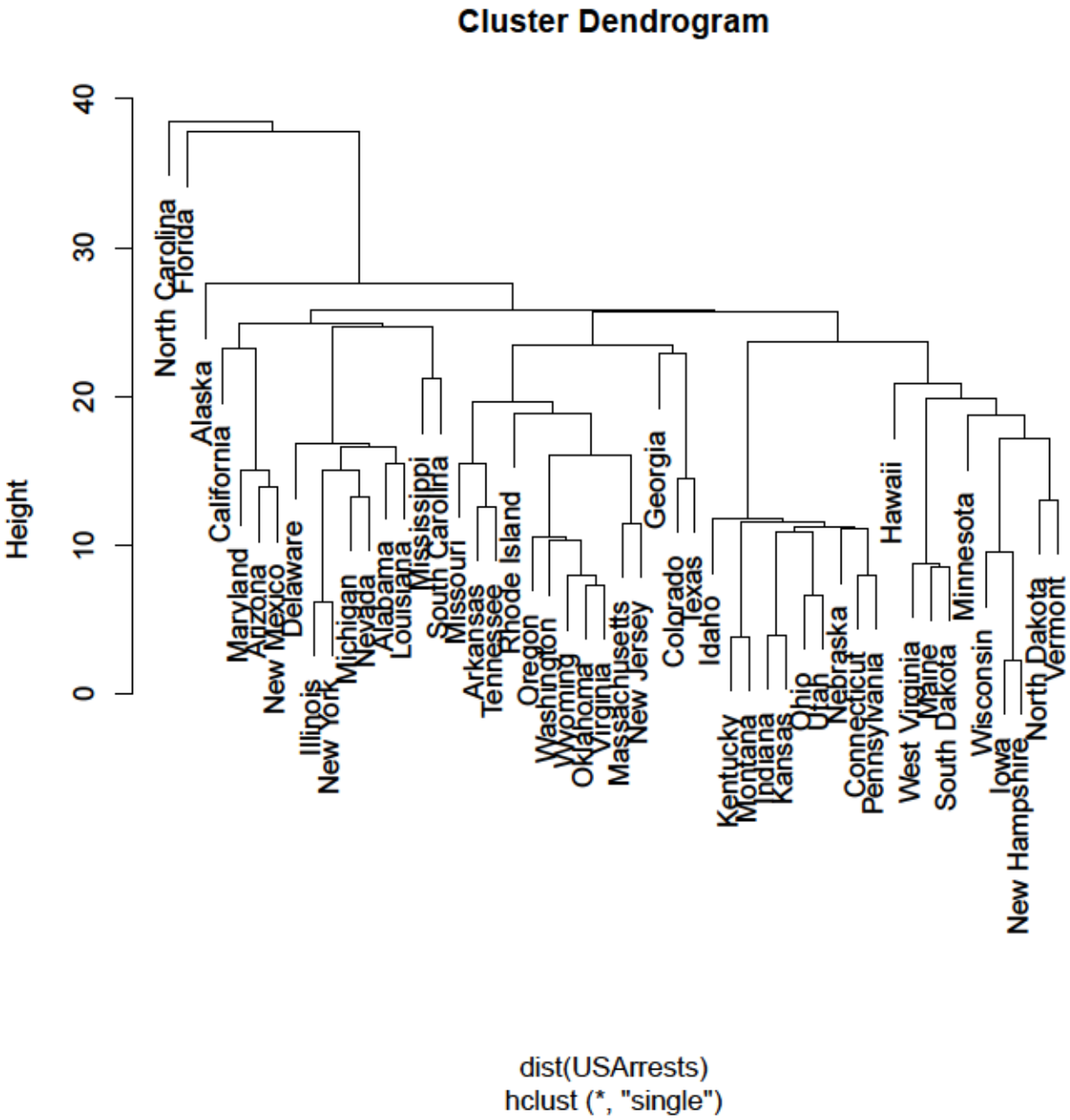


Figure 2.1: Dendrogram of agglomerative hierarchical clustering applied to US Arrest data using the maximum linkage with Euclidean distance objective.

Algorithm 1: Agglomerative Hierarchical Clustering

Input: $\mathcal{X}$ (n datapoints)
$g : 2^{\mathcal{X}} \times 2^{\mathcal{X}} \to \mathbb{R}^+$ (cluster distance function)
k (desired number of clusters)
Output: $\ell : \mathcal{X} \to \{1, 2, \ldots, k\}$ (cluster assignment mapping)

Define $A_i = \{x : \ell(x) = i\}$ for $i = 1, 2, \ldots, n$ be the current clusters.
Let $K = \{1, 2, \ldots n\}$ be the labels of the current clusters.
Define $\ell(x_i) = i$ for $i = 1, 2, \ldots, n$.
repeat
  Let $(i^*, j^*) = \text{argmin}_{i,j \in K \text{ with } i<j} g(A_i, A_j)$.
  Merge clusters A_{i^*} and A_{j^*}. Remove j^* from K.
until $card(K) > k$
Relabel all clusters so that if $K = \{a_1, a_2, \ldots, a_k\}$, A_{a_j} becomes A_j.
Update $\ell(x) = j$ for each $x \in A_j$ for $j \in K$.
return ℓ

2.1.1 Semi-supervised Hierarchical Clustering

One of the main difficulties in semi-supervised learning is incorporating the supervision information. Several authors have noted that the general must link and cannot link pairwise constraints are not appropriate for hierarchical clustering due to the multiple levels of clustering (see, for example, [60, 7]). Namely, how can any pairs have a must-link constraint when all clusters are initially size 1? Zheng and Li modified the constraints allowed to "must link before" and using a modified pairwise distance matrix using an *uberdistance* that distorts the original pairwise distances to take into account the constraints. Due to fact that these modifications are necessary for semi-supervised hierarchical clustering to be implemented, we prefer to use other semi-supervised methods.

2.2 Parametric

2.2.1 Model Based Clustering

Some clustering procedures are based on heuristics that lack a principled justification, and many of the basic questions of clustering (e.g., how many classes there are) are often left to the intuition of the practitioner. Fraley and Raftery [24] review model-based clustering, which recasts the task of clustering as a model selection problem, which is well-studied in the field of statistics. The main assumption in model-based clustering is that the data are drawn from one of G distributions, where each distribution represents a cluster. Model selection can be accomplished by computing approximate Bayes factors for competing models with different numbers and/or types of components.

Formally, assume that the data $X_1, X_2, \ldots, X_n$ are distributed i.i.d. according to a mixture model,

$$f_\theta(\cdot) = \sum_{k=1}^{G} \pi_k f_{\theta_k}(\cdot),$$

where θ_k are the parameters of the k^{th} component of the mixture and G is the total number of components. It can be shown that this is equivalent to the following generative process: for each $i \in \{1, 2, \ldots, n\}$

(1) draw Z_i from multinomial $(\pi_1, \pi_2, \ldots, \pi_G)$

(2) draw X_i from $f_{\theta_{Z_i}}$.

If we consider each of the G components as representing a single cluster, then the problem of clustering the data is that of unveiling each latent Z_i from the generative process. The expectation-maximization (EM) algorithm can be used to find the maximum likelihood estimates for the parameters of the model and the posterior probabilities of cluster membership for each X_i (cf. [15]).

With this formulation, we have a fully probabilistic clustering scheme. We can set the cluster of the i^{th} observation to the maximum a posteriori estimate:

$$\hat{Z}_i = \text{argmax}_k \pi_k f_k(x_i).$$

Note that the posterior probability can give us a measure of confidence about the i^{th} classification.

Here, we have used f_{θ_k} as a general density function, but it should be noted that the multivariate Gaussian distribution is the most common choice of distribution. We will eventually use an information criterion, such as the Bayesian Information Criterion (BIC), to assess the relative quality of different clusterings. Then, the number of clusters can be chosen using the BIC, which rewards model fit and penalizes model complexity.

Figure 2.2 depicts the model based clustering of the famous Old Faithful dataset. [2] We have 272 observations on 2 variables: eruption duration and waiting time until the next eruption. We use Mclust, [25] an R implementation of Gaussian Mixture

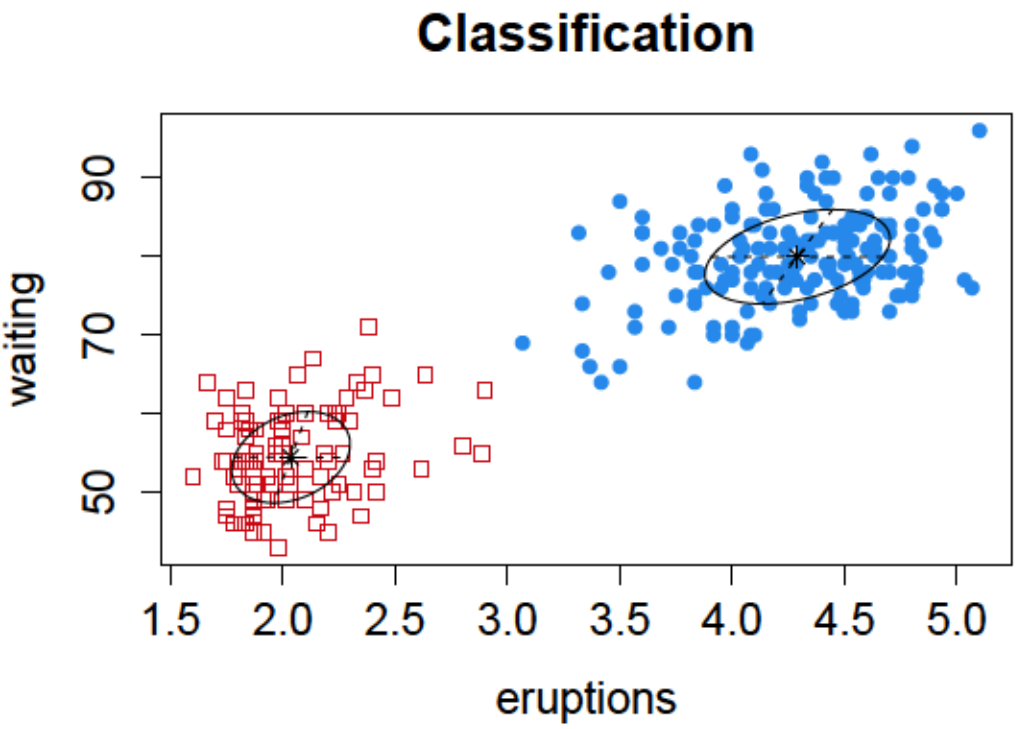


Figure 2.2: Model based clustering of the Old Faithful dataset. Colors/shapes represent cluster assignments. Black ellipsoids represent orientations of the two components in the final model.

Modeling (GMM) (model based clustering using Gaussian components), to cluster the observations by allowing it to consider between 2 and 5 components with different covariance parameterizations allowed (VII, VVI, and VVV, cf. Table 2.1 for descriptions). Note that it selects two components, which matches with the intuitive "by eye" clustering.

By considering different constraints to the covariance matrices in the Gaussian components, we can reduce the number of parameters estimated, thus lowering the influence of the penalty term in the BIC. This allows for simpler clusters to be chosen

over complex clusters. Celeux and Govaert [13] consider a spectral decomposition of the k^{th} component's covariance matrix,

$$\Sigma_k = \lambda_k D_k^T A_k D_k,$$

where λ_k represents the largest eigenvalue, A_k is a diagonal matrix whose largest entry is 1, and D_k is a matrix of the corresponding eigenvectors. The interpretation of each term as it relates to cluster k is as follows: λ_k represents the volume, D_k the orientation, and A_k the shape. By forcing one or more of these terms to be the same across all clusters, we can reduce the number of parameters to be estimated from $Gd + G\frac{d(d+1)}{2}$ in the unconstrained case ($\Sigma_k = \lambda_k D_k^T A_k D_k$) to $Gd + G - 1 + \frac{d(d+1)}{2}$ in the most constrained case ($\Sigma_k = \lambda D^T A D$). We can also force additional constraints, such $D_k = I$ and/or $A_k = I$, leading to the simplest model: $\Sigma_k = \lambda I$ with only $G(d+1)$ parameters to estimate.

Mclust breaks down different choices of the constraints on the covariance matrices using short (≤ 3) letter descriptions (cf. Table 2.1). Multivariate parameterizations receive their by the concatenation of 3 letters. Position 1 indicates either (E)qual or (V)arying volume (i.e. λ_k). Position 2 indicates (I)dentity, (E)qual, or (V)arying shape (i.e. A_k). Position 3 indicates (I)dentity, (E)qual, or (V)arying orientation (i.e. D_k). Univariate parameterizations do not have shape or orientation, and thus only are named by one letter.

Name	Applicable To	Σ_k	Volume	Shape	Orientation
E	$\mathbb{R}$	λ	Equal	NA	NA
V	$\mathbb{R}$	λ_k	Varying	NA	NA
X	$\mathbb{R}, G = 1$	λ	NA	NA	NA
EII	$\mathbb{R}^d$	λJ	Equal	Equal, spherical	Coordinate axes
VII	$\mathbb{R}^d$	$\lambda_k J$	Varying	Equal, spherical	Coordinate axes
EEI	$\mathbb{R}^d$	λA	Equal	Equal, ellipsoidal	Coordinate axes
VEI	$\mathbb{R}^d$	$\lambda_k A$	Varying	Equal, ellipsoidal	Coordinate axes
EVI	$\mathbb{R}^d$	λA_k	Equal	Varying, ellipsoidal	Coordinate axes
VVI	$\mathbb{R}^d$	$\lambda_k A_k$	Varying	Varying, ellipsoidal	Coordinate axes
EEE	$\mathbb{R}^d$	$\lambda D^T A D$	Equal	Equal, ellipsoidal	Equal
EVE	$\mathbb{R}^d$	$\lambda D^T A_k D$	Equal	Varying, ellipsoidal	Equal
VEE	$\mathbb{R}^d$	$\lambda_k D^T A D$	Varying	Equal, ellipsoidal	Equal
VVE	$\mathbb{R}^d$	$\lambda_k D^T A_k D$	Varying	Varying, ellipsoidal	Equal
EEV	$\mathbb{R}^d$	$\lambda D_k^T A D_k$	Equal	Equal, ellipsoidal	Varying
VEV	$\mathbb{R}^d$	$\lambda_k D_k^T A D_k$	Varying	Equal, ellipsoidal	Varying
EVV	$\mathbb{R}^d$	$\lambda D_k^T A_k D_k$	Equal	Varying, ellipsoidal	Varying
VVV	$\mathbb{R}^d$	$\lambda_k D_k^T A_k D_k$	Varying	Varying, ellipsoidal	Varying
XII	$\mathbb{R}^d, G = 1$	λJ	NA	Spherical	Coordinate Axes
XXI	$\mathbb{R}^d, G = 1$	λA	NA	Ellipsoidal	Coordinate Axes
XXX	$\mathbb{R}^d, G = 1$	$\lambda D^T A D$	NA	Ellipsoidal	NA

Table 2.1: Mclust covariance parameterizations. Let J be the identity matrix. Expanded from Table 1 in [46].

2.2.1.1 Model Selection in Model Based Clustering

Bayesian model selection can be accomplished through Bayes Factors, or ratios of posterior probabilities of the data. Suppose that there are two models under competition for selection, say M_1 and M_2. The Bayes Factor comparing M_1 to M_2 is

$$B_{1,2} \equiv \frac{P(D|M_1)P(M_1)}{P(D|M_2)P(M_2)},$$

where $P(D|M_1)$ is the posterior likelihood of the data under model M_1 and $P(M_1)$ is the prior probability of M_1. After calculating the Bayes Factor comparing M_1 to M_2, one would choose M_1 if $B_{1,2} > 1$ and M_2 otherwise.

The BIC for a model under consideration (indexed by the form of Σ_k and G) is defined as

$$BIC_M \equiv 2\mathcal{L}_M - d_M \log(n),$$

where $\mathcal{L}_M$ is the maximized log-likelihood under model M, d_M is the number of parameters estimated, and n is the number of points used to estimate those parameters. Heuristically, the BIC rewards model fit with the first term and penalizes model complexity with the second term. Higher values are preferred.

It can be shown that the BIC $\approx 2\log(P(D|M))$ (cf. Section 3.2 for a related derivation). Thus, under a flat prior over all models, $BIC_{M_1} - BIC_{M_2} > 0$ if and only if $B_{1,2} > 1$. Hence, calculating the BIC for all models and choosing the argmax gives the same results as computing pairwise Bayes Factors (under a flat prior). Therefore, decisions between different parameterizations of Σ_k and number of components G can be made on the basis of the BIC approximation to the log of the posterior likelihood.

2.2.2 Simultaneous Clustering and Variable Selection

Much of the recent work in model-based clustering has centered on variable selection. Raftery and Dean [46] proposed a greedy algorithm for model-based clustering in the unsupervised setting; it used the BIC to choose the number of clusters and the features to consider while clustering by proposing linear relationships between variables that influence clustering and those independent of clustering. Murphy *et al.* [41] adapted Raftery and Dean's algorithm to semi-supervised learning. Maugis *et al.* [36, 35] consider many more scenarios of dependency structures between the clustering variables and the remaining variables than in [46]. Taking a different approach, Witten *et al.* [56] offer a framework for variable selection in clustering using sparse k-means or sparse hierarchical algorithms by modifying the corresponding objective function to include penalty constraints. In their simulations, they outperform the methods of [46] considerably.

2.2.3 K-Means

`K-means` [22, 33] is one of the most widely known clustering algorithms. The basic problem it solves is as follows: for a fixed natural number k and dataset $\mathcal{X} \subset \mathbb{R}^d$, return a set of centers $C = \{c_i \in \mathbb{R}^d : i = 1, 2, \ldots k\}$ such that it is the solution to

the k-means problem.

$$C = \text{argmin}_{\{A \subset \mathbb{R}^d \text{ s.t. } |A|=k\}} \phi_A(\mathcal{X}), \tag{2.1}$$

where $\phi_A(\mathcal{X}) = \sum_{x \in \mathcal{X}} \min_{c \in A} \|x - c\|^2$. Then, using this set of centers, return one of k labels for each datum:

$$\ell(x) = \text{argmin}_{i \in \{1,2,\ldots,k\}} \|x - c_i\|.$$

Lloyd's algorithm [31] is a particularly long-lived strategy for locally solving the k-means problem (cf. Algorithm 2); it suggests randomly selecting a subset of size k from $\mathcal{X}$ as initial centers, then alternating updates to cluster assignments and new centers. Lloyd's algorithm does not have an approximation guarantee. Such a guarantee, if it existed, would have the following form:

$$\mathbb{E}[\phi(\mathcal{X})] \leq \alpha \phi_{OPT}(\mathcal{X}),$$

where ϕ_{OPT} is the optimal value of ϕ corresponding to exactly solving the k-means problem. If α was constant, then this would be an $\mathcal{O}(1)$ approximation bound. Bounds of other orders, such as those scaling with k would have an α that depended on k. Because Lloyd's algorithm does not have such an approximation bound, we have that for certain datasets, it can return centers that result in a large value of

the objective function in equation (2.1) with high probability. Thus, even running independent copies of the algorithm and choosing the best result could yield poor results.

Macqueen [33] conjectures that exactly solving equation (2.1) is difficult. He is correct; Mahajan et al. [34] show that even for two dimensional data, k-means is NP-hard. Because of this, practitioners instead seek to approximately solve the k-means problem. Arthur and Vassilvitskii [5] present `k-means++`, a randomized $\mathcal{O}(\log(k))$ approximation algorithm running in $\mathcal{O}(kn)$ time (for their initialization step, which is all that is required for the approximation bound) that works by modifying Lloyd's algorithm to choose initial centers with unequal weighting (cf. Algorithm 2). Their results are remarkable because the algorithm runs in a practical amount of time. This work inspired others to propose alternative randomized initializations for Lloyd's algorithm for streaming data [4], parallel implementations [8], and bi-approximations with extra ($> k$) centers that can then be re-clustered to yield k centers. [3, 4]

`K-means` is included as a parametric algorithm due to its relationship to model based clustering with spherical equally oriented gaussians. That is, suppose that a set of k centers $\mathcal{C}$ is already chosen. Then, for $\lambda > 0$, consider the mixture model with $X_i \overset{i.i.d.}{\sim} f(x) = \frac{1}{k}\sum_{g=1}^{k} f_g(x; c_g, \lambda I)$, where I is the identity matrix and $f_g(\cdot; c_g, \lambda I)$ is the multivariate normal pdf with mean c_g and covariance λI. In this case, any new point $x \in \mathcal{X}$ will be assigned by `K-means` to the closest center (as determined by Euclidean distance). By GMM, it will also be assigned to the closest center by

Mahalanobis distance, which for equal and spherical covariance matrices, corresponds to the Euclidean distance. Thus, the algorithms have the same decision boundary. Also, Lloyd's algorithm updates will be akin to E-M updates with hard assignments (instead of soft assignments as in typical GMM). Thus, the movement of the centers will be similar in both algorithms.

Algorithm 2: Lloyd's k-means algorithm

Input: $\mathcal{X}$ (n datapoints)
$\mathcal{C}$ (k initial centers)
Output: $\mathcal{C}$ (updated centers)

repeat
  Assign each $x_i \in \mathcal{X}$ to the nearest center $c(x_i) \in \mathcal{C}$.
  Update each $c_j \in \mathcal{X}$ as the centroid of the points $x \in \mathcal{X}$ such that $c(x) = c_j$.
until $\mathcal{C}$ *has not changed*
return $\mathcal{C}$

Algorithm 3: Initialization of centers for k-means++

Input: $\mathcal{X}$ (n datapoints)
k (number of centers)
Output: $\mathcal{C}$ (set of initial centers)

Choose an $x \in \mathcal{X}$ uniformly at random.
Let $\mathcal{C} = \{x\}$.
while $card(\mathcal{C}) < k$ **do**
  Choose a datapoint $x \in \mathcal{X}$ with probability proportional to $D^2(x)$.
  Update $\mathcal{C} = \mathcal{C} \cup \{x\}$.
return $\mathcal{C}$

2.2.3.1 Previous Works on Semi-supervised K-means

In semi-supervised learning, there is additional information available about the true labels of some of the data. These typically take the form of label information

(e.g. $\ell(x_1) = 2$) or pair-wise constrains (e.g. $\ell(x_1) = \ell(x_2)$ or $\ell(x_1) \neq \ell(x_2)$). In recent years, there has been a fair amount of interest in solving problems with these additional constraints. Wagstaff et al. [54] propose the `COP-KMeans` algorithm, which uses a modified assignment step in Lloyd's algorithm to avoid making cluster assignments that would be in violation of the constraints. Basu et al. [9] focused on using label information in their `Seeded-KMeans` and `Constrained-KMeans` algorithms. Both algorithms use the centroids of the labeled points as initial starting centers. Basu et al. [10] use the `Expectation-Maximization` (EM) algorithm [15] as a modified Llyod's algorithm to modify the pairwise supervision algorithms to include a step wherein the distance measure is modified (so that they do not necessarily use Euclidean distance). Finley and Joachims [18] learn pairwise similarities to account for semi-supervision.

Chapter 3

Model Selection and Semi-supervised Clustering

> It is better to be right than to be rigorous.
>
> —— Andrey Kolmogorov

Much of this chapter appears in [58].

We will formulate a model encompassing the semi-supervised case, derive a BIC approximation to the posterior loglikelihood under this model, then apply our result to the special case of semi-supervised clustering. The modified BIC then represents a principled measure by which to choose the number of clusters and the variables for clustering when performing semi-supervised clustering. Next, we will consider some aspects of $O(1)$ differences in information criteria for model selection in model-based

semi-supervised clustering in theory and for specific examples.

3.1 Model encompassing the semi-supervised case

Consider $n = \sum_{i=1}^{C} n_i$ independent random variables

$$
\begin{aligned}
X_1^{(1)}, X_2^{(1)}, \ldots, X_{n_1}^{(1)} &\overset{\text{i.i.d.}}{\sim} f_{\theta_1}, \\
X_1^{(2)}, X_2^{(2)}, \ldots, X_{n_2}^{(2)} &\overset{\text{i.i.d.}}{\sim} f_{\theta_2}, \\
&\vdots \\
X_1^{(C)}, X_2^{(C)}, \ldots, X_{n_C}^{(C)} &\overset{\text{i.i.d.}}{\sim} f_{\theta_C},
\end{aligned}
$$

where $\theta_1 \in \Theta_1 \subset \mathbb{R}^{d_1}$ are the parameters for the group of unsupervised data. The other rows of supervised data are drawn using distributions whose parameters are in restricted (from Θ_1) spaces; formally, $\theta_j \in \Theta_j \subset \Theta_1$ for $j = 2, 3, \ldots C$. Finally,

$$
(f_{\theta_1}, f_{\theta_2}, \ldots, f_{\theta_C}) \in M = \{(f_{\theta^{(1)}}, f_{\theta^{(2)}}, \ldots, f_{\theta^{(C)}}) : \theta = (\theta^{(1)}, \ldots, \theta^{(C)}) \in \Theta_1 \times \Theta_2 \times \cdots \times \Theta_C\}.
$$

Collect all of the $X's$ into set D.

Model M, while more general than strictly necessary, encompasses the semi-supervised case. Consider the first group of n_1 random variables as those for which

we do not have labels and each of the other $C-1$ groups as those whose labels we know. Then, for a proposed total number of clusters $G \geq C-1$, we assume

$$f_{\theta_1}(x) = \sum_{j=1}^{G} \pi_j \phi(x; \mu_j, \Sigma_j),$$

where $\phi(\cdot; \mu_j, \Sigma_j)$ is a multivariate normal pdf with mean μ_j and covariance matrix Σ_j, and $\sum_{j=1}^{G} \pi_j = 1$. Also, for each $k \in \{2, 3, \ldots, C\}$,

$$f_{\theta_k}(x) = \phi(x; \mu_{j_k}, \Sigma_{j_k}),$$

where the double subscript j_k is to account for possible relabeling. It follows that

$$\theta_1 = ((\pi_1, \pi_2, \ldots, \pi_G), (\mu_1, \mu_2, \ldots, \mu_G), (\Sigma_1, \Sigma_2, \ldots, \Sigma_G)) \in \Theta_1,$$

where

$$\Theta_1 = \{((\pi_1, \pi_2, \ldots, \pi_G) : \sum_{t=1}^{G} \pi_t = 1, \pi_t \in [0,1]\} \times \mathbb{R}^{d \times G} \times \{(\Sigma_1, \Sigma_2, \ldots, \Sigma_G) | \Sigma_t \succeq 0, \Sigma_t \in \mathbb{R}^{d \times d}\}.$$

Then, for each $k \in \{2, 3, \ldots, C\}$, Θ_k is the same as Θ_1 except that the mixing coefficients $(\pi_1, \pi_2, \ldots, \pi_G)$ are constrained such that $\pi_{j_k} = 1$ and all other $\pi_i = 0$.

With this model, we can derive the Expectation step (E-step) and Maximization (M-step) of the EM algorithm. Note that semi-supervised EM is not new; indeed, sim-

ilar calculations can be found in [37] Section 2.19 and [48]. We include our derivation of the E-step and M-step for our model below for consistent notation and completeness.

E-Step for semi-supervised clustering.

Let $Z_{i,k} = \mathbb{1}\{X_i \text{ is in cluster k}\}$ denote the (hidden) cluster memberships, where $\mathbb{1}\{.\}$ is the indicator function. Let $Z_i = \ell_i$, where ℓ_i is cluster that X_i is a member of. The likelihood of the complete data (X, Z) is

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{n} f(X_i, Z_i) \\ &= \prod_{i=1}^{n} P(Z_i = \ell_i) f(X_i | Z_i = \ell_i) \\ &= \prod_{i=1}^{n} \pi_{\ell_i} f_{\theta_{\ell_i}}(X_i) \\ &= \prod_{i=1}^{n} \left(\pi_{\ell_i} f_{\theta_{\ell_i}}(X_i)\right)^{Z_{i,\ell_i}} \\ &= \prod_{i=1}^{n} \prod_{k=1}^{G} \left(\pi_k f_{\theta_k}(X_i)\right)^{Z_{i,k}}, \end{aligned}$$

where the last step follows because only the ℓ_ith entry of $Z_{i,\cdot}$ is 1 (and the others are 0). Taking logs and substituting $\phi(\cdot; \mu_k, \Sigma_k)$ for each f_{θ_k}, the complete log-likelihood of the parameters given the data and labels is

$$l(\theta | D) = \sum_{i=1}^{n} \sum_{k=1}^{G} Z_{i,k} \log(\pi_k \phi(X_i; \mu_k, \Sigma_k)).$$

Note that for the $n - n_1$ data not in the first group, we know the true $Z_{i,k}$, but for the n_1 unlabeled data we do not. Thus, the E-step of the EM algorithm is

$$\mathbb{E}[l(\theta, Z_{i,k}|D)] = \sum_{i=1}^{n_1}\sum_{k=1}^{G}\mathbb{E}[Z_{i,k}|D]\log(\pi_k\phi(X_i;\mu_k,\Sigma_k))+\sum_{k=2}^{C}\sum_{i=n_{k-1}+1}^{n_k}\log(\phi(X_i;\mu_{j_k},\Sigma_{j_k})).$$

By the definition of $Z_{i,k}$ and Bayes Theorem, we have for all $i = 1, 2, \ldots, n_1$ that

$$\mathbb{E}[Z_{i,k}|D] = P(Z_{i,k} = 1|D) = \frac{\pi_k\phi(X_i;\mu_k,\Sigma_k)}{\sum_{j=1}^{G}\pi_j\phi(X_i;\mu_j,\Sigma_j)}.$$

M-Step for semi-supervised clustering.

Define $\rho_{i,k} = \mathbb{E}[Z_{i,k}|D]$. We must maximize $\mathbb{E}[l(\theta, Z_{i,k}|D)]$ with respect to the mixing coefficients π_k, the mean vectors μ_k, and the covariance matrices Σ_k for $k = 1, 2, \ldots, G$. It can be shown that the update equations for π_k are

$$\pi_k = \frac{\sum_{i=1}^{n_1}\rho_{i,k}}{n_1}.$$

Recalling that $\rho_{i,k} = \mathbb{1}\{X_i \text{ is in cluster k}\}$ for $i = n_1+1, n_1+2, \ldots n$, we can now use the standard update equations for μ_k and Σ_k. Specifically,

$$\mu_k = \frac{\sum_{i=1}^{n}\rho_{i,k}X_i}{\sum_{i=1}^{n}\rho_{i,k}},$$

the weighted average of the data with weights equal to the posterior probabilities of

cluster membership. The exact equations for the update equations for Σ_k varies based on the parsimonious parameterization chosen (e.g. EEE, cf. Table 2.1). Celeux and Góvert derive the closed and iterative forms for a variety of parameterizations in [13].

3.2 A Derivation of the BIC for the Semi-supervised Model

Assume the data are distributed according to a member of model M described in Section 3.1. In Section 2.2.1.1, we mentioned that the BIC approximates the posterior loglikelihood under model some model M_1: BIC $\approx 2\log(P(D|M_1))$. In this section, we will justify a modified version of this equation for model M specifically. Consider the integrated likelihood:

$$P(D) = \int P(D|\theta, M)P(\theta|M)d\theta, \tag{3.1}$$

where $P(\cdot)$ is a probability, pmf, or pdf where appropriate. Let

$$Q(\theta) = \log\left(P(D|\theta, M)P(\theta|M)\right),$$

the log of the posterior likelihood. Suppose that the posterior mode exists, say $\bar{\theta}$.

A second order Taylor expansion about $\bar{\theta}$ gives

$$Q(\theta) = Q(\bar{\theta}) + (\theta - \bar{\theta})^T \nabla Q(\bar{\theta}) + \frac{1}{2}(\theta - \bar{\theta})^T \nabla^2 Q(\bar{\theta})(\theta - \bar{\theta}) + O(\|(\theta - \bar{\theta})\|_2^3).$$

By the first order optimality necessary conditions, we know $\nabla Q(\bar{\theta}) = 0$. By ignoring the last term, we will approximate $Q(\theta)$ with the truncated Taylor expansion:

$$Q(\theta) \approx Q(\bar{\theta}) + \frac{1}{2}(\theta - \bar{\theta})^T \nabla^2 Q(\bar{\theta})(\theta - \bar{\theta}).$$

Recalling (3.1), we may approximate $P(D|M)$ using a saddle point approximation:

$$\begin{aligned}
P(D|M) &= \int \exp\left(\log(P(D|\theta, M)P(\theta|M))\right) d\theta \\
&= \int \exp(Q(\theta)) d\theta \\
&\approx \int \exp\left(Q(\bar{\theta}) + \frac{1}{2}(\theta - \bar{\theta})^T \nabla^2 Q(\bar{\theta})(\theta - \bar{\theta})\right) d\theta \\
&= \exp\left(Q(\bar{\theta})\right) \int \exp\left(\frac{1}{2}(\theta - \bar{\theta})^T \nabla^2 Q(\bar{\theta})(\theta - \bar{\theta})\right) d\theta \\
&= \exp\left(Q(\bar{\theta})\right) \int \exp\left(-\frac{1}{2}(\theta - \bar{\theta})^T (-\nabla^2 Q(\bar{\theta}))(\theta - \bar{\theta})\right) d\theta.
\end{aligned}$$

Recognize the integral as proportional to the density of a multivariate Guassian with mean $\bar{\theta}$ and covariance $-\nabla^2 Q(\bar{\theta})$. Let $H = -\nabla^2 Q(\bar{\theta})$.

Then, we have

$$P(D|M) \approx \frac{(2\pi)^{\frac{d}{2}} \exp\left(Q(\bar{\theta})\right)}{\det(H)^{\frac{1}{2}}} \tag{3.2}$$

where d is number of free parameters in θ.

Now we will relate H to the Fisher information matrix as defined by

$$I(\theta)_{jk} \equiv \mathbb{E}\left[\frac{\partial}{\partial\theta_j}\log(p(X,\theta))\frac{\partial}{\partial\theta_k}\log(p(X,\theta))\right].$$

Under the conditions[1] , Proposition 3.4.4. in [11] yields

$$I(\theta)_{jk} = -\mathbb{E}_\theta\left[\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log P(X,\theta)\right].$$

Thus, if $P(\theta|M)$ is uninformative, then $H = -\nabla^2 Q(\bar{\theta}) = -\sum_{i=1}^{n}\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log P(X,\theta) \approx$

[1]Conditions for Proposition 3.4.4 in [11]. Assume

1. $\theta \in \Theta$, for open $\Theta \subset \mathbb{R}^d$,
2. $\{P(x,\theta) : \theta \in \Theta\}$ is a regular parametric model,
3. P is twice continuously differentiable,
4. for any statistic T such that $\mathbb{E}_\theta[|T|] < \infty$ for all $\theta \in \Theta$, we have that
$$\frac{\partial}{\partial\theta_j}\left[\int T(x)P(x,\theta)dx\right] = \int T(x)\frac{\partial}{\partial\theta_j}P(x,\theta)dx,$$
5. for any statistic T such that $\mathbb{E}_\theta[|T|] < \infty$ for all $\theta \in \Theta$, we have that
$$\frac{\partial^2}{\partial\theta_j\partial\theta_k}\left[\int T(x)P(x,\theta)dx\right] = \int T(x)\frac{\partial}{\partial\theta_j\partial\theta_k}P(x,\theta)dx,$$
and
6. the support of $P(\cdot,\theta)$ does not depend on θ.

$nI(\theta)$. H is often referred to as the observed Fisher information.

By consistency of the observed Fisher information matrix, if $\hat{\theta}$ is the Maximum Likelihood Estimator (MLE), then $H(\hat{\theta}) = I(\theta_0) + o_p(1)$, where $I(\theta_0)$ is the Fisher information at the true generating parameters. When the posterior mode is nearly or is equal to the MLE, as is the case when the prior on θ is uniform and Θ is finite (cf. Bickel and Doksum pp. 114), substitute $\hat{\theta}$, the MLE, for $\bar{\theta}$, the posterior mode.

Taking $2\log(\cdot)$ of both sides, (3.2) becomes

$$\begin{aligned} 2\log(P(D|M)) &\approx 2Q(\hat{\theta}) + d\log(2\pi) + \log(\det(I(\hat{\theta})^{-1})) \\ &= 2\log(P(D|\hat{\theta}, M)) + 2\log(P(\hat{\theta}|M)) + d\log(2\pi) - \log(\det(I(\hat{\theta})). \end{aligned}$$

Now, we must calculate $-\log(\det(I(\hat{\theta}))$. Define $n \equiv \sum_{i=1}^{C} n_i$. Observe

$$\begin{aligned} I(\theta) &= \text{Var}\left[\frac{\partial}{\partial\theta}\log(p(X,\theta)\right] \\ &= \text{Var}\left[\sum_{i=1}^{n}\frac{\partial}{\partial\theta}\log(p(X_i,\theta))\right] \\ &= \sum_{i=1}^{C} n_i I(\theta_i) \text{ by independence.} \end{aligned}$$

Henceforth, we will use I_j to denote $I(\theta_j)$. We will assume that I_1 is positive definite, so that it can be written as

$$I_1 = S_1^T S_1$$

for some non-singular matrix S_1. Let J denote the d-dimensional identity matrix. For notational purposes, let

$$n^* := \max_{2 \leq j \leq C} (n_j).$$

Observe

$$\begin{aligned}
\det\left(\sum_{i=1}^{C} n_i I(\theta_i)\right) &= \det\left(S_1^T (n_1 J + \left(S_1^T\right)^{-1}\left(\sum_{i=2}^{C} n_i I(\theta_i)\right) S_1^{-1}) S_1\right) \\
&= \det(I_1)\det\left(n_1 J + n^* \left(S_1^T\right)^{-1}\left(\sum_{i=2}^{C} \frac{n_i}{n^*} I(\theta_i)\right) S_1^{-1}\right) \\
&= \det(I_1) n_1^d \det\left(J + \frac{n^*}{n_1}\left(S_1^T\right)^{-1}\left(\sum_{i=2}^{C} \frac{n_i}{n^*} I(\theta_i)\right) S_1^{-1}\right) \\
&= \det(I_1) n_1^d \det\left(J + \frac{n^*}{n_1} B\right),
\end{aligned}$$

where $B = \left(S_1^T\right)^{-1} (\sum_{i=2}^{C} \frac{n_i}{n^*} I(\theta_i)) S_1^{-1}$. A matrix C is a $*-$transform of another matrix D if there exists a nonsingular matrix S such that $C = S^* D S$ (i.e. C is congruent to D). Note that since S_1 is nonsingular, B is a $*-$transform of $(\sum_{i=2}^{C} \frac{n_i}{n^*} I(\theta_i))$. Then, by Sylvester's Law of Inertia, B has the same inertia as $\sum_{i=2}^{C} \frac{n_i}{n^*} I(\theta_i)$, which is a sum of positive semi-definite matrices. Therefore, B is a positive semi-definite matrix.

Next, we would like to describe the growth of $\det\left(J + \frac{n^*}{n_1} B\right)$. For any eigenvalue of $J + \frac{n^*}{n_1} B$, say λ, we have by Weyl's theorem

$$1 \leq \lambda \leq 1 + \frac{n^*}{n_1} \|B\|_2.$$

Observe

$$\begin{aligned}
\|B\|_2 &= \max_{\{x\in\mathbb{R}^d:\|x\|_2=1\}} x^T B x \\
&\leq \max_{\{x\in\mathbb{R}_1^d:\|x\|_2=1\}} \|S_1^{-1}\|_2^2 x^T \left(\sum_{j=2}^{C} \frac{n_j}{n^*} I_j\right) x \\
&\leq \|S_1^{-1}\|_2^2 \left(\sum_{j=2}^{C} \frac{n_j}{n^*} \|I_j\|_2^2\right) \\
&\leq \|S_1^{-1}\|_2^2 \left(\sum_{j=2}^{C} \|I_j\|_2^2\right),
\end{aligned}$$

which is independent of n. Let $M_2 \equiv \|S_1^{-1}\|_2^2 \left(\sum_{j=2}^{C} \|I_j\|_2^2\right)$. Let $\sigma(J + \frac{n^*}{n_1}B) = \{\lambda_1, \lambda_2, \ldots, \lambda_d\}$ be the spectrum of $J + \frac{n^*}{n_1}B$, ordered in decreasing magnitude and counting multiplicities. Then, we have

$$\begin{aligned}
\det(J + \frac{n^*}{n_1}B) &= \Pi_{m=1}^{d} \lambda_m \\
&\leq \left(1 + \frac{n^*}{n_1}(M_2)\right)^d \\
&\leq \left(1 + \frac{n - n_1}{n_1}(M_2)\right)^d
\end{aligned}$$

Because d is fixed, the only term growing with n above is $\frac{n-n_1}{n_1}$, the ratio of the supervised data over the unsupervised data. In general, it is usually much more expensive to obtain additional supervised data than unsupervised; thus, we find it reasonable to posit that $\frac{n-n_1}{n_1} \to 0$ in n (i.e. is $o(1)$). In this case, $\log\left(\det(J + \frac{n^*}{n_1}B)\right)$

is $o(1)$ by continuity and our bounds.

Hence,

$$\log\left(\det(\, I(\theta)\,)\right) = \log\left(\det(I_1)\right) + d\log(n_1) + \log\left(\det(J + \frac{n^*}{n_1}B)\right) \quad (3.3)$$

$$= \log\left(\det(I_1)\right) + d\log(n_1) + o(1). \quad (3.4)$$

When the posterior mode is nearly or is equal to the MLE, as is the case when the prior on θ is uniform and Θ is finite (cf. Bickel and Doksum pp. 114), substitute $\hat{\theta}$, the MLE, for $\bar{\theta}$, the posterior mode.

Recall that we had

$$2\log(P(D|M)) \approx 2\log(P(D|\hat{\theta}, M)) + 2\log(P(\hat{\theta}|M)) + d\log(2\pi) - \log(\det(I(\hat{\theta})).$$

We can handle each term in the above approximation:

- $2\log(P(\bar{\theta}|M)) = O(1)$ by flat prior on θ
- $d\log(2\pi) = O(1)$ because d is fixed
- $\log(\det(I(\hat{\theta})) = \log\left(\det(I_1)\right) + d\log(n_1) + o(1)$ by Equation 3.4

Finally, note that I_1 in is the Fisher information matrix for one datum from group 1 evaluated at the MLE. Suppose. $\theta_{1,0}$ were the true generating parameters of the data in group 1. We would like to say that $I_1 \approx I_{\theta_{1,0}}$. We know $\hat{\theta}_1 \to \theta_{1,0}$ in probability by consistency of the MLE. Suppose P is thrice continuously differentiable with bounded

third derivative. Since

$$I_1(\theta_{1,0}) = \mathbb{E}\left[\frac{\partial^2}{\partial\theta^2}\log(P(X,\theta=\theta_{1,0}))\right] = \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial^2}{\partial\theta^2}\log(P(x_i^{(1)},\theta=\theta_{1,0})) + o_P(1),$$

and P is thrice differentiable, the mean value theorem gives that there exists a θ' between $\hat{\theta}_1$ and $\theta_{1,0}$ such that

$$\begin{aligned}\frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial^2}{\partial\theta^2}\log(P(x_i^{(1)},\theta=\theta_{1,0})) - \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial^2}{\partial\theta^2}\log(P(x_i^{(1)},\theta=\hat{\theta}_1)) \\ = \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\partial^3}{\partial\theta^3}\log(P(x_i^{(1)},\theta=\theta'))\left(\hat{\theta}_1-\theta_{1,0}\right).\end{aligned}$$

If the third derivative is bounded in probability, we have that $I_1 = I_1(\theta_{1,0}) + o_P(1)$. Hence, $\log\left(\det(I_1)\right) = O_P(1)$.

Therefore, $2\log(P(D|M)) \approx 2\log(P(D|\hat{\theta},M)) - d\log(n_1) + o_P(1) + O_P(1)$. Note that the terms of order less than $O_P(1)$ get washed out in the limit as $n\to\infty$ in the sense that the other terms are diverging. If we drop them, we have our derivation of the adjusted BIC.

Remark The choice to keep $d\log(n_1)$ around instead of $d\log(n)$ is subtle. Suppose we made the latter choice, noting that

$$\log(n_1) = \log(\frac{n_1}{n}n) = \log(\frac{n_1}{n}) + \log(n) = o(1) + \log(n).$$

Thus an approximation of this type only adds error that we apparently feel com-

fortable dropping. Why do we not discount this error now? Note that the BIC is used in finite sample situations; it will perform differently if the actual value for the penalty term uses n_1 vs. n. The use of n_1 will lead to less parsimonious models. One intuitive reason we would want to penalize less harshly than the BIC does is that the complexity introduced by the largely constrained known datum should be lower than an unconstrained point.

3.3 Discussion of Inclusion of $O(1)$ Terms in Information Criteria

The Bayesian Information Criterion (BIC) is generally denoted as

$$BIC = 2\mathcal{L} - d\log(n),$$

where $\mathcal{L}$ is the maximum of the likelihood, d is the number of parameters estimated, and n is the number of data used to estimate those parameters.

Alternative information criteria generally differ by having different values for the penalty term. For example, the sample size adjusted BIC [12] is defined as

$$sBIC = 2\mathcal{L} - d\log(\frac{n+2}{24}).$$

Because $BIC - sBIC = O(1)$, one may wonder of the efficacy of such a deviation from the BIC. This question is particularly salient when considering that the standard derivation of the BIC as the approximation to the integrated loglikelihood is only $O(1)$ accurate asymptotically (so that other O(1) terms are dropped). Here, we comment on the use of different penalty terms. Suppose that we define an adjusted BIC, say BIC' as a function of $m \in [1, n)$ as $BIC'(m) = 2\mathcal{L} - d\log(m) = BIC - d\log(\frac{m}{n})$.

When does using such an adjustment matter? Consider two models, say M_0 and M_1, in competition for selection. Fix an m. Suppose that

(i) $d_1 > d_0$

(ii) $BIC_0 > BIC_1$ and

(iii) $BIC'_0(m) < BIC'_1(m)$.

In this case, we see that under the original definition of the BIC, we would choose M_1 over M_0, and with the modified definition, we would do the opposite.

In order to analyze this under some assumptions, define two statistical tests:

$$T(\omega) = \mathbb{1}\{BIC_1 > BIC_0\}$$

and

$$T'_m(\omega) = \mathbb{1}\{BIC'_1(m) > BIC'_0(m)\}.$$

When do T and T' differ?

Case 1: $T = 1$ and $T'_m = 0$. For $m \leq n$ this is impossible.

Case 2: $T = 0$ and $T'_m = 1$. We have two subcases.

(a) Suppose M_1 is the true model. We would be correct in choosing T'_m over T.

(b) Suppose M_0 is the true model. Note that in this case, we would make a mistake by choosing T'_m over T.

We would like to analyze the probabilities in Case 2. Preferably, (2.a) is much more probable than (2.b). Unfortunately, under (2.a) the distribution of $W_{1,0}$ is only known for certain cases. For example, with local alternatives of the form $\theta_n = \theta_0 + \frac{\Delta}{\sqrt{n}}$, $W_{1,0}(d_1 - d_0)$ is known to have noncentral chi square distribution with $df = d_1 - d_0$ and the appropriate non-centrality parameter. Local alternatives are unrealistic in the applications we have in mind; in model based clustering, we compare models of different dimensions. Therefore local alternatives are generally not applicable in the motivating application. Hence, we will defer our analysis under (2.a) to a specific example.

We can say more about (2.b) with some assumptions.

Proposition 3.3.1 *Suppose the models are nested (so that M_0 is a submodel of M_1). Define $W_{1,0,n} := \frac{2(\mathcal{L}_1 - \mathcal{L}_0)}{(d_1 - d_0)}$. Assume the distribution functions of $W_{1,0,n}$ are uniformly*

Lipschitz continuous in n. Under M_0, $T = 0$ and $T'_m = 1$ with probability

$$Pr_{M_0}\left(\log(m) < W_{1,0,n} < \log(n)\right) \to F(\log(n)) - F(\log(m)), \tag{3.5}$$

where $F(\cdot)$ is the distribution function of an F distribution with $df = (d_1 - d_0, \infty)$.

Proof We can do some algebra on (ii) and (iii) to obtain equivalent condition that

$$(d_1 - d_0)\log(m) < 2(\mathcal{L}_1 - \mathcal{L}_0) < (d_1 - d_0)\log(n).$$

If the models are nested (so that M_0 is a submodel of M_1), then the numerator of $W_{1,0,n}$ converges in law to a χ^2 distribution with $df = d_1 - d_0$ by Wilk's Theorem. In this case, we note that by Slutsky's Theorem, $W_{1,0,n}$ asymptotically has the distribution of an F-statistic with $df = (d_1 - d_0, \infty)$. By uniform Lipschitz continuity, the probability that (ii) and (iii) hold when they should **not** (i.e. choose M_1 when M_0 is the truth) is given by equation (3.5).

3.3.1 Illustrative Example

We now present an analysis for specific null and alternative models where we can explicitly calculate the probabilities of cases (2.a) and (2.b). The purpose is to demonstrate that for finite n, the choice of penalty term is nontrivial.

Let $d, d_0 \in \mathbb{N}$ with $d_0 < d$. Consider the nested models:

- $M_1 = \{N(\mu, I) : \mu \in \mathbb{R}^d\}$
- $M_{0,d_0} = \{N(\mu'_{d_0}, I) : \mu' \in \mathbb{R}^d, \mu'_j = 0 \text{ for } j = d_0 + 1, d_0 + 2, \ldots, d\}$,

where I is the $d-$dimensional identity matrix.

Let

$$\mu^* = \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \cdots & \frac{1}{\sqrt{d}} \end{bmatrix}^T$$
$$\mu^*_{0,d_0} = \begin{bmatrix} 1 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \cdots & \frac{1}{\sqrt{d_0}} & 0 & 0 & \cdots & 0 \end{bmatrix}^T.$$

Then, $f_1 := N(\mu^*, I) \in M_1$ and $f_{0,d_0} := N(\mu^*_{0,d_0}, I) \in M_0$. Suppose that $X_1, X_2, \ldots, X_n \overset{i.i.d.}{\sim}$ f_1 or f_{0,d_0} according to whether or not M_1 is the true model.

Fix a realization of the data $(x_1, x_2, \ldots, x_n)$. Let $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ and $\bar{x}_{0,d_0}$ be $\bar{x}$ with the coordinates after d_0 set to 0. Twice the maximized loglikelihood of the data under M_1 is (up to constants)

$$2\mathcal{L}_1 := -\sum_{i=1}^n (x_i - \bar{x})^T (x_i - \bar{x}).$$

Under M_{0,d_0} it is

$$\begin{aligned} 2\mathcal{L}_{0,d_0} &:= -\sum_{i=1}^{n}(x_i - \bar{x}_{0,d_0})^T(x_i - \bar{x}_{0,d_0}) \\ &= -\sum_{i=1}^{n}\|x_i - \bar{x} + \bar{x} - \bar{x}_{0,d_0}\|_2^2 \\ &= \mathcal{L}_1 - 2\sum_{i=1}^{n}(x_i - \bar{x})^T(\bar{x} - \bar{x}_{0,d_0}) - n\|\bar{x} - \bar{x}_{0,d_0}\|_2^2 \\ &= \mathcal{L}_1 - n\|\bar{x} - \bar{x}_{0,d_0}\|_2^2, \end{aligned}$$

so that $\mathcal{L}_{0,d_0} < \mathcal{L}_1$.

(2.a) Under f_1,

$$\begin{aligned} n\|\bar{x} - \bar{x}_{0,d_0}\|_2^2 &= \sum_{j=d_0+1}^{d}(\sqrt{n}\bar{x}_j)^2 \\ &= Y_1, \end{aligned}$$

where $Y_1 \sim$ noncentral $\chi^2_{df=(d-d_0)}(n\sum_{j=d_0+1}^{d}\frac{1}{j})$.

Hence, for any $m > 0$,

$$Pr_{f_1}\left(2\mathcal{L}_1 - d\log(m) < 2\mathcal{L}_{0,d_0} - d_0\log(m)\right) = Pr\left(Y_1 < (d-d_0)\log(m)\right).$$

Therefore, $Pr_{f_1}(T = 0 \text{ and } T'_m = 1) = Pr\left((d-d_0)\log(m) < Y_1 < (d-d_0)\log(n)\right)$.

(2.b) Under f_{0,d_0},

$$n\|\bar{x} - \bar{x}_{0,d_0}\|_2^2 = Y_{d_0}, \tag{3.6}$$

where $Y_{d_0} \sim \chi^2_{df=(d-d_0)}$

Thus,

$$Pr_{f_{0,d_0}}\left(2\mathcal{L}_1 - d\log(m) < 2\mathcal{L}_{0,d_0} - d_0\log(m)\right) = Pr\left(Y_{d_0} < (d-d_0)\log(m)\right).$$

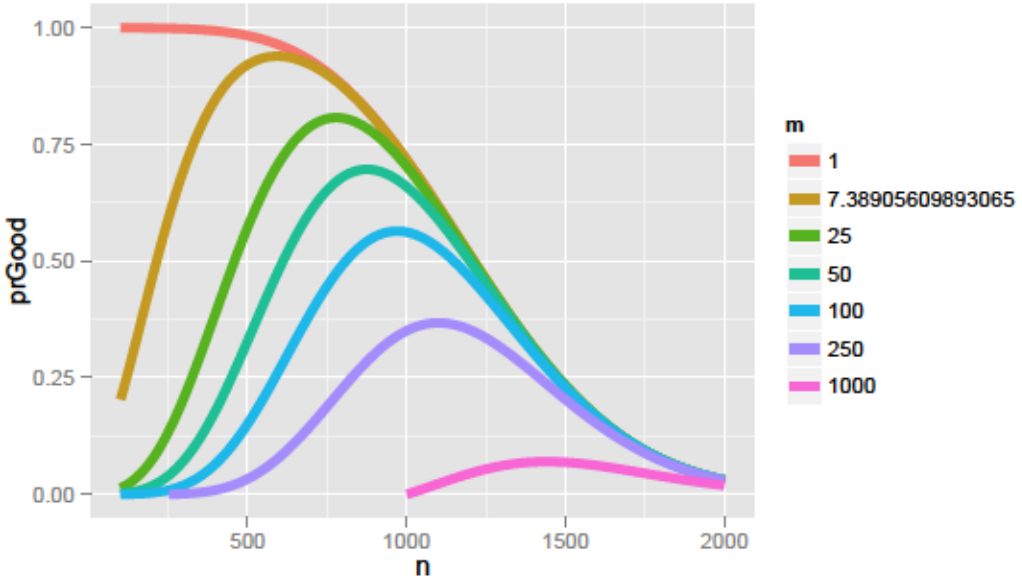
Hence, $Pr_{f_{0,d_0}}(T = 0 \text{ and } T'_m = 1) = Pr\left((d-d_0)\log(m) < Y_{d_0} < (d-d_0)\log(n)\right)$.

Clearly, we can vary n, m, d, and d_0 to influence these probabilities. We will now do so to show how (a) some penalty is better than none; (b) the AIC penalty can result in mistakes; and (c) the optimal penalty depends on n, m, d, and d_0.

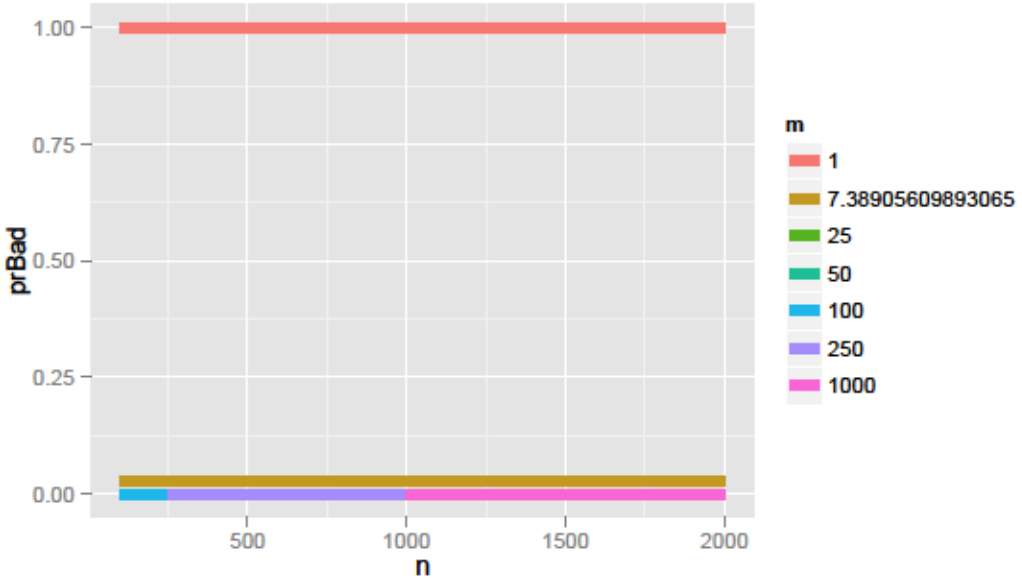
Figure 3.1 depicts the probability of (2.a) and (2.b) when n and m vary for fixed $d = 200$ and $d_0 = 190$ under the alternative (a) and null (b) hypotheses. Lower penalties than the BIC would use generally result in better decisions under f_1. However, there is approximately a 3% chance of error under f_0 when the BIC would be correct with the penalty of exp(2), that of the AIC.

Figure 3.2 depicts the probability of (2.a) and (2.b) when d and m vary for fixed $n = 1000$ and $d_0 = d - 10$ under the alternative (a) and null (b) hypotheses. Lower penalties than the BIC would use generally result in better decisions under f_1 without sacrificing making too many new mistakes under f_0.

Figure 3.3 depicts the probability of (2.a) and (2.b) when d_0 and m vary for fixed

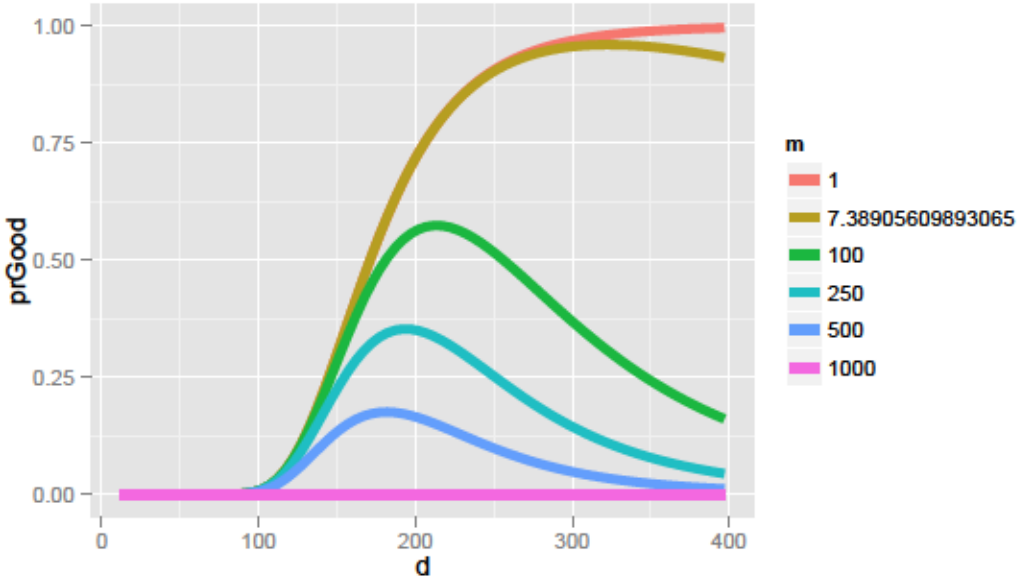


(a) $Pr_{f_1}(T = 0 \text{ and } T'_m = 1)$

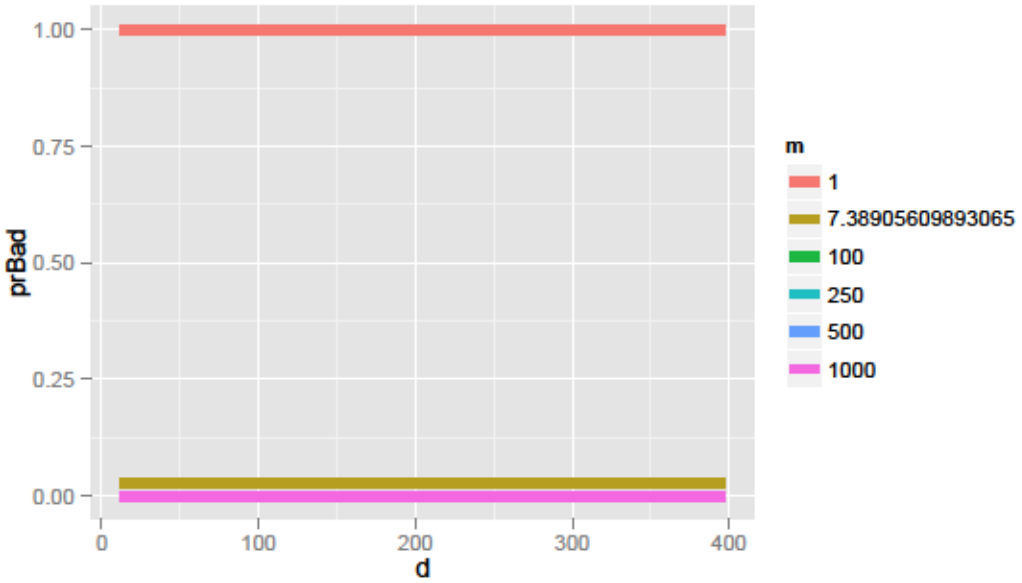


(b) $Pr_{f_{0,d_0}}(T = 0 \text{ and } T'_m = 1)$

Figure 3.1: Probabilities of correctly (a) or mistakingly (b) choosing the alternative over the null when the BIC would not do so. For all curves, $d = 200$ and $d_0 = 190$. Note that 7.389... = exp(2), the AIC penalty.



(a) $Pr_{f_1}(T = 0 \text{ and } T'_m = 1)$



(b) $Pr_{f_{0,d_0}}(T = 0 \text{ and } T'_m = 1)$

Figure 3.2: Probabilities of correctly (a) or mistakingly (b) choosing the alternative over the null when the BIC would not do so. For all curves, $n = 1000$ and $d_0 = d - 10$.

$n = 1000$ and $d = 200$ under the alternative (a) and null (b) hypotheses. Lower penalties than the BIC would use generally result in better decisions under f_1 and not too much worse decisions under f_0. However, there is approximately a larger chance of error under f_0 when the BIC would be correct with the AIC penalty for smaller d_0.

This example may seem overly simplistic. Indeed, it does not involve semi-supervised clustering at all. However, it does demonstrate the complexities of penalization in model selection with finite samples.

3.3.2 Simulation Study

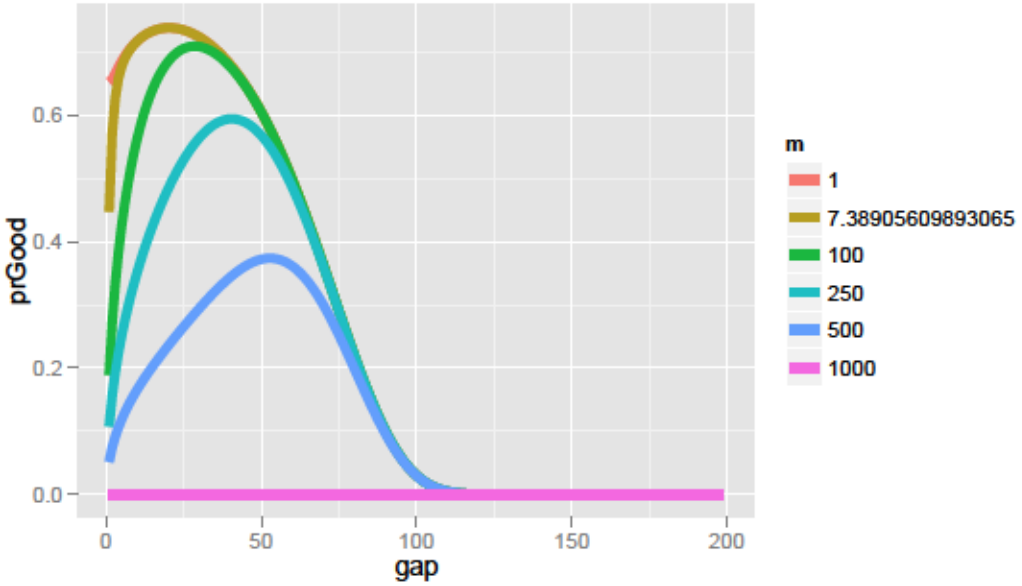
We would like to demonstrate the impact of the previously discussed model selection complexities from the illustrative example on the inference task of semi-supervised clustering. To that end, consider the following procedure for constructing a dataset $\mathcal{X} \subset \mathbb{R}^2$, semi-supervising it, and clustering it using model-based clustering with different penalties. We will then compare the resulting ARI scores.

Let

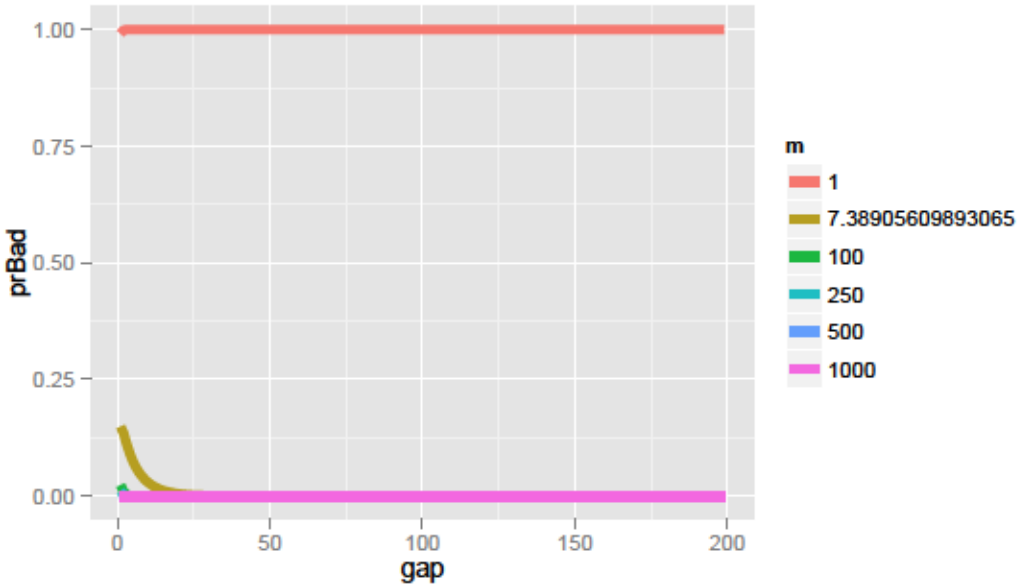
$$\mu_1 = \mu_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \text{ and } \mu_3 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}.$$

Also, let

$$\Sigma_1 = \begin{bmatrix} .5 & .35 \\ .35 & .5 \end{bmatrix}$$



(a) $Pr_{f_1}(T = 0 \text{ and } T'_m = 1)$



(b) $Pr_{f_{0,d_0}}(T = 0 \text{ and } T'_m = 1)$

Figure 3.3: Probabilities of correctly (a) or mistakingly (b) choosing the alternative over the null when the BIC would not do so. For all curves, $n = 1000$ and $d = 200$.

and

$$\Sigma_2 = \begin{bmatrix} .5 & -.35 \\ -.35 & .5 \end{bmatrix}.$$

Consider the following procedure for generating one Monte Carlo Replicate.

1. Define the pdf of each datum as being from a mixture model $f(X) = \sum_{k=1}^{3} \pi_k f_k(X)$, where $f_k = N(\mu_k, \Sigma_k)$ with $6\Sigma_3$ generated using the onion method of [27]

2. Draw $n^S = 100$ supervised from the first two components.

3. Draw $n^U = 5, 10, 20, 40, 80, \ldots, 640$ unsupervised from the full mixture model.

4. Cluster using $2 - 5$ Gaussians with various constraints on the proposed covariance matrices.

5. Choose the models for clustering based on different values of m in the penalty term $d\log(m)$, where $m \in [n^U, n^U + n^S]$.

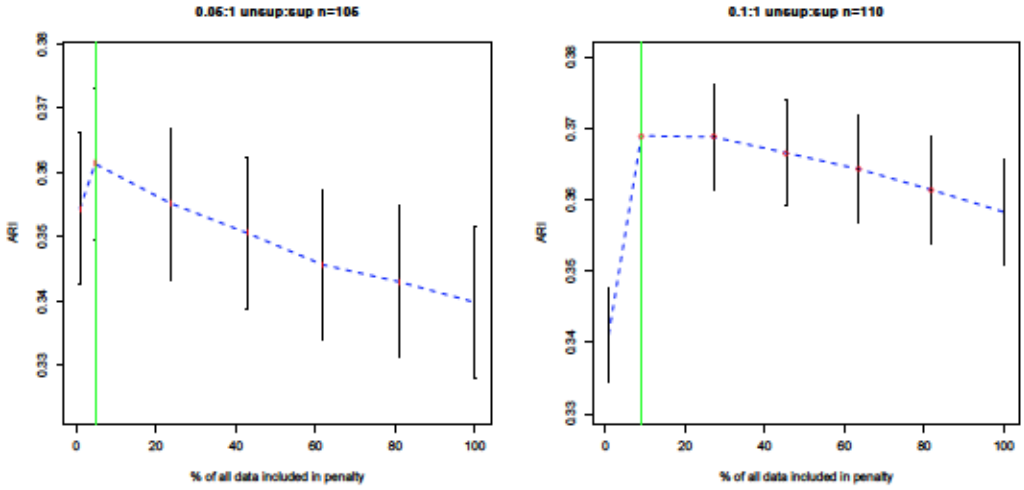
6. Calculate resulting ARIs using the chosen models..

Figures 3.4 and 3.5 shows how different penalties can affect the overall clustering performance on the unlabeled data. Generally, we see that with more unsupervised data, there are less differences between the different choices of penalty terms in the interval $[n^U, n^U + n^S]$, which makes sense given the asymptotic results. In this particular example, less penalization allowed us to detect the third component sooner, improving the ARI values in the resulting clustering. Thus, we have shown that in an

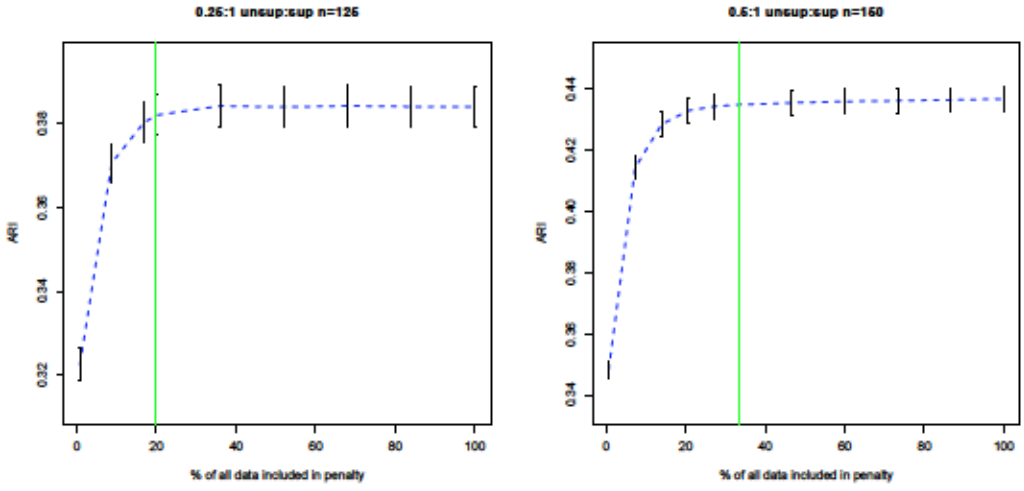
example closer to our problem of interest, the restricted penalization derived in the previous section can be efficacious as compared to the standard penalty. Additionally, as n grows, the differences between the information criteria that are dependent on n to the same order becomes negligible even for relatively small values of n.

3.4 Conclusions

In this chapter, we consider some aspects of model-based semi-supervised clustering. First, we define a model for the semi-supervised case. Next, we derive a BIC applicable for it. Finally, we explicate some aspects of $O(1)$ differences in information criterion (like the BIC) to justify the modifications we make in our derivation. To assist us in this explanation, we explicitly calculate the probabilities of a benefit or detriment in using a modified BIC in an illustrative example. Finally, we present a simulation study in the context of semi-supervised clustering.



(a) $n = 105$ (b) $n = 110$



(c) $n = 125$ (d) $n = 150$

Figure 3.4: ARI vs penalty in $BIC'(m)$. Higher is better. Red circles represent significant paired Wilcoxon tests for larger ARI values than vs the standard BIC (at the .05 level). The area to the right of the green line represents the interval $[n^U, n^U + n^S]$.

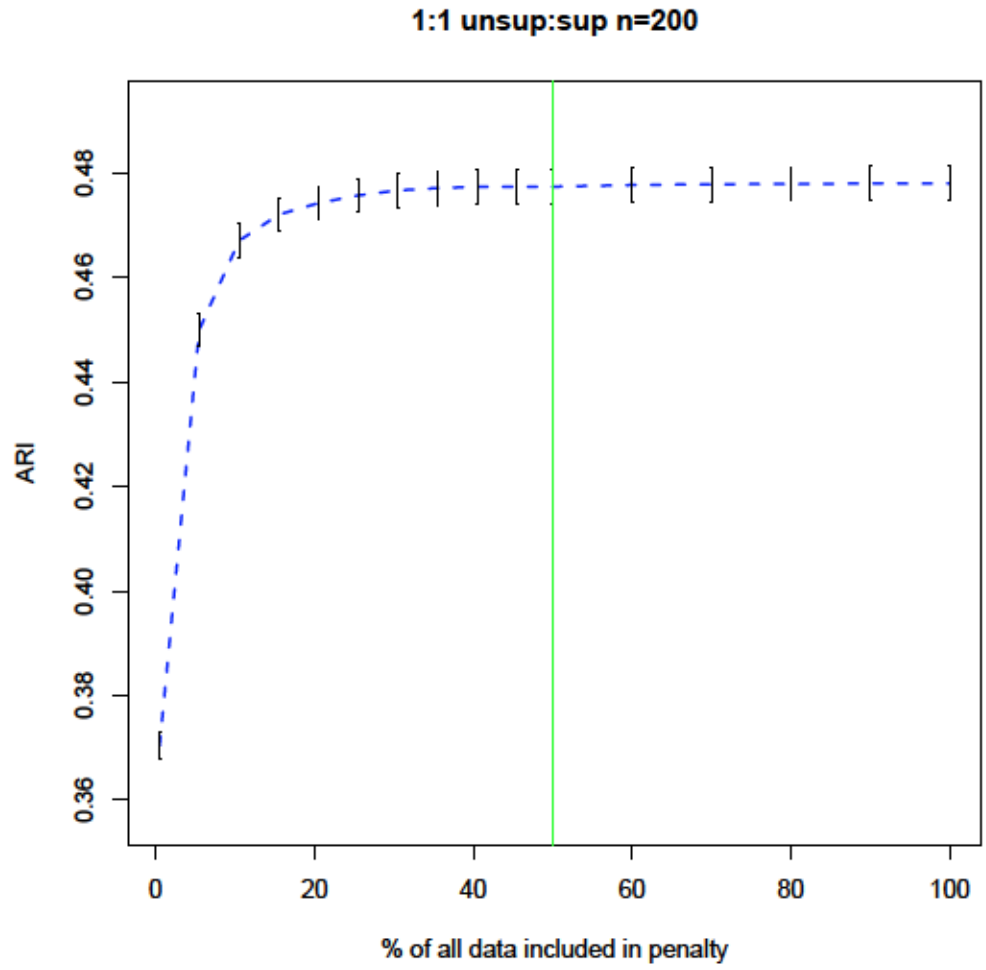


Figure 3.5: ARI vs penalty in $BIC'(m)$ for largest value of the total dataset simulated. Higher is better. Red circles represent significant paired Wilcoxon tests for larger ARI values than vs the standard BIC (at the .05 level). The area to the right of the green line represents the interval $[n^U, n^U + n^S]$.

Chapter 4

Semi-supervised K-means++

It is better to be right than to be wrong.

Andrey Kolmogorov, as attributed by Theodore Drivas.

Most of this chapter appears in [59].

The EM algorithm for clustering requires initialization with either initial parameters or initial (possibly soft) cluster assignments. Because it is generally unrealistic to obtain adequate initial parameters, initial cluster assignments are typically used. Two main strategies are employed to this end:

(1) randomly partition the data, run the EM algorithm, save the likelihood, and repeat, saving the results corresponding to the highest likelihood; or

(2) use another algorithm (e.g. hierarchical clustering) that is easier to initialize in a principled fashion.

Due to the modifications necessary for semi-supervised hierarchical clustering (namely, moving towards must-link-before style constraints), we would prefer not to use that for the second strategy of initializing the EM algorithm. Thus, we present another semi-supervised clustering algorithm in this chapter.

In this chapter, we propose a semi-supervised version of `k-means++`. We first introduce the definitions and notation to be used afterwards in the remainder of the paper. Next, we present the main algorithm, where we modify the `k-means++` algorithm for the semi-supervised with labels case. We then prove an approximation bound that improves with the amount of supervision. Finally, we include numerical experiments showing the efficacy of the algorithm on simulated and real data under a few performance metrics.

4.1 Preliminaries

We will present a few definitions to clarify the notation used in the theoretical results. Recall that $\mathcal{X} \subset \mathbb{R}^d$ is our data. We will additionally partition $\mathcal{X} = \mathcal{X}^{(u)} \cup \mathcal{X}^{(s)}$ into the unsupervised and supervised data, respectively.

Clustering A clustering, say $\mathcal{C}$, is a set of centers that are used for cluster assignments of unlabeled points.

Potential function Fix a clustering $\mathcal{C}$. Define the potential function as $\phi_{\mathcal{C}} : 2^{\mathcal{X}} \to \mathbb{R}^+$ such that for $A \subset \mathcal{X}$,

$$\phi_{\mathcal{C}}(A) = \phi(A; \mathcal{C}) = \sum_{a \in A} \min_{c \in \mathcal{C}} \|a - c\|.$$

Optimal cluster Let $\mathcal{C}^*$ be a clustering solving the k-means problem from equation (2.1). Let $c^* \in \mathcal{C}^*$. An optimal cluster $\mathcal{X}_{c^*}$ is defined such that

$$\mathcal{X}_{c^*} = \{x \in \mathcal{X} : c^* = \text{argmin}_{c \in \mathcal{C}^*} \|x - c\|^2\}$$

Distance squared weighting function Call the current clustering $\mathcal{C}$. Define $D^2 : \mathbb{R}^d \to \mathbb{R}^+$ such that

$$D^2(x) = \phi(\{x\}; \mathcal{C}).$$

Covered cluster Call the current clustering $\mathcal{C}$. An optimal cluster $\mathcal{X}_{c^*}$ is covered if

$$\mathcal{C} \cap \mathcal{X}_{c^*} \neq \emptyset.$$

4.2 Semi-supervised K-means++ Algorithm

We now propose an extension to the `k-means++` algorithm for the semi-supervised case. We will called this the `ss-k-means++` algorithm. Suppose that we want to partition our data $\mathcal{X} \subset \mathbb{R}^d$ into G groups. Assume that the semi-supervision occurs

in the following way:

(1) choose a class c_i uniformly at random;

(2) choose g_i observations uniformly at random from $\mathcal{X}_{c_i}$

(3) and label these g_i observations as being from class i.

We optionally allow repetition of steps $1-3$ to give more partially supervised classes. The modified k-means algorithm, which is Algorithm 4 followed by Algorithm 2, replaces the initial step of choosing a point at random by choosing g_i points as above, then setting the first center to the centroid of those points. Also, during the D^2 probabilistic selection process, we do not allow centers to be chosen from the supervised points. Note that this is essentially the `k-means++` version of the `Constrained-KMeans` algorithm [9].

4.3 Theoretical Results

Consider the objective function $\phi(\mathcal{X}; C) = \sum_{x \in \mathcal{X}} \min_{c \in C} \|x - c\|^2$, the potential function associated with a clustering C. Arthur and Vassilvitskii [5] prove that the expectation of the potential function after the seeding phase of the `k-means++` algorithm satisfies

$$\mathbb{E}[\phi(\mathcal{X})] \leq 8(\log(k) + 2)\phi_{OPT},$$

Algorithm 4: Initialization of centers for semi-supervised k-means++

Input: $\mathcal{X}^{(u)}$ (n_u unlabeled datapoints)
$\mathcal{X}^{(s)}$ (n_s labeled datapoints)
$L \in \{1, 2, 3, \ldots, k\}^{n_s}$ (labels corresponding to the data in $\mathcal{X}^{(s)}$
k (number of centers)
Output: $\mathcal{C}$ (set of initial centers)

```
Let n_ℓ be the number of supervised datapoints with label ℓ.
Let C = ∅.
for ℓ = 1, 2, ... k do
    if n_ℓ > 0 then
        Let c_ℓ be the centroid of the labeled datapoints with label ℓ.
        Update C = C ∪ {c_ℓ}.
while card(C) < k do
    Choose a datapoint x ∈ X^(u) with probability proportional to D^2(x).
    Update C = C ∪ {x}.
return C
```

where ϕ_{OPT} corresponds to the potential using the optimal G centers. We will improve this bound for our algorithm by mostly following their analysis, *mutatis mutandis*.

The sketch of the proof is as follows:

1. Bound the expectation of the potential for the first cluster (chosen by semi-supervision)

2. Bound the expectation of the potential for clusters with centers chosen via D^2 weighting conditioned on the center being from an uncovered cluster.

3. Bound the expectation of the potential when choosing a number of new centers at once in a technical result

4. Specialize the technical result to our algorithm and get the overall bound

Consider a collection of data A of size n. Suppose we have g uniformly chosen at random members of A in a set $S \subset A$ that we consider pre-labeled. Consider the mean of these datum, say $\bar{a}_S$, to be the proposed center of A, then the expectation of the potential function is

$$\mathbb{E}[\phi(A; \bar{a}_S)] = \mathbb{E}\left[\sum_{a \in A} \|a - \bar{a}_S\|^2\right],$$

where the expectation is over the choice of the elements of S. We can compute this expectation explicitly. We will first need some lemmas.

Lemma 4.3.1 *Let $A \subset \mathbb{R}^d$ and c be the centroid of A. Let $n := card(A)$. For any $z \in \mathbb{R}^d$, $\sum_{a \in A} \|a - z\|^2 = \sum_{a \in A} \|a - c\|^2 + n\|z - c\|^2$*

Proof Observe

$$\begin{aligned}
\sum_{a \in A} \|a - c\|^2 &= \sum_{a \in A} \|(a - z) + (z - c)\|^2 \\
&= \sum_{a \in A} \left(\|a - z\|^2 + 2(a - z)^T(z - c)\right) + n\|z - c\|^2 \\
&= \sum_{a \in A} \|a - z\|^2 + 2n(c - z)^T(z - c) + n\|z - c\|^2 \\
&= \sum_{a \in A} \|a - z\|^2 - n\|z - c\|^2.
\end{aligned}$$

Hence,

$$\sum_{a \in A} \|a - z\|^2 = \sum_{a \in A} \|a - c\|^2 + n\|z - c\|^2,$$

which was what was wanted. ∎

Lemma 4.3.2 *If $S \subset A$ is a subset of $A = \{x_i \in \mathbb{R}^d : i = 1, 2, \ldots n\}$ of size g chosen uniformly at random from all subsets of A of size g, then*

$$\mathbb{E}[\sum_{x_i \in S} x_i] = \frac{g}{n} \sum_{x_i \in A} x_i.$$

Proof Let X_i be the indicator random variable that is 1 if $x_i \in S$ and 0 otherwise (i.e. $X_i = \mathbb{1}_{\{x_i \in S\}}$). Observe

$$\begin{aligned} \mathbb{E}[\sum_{x_i \in S} x_i] &= \mathbb{E}[\sum_{i=1}^{n} x_i X_i | \sum_{i=1}^{n} X_i = g] \\ &= \sum_{i=1}^{n} x_i \mathbb{E}[X_i | \sum_{i=1}^{n} X_i = g] \end{aligned}$$

Observe $\mathbb{E}[X_i | \sum_{i=1}^{n} X_i = g] = \frac{\binom{n-1}{g-1}}{\binom{n}{g}}$, the probability that x_i is chosen for a group of size g from n objects in A. The conclusion follows. ∎

Lemma 4.3.3 *If $S \subset A$ is a subset of $A = \{x_i \in \mathbb{R}^d : i = 1, 2, \ldots n\}$ of size g chosen uniformly at random from all subsets of A of size g, then*

$$\mathbb{E}\left[\left(\sum_{x_i \in S} x_i\right)^T \left(\sum_{x_i \in S} x_i\right)\right] = \frac{g(g-1)}{n(n-1)} \sum_{i \neq j} x_i^T x_j + \frac{g}{n} \sum_{i=1}^{n} x_i^T x_i.$$

Proof Let X_i be the indicator random variable that is 1 if $x_i \in S$ and 0 otherwise

(i.e. $X_i = \chi_{\{x_i \in S\}}$). Observe

$$\begin{aligned}
\mathbb{E}\left[\left(\sum_{x_i \in S} x_i\right)^T \left(\sum_{x_i \in S} x_i\right)\right] &= \mathbb{E}\left[\left(\sum_{i=1}^{n} x_i X_i\right)^T \left(\sum_{j=1}^{n} x_j X_j\right) | \sum_{\ell=1}^{n} X_\ell = g\right] \\
&= \mathbb{E}\left[\sum_{i,j}^{n} x_i^T x_j X_i X_j \, | \sum_{\ell=1}^{n} X_\ell = g\right] \\
&= \sum_{i,j}^{n} x_i^T x_j \mathbb{E}\left[X_i X_j \, | \sum_{\ell=1}^{n} X_\ell = g\right].
\end{aligned}$$

Observe

$$\mathbb{E}\left[X_i X_j \, | \sum_{\ell=1}^{n} X_\ell = g\right] = \begin{cases} \frac{\binom{n-2}{g-2}}{\binom{n}{g}} & \text{if } i \neq j \\ \frac{\binom{n-1}{g-1}}{\binom{n}{g}} & \text{if } i = j \end{cases},$$

since the first case represents the probability that x_i and x_j are chosen together and the second case represents the probability that x_i is chosen, as $X_i^2 = X_i$. The conclusion follows. ∎

Now we present the promised expectation.

Lemma 4.3.4 *If $S \subset A$ is a subset of $A = \{x_i \in \mathbb{R}^d : i = 1, 2, \ldots n\}$ of size g chosen uniformly at random from all subsets of A of size g, then*

$$\mathbb{E}[\phi(A; \bar{a}_S)] = \left(1 + \frac{n-g}{g(n-1)}\right) \phi_{OPT}(A),$$

where

$$\phi_{OPT}(A) = \sum_{a_i \in A} \|a_i - c(A)\|^2,$$

$c(A)$ is the centroid of A (i.e. $c(A) = \frac{1}{n}\sum_{a\in A} a$), and $\bar{a}_S$ is the centroid of S.

Proof Let $n = card(A)$. Observe

$$\begin{aligned}
\mathbb{E}\left[\phi(A;\bar{a}_S)\right] &= \mathbb{E}\left[\sum_{a\in A}\|a-\bar{a}_S\|^2\right] \\
&= \mathbb{E}\left[\sum_{a\in A}\|a-c(A)\|^2 + n\|\bar{a}_S - c(A)\|^2\right] \text{ by Lemma 4.3.1} \\
&= \sum_{a\in A}\|a-c(A)\|^2 + n\mathbb{E}\left[\|\bar{a}_S - c(A)\|^2\right] \\
&= \phi_{OPT}(A) + n\mathbb{E}\left[\|\bar{a}_S - c(A)\|^2\right].
\end{aligned}$$

Let us determine $\mathbb{E}\left[\|\bar{a}_S - c(A)\|^2\right]$. Observe

$$\begin{aligned}
\mathbb{E}\left[\|\bar{a}_S - c(A)\|^2\right] &= \mathbb{E}\left[\bar{a}_S^T\bar{a}_S\right] - 2c(A)^T\mathbb{E}\left[\bar{a}_S\right] + c(A)^Tc(A) \\
&= \mathbb{E}\left[\bar{a}_S^T\bar{a}_S\right] - 2g^{-1}c(A)^T\mathbb{E}\left[\left(\sum_{a\in S}^{n} a\right)\right] + c(A)^Tc(A) \\
&= \mathbb{E}\left[\bar{a}_S^T\bar{a}_S\right] - 2g^{-1}c(A)^T\frac{g}{n}\sum_{i=1}^{n} a_i + c(A)^Tc(A) \text{ by Lemma 4.3.2} \\
&= \mathbb{E}\left[\bar{a}_S^T\bar{a}_S\right] - 2c(A)^Tc(A) + c(A)^Tc(A) \\
&= g^{-2}\mathbb{E}\left[\left(\sum_{a\in S}^{n} a\right)^T\left(\sum_{a\in S}^{n} a\right)\right] - c(A)^Tc(A).
\end{aligned}$$

Applying Lemma 4.3.3, we have

$$\begin{aligned}
\mathbb{E}\left[\|\bar{a}_S - c(A)\|^2\right] &= g^{-2}\left(\frac{g(g-1)}{n(n-1)}\sum_{i\neq j} a_i^T a_j + \frac{g}{n}\sum_{i=1}^{n} a_i^T a_i\right) - c(A)^T c(A) \\
&= \frac{1}{gn}\left(\frac{g-1}{n-1}\sum_{i,j} a_i^T a_j + \left(1 - \frac{g-1}{n-1}\right)\sum_{i=1}^{n} a_i^T a_i\right) - c(A)^T c(A) \\
&= \frac{1}{gn}\left(\frac{g-1}{n-1}n^2 c(A)^T c(A) + \left(1 - \frac{g-1}{n-1}\right)\sum_{i=1}^{n} a_i^T a_i\right) - c(A)^T c(A) \\
&= \left(\frac{g-1}{g}\frac{n}{n-1} - 1\right)c(A)^T c(A) + \frac{1}{gn}\left(1 - \frac{g-1}{n-1}\right)\sum_{i=1}^{n} a_i^T a_i \\
&= -\frac{n-g}{g(n-1)}c(A)^T c(A) + \frac{n-g}{gn(n-1)}\sum_{i=1}^{n} a_i^T a_i \\
&= \frac{n-g}{g(n-1)}\left(\frac{1}{n}\sum_{i=1}^{n} a_i^T a_i - c(A)^T c(A)\right) \\
&= \frac{n-g}{gn(n-1)}\left(\sum_{i=1}^{n} a_i^T a_i - nc(A)^T c(A)\right) \\
&= \frac{n-g}{gn(n-1)}\phi_{OPT}(A).
\end{aligned}$$

Hence, $\mathbb{E}\left[\phi(A;\bar{a}_S)\right] = \left(1 + \frac{n-g}{g(n-1)}\right)\phi_{OPT}(A)$. ∎

Lemma 4.3.4 will handle the semi-supervised clusters. Suppose that C_{OPT} is the optimal set of cluster centers. Now, we consider the contribution to the potential function of an optimal cluster $\mathcal{X}^* := \{x \in \mathcal{X} : c^* = argmin_{c\in C_{OPT}}\|x - c\|^2\}$ from C_{OPT} when a center is chosen from C_{OPT} with D^2 weighting. If we can prove a good approximation bound, then we can say that conditioned on choosing centers from uncovered clusters, we will have a good result on average.

Lemma 4.3.5 *Let C be the current (arbitrary) set of cluster centers. Let $\mathcal{X}^*$ be any cluster in C_{OPT}. Let x^* be a point from $\mathcal{X}^*$ chosen at random with D^2 weighting. Then,*

$$\mathbb{E}\left[\phi(\mathcal{X}^*; C \cup \{x^*\}) | x^* \in \mathcal{X}^*\right] \leq 8\phi(\mathcal{X}^*; C_{OPT}),$$

where the expectation is over the choice of new center x^*.

Proof This is essentially the proof from Lemma 3.2 [5]. We will present our own version for completeness, however.

Conditioned on the fact that the new cluster center will be chosen from $\mathcal{X}^*$, any given point $x \in \mathcal{X}^*$ is chosen as the new center with probability

$$\frac{D^2(x)}{\sum_{x' \in \mathcal{X}^*} D^2(x')} = \frac{\min_{c \in C} \|x - c\|^2}{\sum_{x' \in \mathcal{X}^*} \min_{c \in C} \|x' - c\|^2}.$$

Then,

$$\begin{aligned}
\mathbb{E}\Big[\phi(\mathcal{X}^*; C \cup \{x^*\}) | x^* \in \mathcal{X}^*\Big] &= \sum_{x \in \mathcal{X}^*} P\left(x \text{ is chosen from } \mathcal{X}^*\right) \phi(\mathcal{X}^*; C \cup \{x\}) \\
&= \sum_{x \in \mathcal{X}^*} \frac{\min_{c \in C} \|x - c\|^2}{\sum_{x' \in \mathcal{X}^*} \min_{c \in C} \|x' - c\|^2} \sum_{x'' \in \mathcal{X}^*} \left(\min\left(D^2(x''), \|x'' - x\|^2\right)\right).
\end{aligned}$$

Recall that $\|x - c\| \leq \|x'' - x\| + \|x'' - c\|$ for each center $c \in C$ by the triangle inequality. Taking the min of both sides over c, we have

$$\min_{c \in C} \|x - c\| \leq \|x'' - x\| + \min_{c \in C} \|x'' - c\|,$$

implying

$$D(x) \leq \|x'' - x\| + D(x'').$$

Squaring both sides, we have that

$$D^2(x) \leq \left(\|x'' - x\| + D(x'')\right)^2.$$

The power-mean inequality implies

$$\left(\frac{\|x'' - x\| + D(x'')}{2}\right)^2 \leq \frac{\|x'' - x\|^2 + D^2(x'')}{2},$$

and hence

$$\left(\|x'' - x\| + D(x'')\right)^2 \leq 2\|x'' - x\|^2 + 2D^2(x'').$$

Therefore, we know that

$$D^2(x) \leq 2\|x'' - x\|^2 + 2D^2(x'').$$

Summing over all $x'' \in \mathcal{X}^*$, we have

$$D^2(x) \leq \frac{2}{card(\mathcal{X}^*)} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2 + \frac{2}{card(\mathcal{X}^*)} \sum_{x'' \in \mathcal{X}^*} D^2(x'').$$

Applying this inequality, we have

$$
\begin{aligned}
&\mathbb{E}\left[\phi(\mathcal{X}^*; C \cup \{x^*\}) | x^* \in \mathcal{X}^*\right] \\
&\leq \sum_{x \in \mathcal{X}^*} \frac{\frac{2}{card(\mathcal{X}^*)} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2}{\sum_{x' \in \mathcal{X}^*} D(x')^2} \sum_{x'' \in \mathcal{X}^*} \left(\min\left(D^2(x''), \|x'' - x\|^2\right)\right) \\
&\quad + \frac{2}{card(\mathcal{X}^*)} \sum_{x \in \mathcal{X}^*} \sum_{x'' \in \mathcal{X}^*} \left(\min\left(D^2(x''), \|x'' - x\|^2\right)\right) \\
&\leq \sum_{x \in \mathcal{X}^*} \frac{\frac{2}{card(\mathcal{X}^*)} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2}{\sum_{x' \in \mathcal{X}^*} D(x')^2} \sum_{x'' \in \mathcal{X}^*} D^2(x'') + \frac{2}{card(\mathcal{X}^*)} \sum_{x \in \mathcal{X}^*} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2 \\
&= \frac{4}{card(\mathcal{X}^*)} \sum_{x \in \mathcal{X}^*} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2.
\end{aligned}
$$

Since

$$
\frac{1}{card(\mathcal{X}^*)} \sum_{x \in \mathcal{X}^*} \sum_{x'' \in \mathcal{X}^*} \|x'' - x\|^2 = \mathbb{E}\left[\phi(\mathcal{X}^*; x)\right],
$$

the expectation of the potential function of a single cluster if a center is chosen uniformly at random from its points, we can apply Lemma 4.3.4 with $g = 1$ and $A = \mathcal{X}^*$. Hence, $\mathbb{E}\left[\phi(\mathcal{X}^*; C \cup \{x^*\}) | x^* \in \mathcal{X}^*\right] \leq 8\phi(\mathcal{X}^*; C_{OPT})$. ∎

Before introducing the main technical workhorse, note that $\phi(Q_1 \cup Q_2) = \phi(Q_1) + \phi(Q_2)$ for any sets Q_1 and Q_2 such that $Q_1 \cap Q_2 = \emptyset$. Also, $\phi(Q_1 \setminus Q_2) = \phi(Q_1) - \phi(Q_2)$ for any sets Q_1 and Q_2 such that $Q_2 \subseteq Q_1$.

Lemma 4.3.6 *Fix a clustering C. Suppose there are $u \in \mathbb{N}$ uncovered clusters from the optimal clustering C_{OPT}. Denote the points in these uncovered clusters as $\mathcal{X}_u$ (not*

to be confused with $\mathcal{X}^{(u)}$*). Let*

$$\mathcal{X}_c = \mathcal{X} \setminus \mathcal{X}_u$$

be the set of points in covered clusters. Use D^2 *weighting (excluding supervised data) to add* $t \leq u$ *new centers to* C *to form* C'*. In a slight abuse of notation, let* $\phi(\cdot) = \phi(\cdot; C)$*,* $\phi'(\cdot) = \phi(\cdot; C')$*, and* $\phi_{OPT} = \phi(\cdot; C_{OPT})$*. Then,*

$$\mathbb{E}[\phi'(\mathcal{X})] \leq (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))\,(1 + H_t) + \frac{u-t}{u}\phi(\mathcal{X}_u),$$

where $H_t = \sum_{j=1}^{t} \frac{1}{j}$ *is the partial sums of the harmonic series. We will define* $H_0 = 0$*.*

Proof We have that the probability of choosing a point from a fixed set A with D^2 weighting ignoring supervised points is

$$Pr(\text{ new center } c \in A) = \frac{\phi(A \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}.$$

Further, note that $\mathcal{X}^{(s)} \cap \mathcal{X}_u = \emptyset$, since all supervised clusters are covered.

Following the argument in [5] using the above probabilities, we will have our result. First, we plan on using induction over both t and u by proving that if the result is true for $(t-1, u)$ and $(t-1, u-1)$, it is true for (t, u). For most of the base cases, we show that the result holds for $(t = 0, u)$ for all $u \in \mathbb{N}$. Clearly, if $t = 0$, $\phi(\mathcal{X}) = \phi(\mathcal{X}_c) + \phi(\mathcal{X}_u) \leq (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))\,(1 + H_0) + \frac{u-0}{u}\phi(\mathcal{X}_u)$, since $H_0 = 0$.

For the final base case, we check the result for $(t = 1, u = 1)$, where we are only

adding one center and only one optimal cluster is uncovered. It helps to break up the expectation of the new potential into two terms by conditioning on an event and its complement and calculating the (bounds on the) conditional expectations in both cases.

Define event B as the event that we choose our new center from the one uncovered cluster. Note that $P(B) = \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}$ and $P(B^c) = \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}$. Then,

$$\begin{aligned}
\mathbb{E}[\phi'(\mathcal{X})] &= P(B)\mathbb{E}[\phi'(\mathcal{X})|B] + P(B^c)\mathbb{E}[\phi'(\mathcal{X})|B^c] \\
&= \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\mathbb{E}[\phi'(\mathcal{X})|B] + \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\mathbb{E}[\phi'(\mathcal{X})|B^c]
\end{aligned}$$

For the first term, we can apply Lemma 4.3.5 because we are conditioning on choosing from an uncovered cluster with D^2 weighting. Hence,

$$\begin{aligned}
\mathbb{E}[\phi'(\mathcal{X})|B] &= \mathbb{E}[\phi'(\mathcal{X}_c) + \phi(\mathcal{X}_u)|B] \\
&= \mathbb{E}[\phi'(\mathcal{X}_c)|B] + \mathbb{E}[\phi(\mathcal{X}_u)|B] \\
&\leq \mathbb{E}[\phi'(\mathcal{X}_c)|B] + 8\phi_{OPT}(\mathcal{X}_u) \text{ by Lemma 4.3.5} \\
&\leq \phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u),
\end{aligned}$$

where the final inequality is from the fact that $\phi'(A) \leq \phi(A)$ for any set A and for any new center.

For the second term, note $\phi'(\mathcal{X}) \leq \phi(\mathcal{X}) = \phi(\mathcal{X}_c) + \phi(\mathcal{X}_u)$. Hence,

$$\mathbb{E}[\phi'(\mathcal{X})|B^c] \leq \phi(\mathcal{X}_c) + \phi(\mathcal{X}_u).$$

Combining terms, we have that

$$\begin{aligned}
\mathbb{E}[\phi'(\mathcal{X})] &\leq \frac{\phi(\mathcal{X}_u) + \phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\phi(\mathcal{X}_c) + \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}8\phi_{OPT}(\mathcal{X}_u) + \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\phi(\mathcal{X}_c) \\
&= \phi(\mathcal{X}_c) + \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}8\phi_{OPT}(\mathcal{X}_u) + \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\phi(\mathcal{X}_c) \\
&\leq \phi(\mathcal{X}_c) + \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}8\phi_{OPT}(\mathcal{X}_u) + \phi(\mathcal{X}_c) \\
&= 2\phi(\mathcal{X}_c) + 8\phi_{OPT}.
\end{aligned}$$

Hence, the proposition holds for $t = u = 1$.

For the inductive step, suppose that the statement holds for $(t = t^* - 1, u = u^* - 1)$ and $(t = t^* - 1, u = u^*)$. We will again consider two cases by conditioning on an event and its complement. Define the event that the first center is chosen from an uncovered cluster as U. Clearly,

$$\begin{aligned}
\mathbb{E}[\phi'(\mathcal{X}); (t = t^*, u = u^*)] =& P(U)\mathbb{E}[\phi'(\mathcal{X})|U; (t = t^*, u = u^*)] \\
&+ P(U^c)\mathbb{E}[\phi'(\mathcal{X})|U^c; (t = t^*, u = u^*)].
\end{aligned}$$

For the first term, note that the first center is chosen from a covered cluster with

probability $P(U) = \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}$. Also, we have the loose bound

$$\mathbb{E}[\phi'(\mathcal{X})|U; (t = t^*, u = u^*)] \leq \mathbb{E}[\phi'(\mathcal{X}); (t = t^* - 1, u = u^*)]$$

based on the fact that choosing any new center improves the potential over not choosing a new center. Thus, applying the inductive hypothesis with $(t = t^* - 1, u = u^*)$, we have

$$P(U)\mathbb{E}[\phi'(\mathcal{X})|U; (t = t^*, u = u^*)]$$
$$\leq \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})} \left((\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u)) (1 + H_{t^*-1}) + \frac{u^* - (t^* - 1)}{u^*} \phi(\mathcal{X}_u) \right).$$

For the second term, assume that U^c occurs; that is, we choose the first center from an uncovered cluster. Further consider that we chose the new center from a particular uncovered cluster, say A, which occurs with probability $\frac{\phi(A)}{\phi(\mathcal{X}^{(u)})}$. Let E_a be the event that a point $a \in A$ was chosen conditioned on the fact that we chose from A using D^2 weighting. Define $p_a = Pr(E_a)$. Define $\phi_a(\cdot)$ as $\phi'(\cdot)$ conditioned on E_a.

Note that for any $a \in A$ chosen as the new center, $\mathcal{X}_u \leftarrow \mathcal{X}_u \setminus A$ and $\mathcal{X}_c \leftarrow \mathcal{X}_c \cup A$. Then, the contribution to the expectation of the final potential is

$$\frac{\phi(A)}{\phi(\mathcal{X}^{(u)})} \sum_{a \in A} \left(p_a \mathbb{E}\left[\phi_a(A \cup \mathcal{X}_c) + \phi_a(\mathcal{X}_u \setminus A); (t = t^* - 1, u = u^* - 1) \right] \right), \tag{4.1}$$

where the expectation is over the choice of $t^* - 1$ centers with $u^* - 1$ remaining

uncovered clusters.

Define $\Xi = (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u)(1 + H_{t^*-1})$ for notational purposes. Apply the inductive hypothesis with $u' = u^* - 1$ and $t' = t^* - 1$ to bound the sum in Equation (4.1) with

$$\begin{aligned}
&\sum_{a\in A} p_a \left((\phi_a(A \cup \mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u \setminus A))(1 + H_{t^*-1}) + \frac{u^* - t^*}{u^* - 1}\phi_a(\mathcal{X}_u \setminus A) \right) \\
&= \sum_{a\in A} p_a \left((\phi_a(A) + \phi_a(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u) - 8\phi_{OPT}(A))(1 + H_{t^*-1}) + \frac{u^* - t^*}{u^* - 1}\phi_a(\mathcal{X}_u \setminus A) \right) \\
&= \sum_{a\in A} p_a \left((\phi_a(A) - 8\phi_{OPT}(A))\,(1 + H_{t^*-1}) + \Xi + \frac{u^* - t^*}{u^* - 1}\phi_a(\mathcal{X}_u \setminus A) \right).
\end{aligned}$$

Note that $\phi_a(\mathcal{X}_u \setminus A) \leq \phi(\mathcal{X}_u \setminus A) = \phi(\mathcal{X}_u) - \phi(A)$. Making this substitution in the equality, we have that the contribution to the potential is bounded by

$$\frac{\phi(A)}{\phi(\mathcal{X}^{(u)})} \sum_{a\in A} p_a \left((\phi_a(A) - 8\phi_{OPT}(A))\,(1 + H_{t^*-1}) + \Xi + \frac{u^* - t^*}{u^* - 1}(\phi(\mathcal{X}_u) - \phi(A)) \right).$$

Because $\sum_{a\in A} p_a\phi_a(A) = \mathbb{E}[\phi'(A) | x \in A \text{ chosen with } D^2 \text{ weighting}]$, we can apply Lemma 4.3.5 to conclude $\sum_{a\in A} p_a\phi_a(A) \leq 8\phi_{OPT}(A)$. Hence, we replace the first term with ≤ 0. Next, note that the remaining terms do not depend on a specifically being chosen. Hence, the contribution to the potential is

$$\begin{aligned}
&\frac{\phi(A)}{\phi(\mathcal{X}^{(u)})} \sum_{a\in A} p_a \left((\phi_a(A) - 8\phi_{OPT}(A))\,(1 + H_{t^*-1}) + \Xi + \frac{u^* - t^*}{u^* - 1}(\phi(\mathcal{X}_u) - \phi(A)) \right) \\
&\leq \frac{\phi(A)}{\phi(\mathcal{X}^{(u)})} \left(\Xi + \frac{u^* - t^*}{u^* - 1}(\phi(\mathcal{X}_u) - \phi(A)) \right).
\end{aligned}$$

We know that $\sum_{i=1}^{u^*} \phi(A_i) = \phi(\mathcal{X}_u)$. Hence, the power-mean inequality gives that

$$-\sum_{i=1}^{u^*} \phi(A_i)^2 \leq -\frac{1}{u^*}\phi(\mathcal{X}_u)^2. \tag{4.2}$$

Sum over all uncovered A_i to give contribution less than or equal to

$$\begin{aligned}
& \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\Xi + \frac{1}{\phi(\mathcal{X}^{(u)})}\left(\frac{u^*-t^*}{u^*-1}(\phi(\mathcal{X}_u)^2 - \sum_{i=1}^{u^*}\phi(A_i)^2)\right) \\
\leq\ & \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\Xi + \frac{1}{\phi(\mathcal{X}^{(u)})}\left(\frac{u^*-t^*}{u^*-1}\left(\phi(\mathcal{X}_u)^2 - \frac{1}{u^*}\phi(\mathcal{X}_u)^2\right)\right) \text{ by Equation (4.2)} \\
=\ & \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\left(\Xi + \frac{u^*-t^*}{u^*}\phi(\mathcal{X}_u)\right).
\end{aligned}$$

Therefore,

$$P(U^c)\mathbb{E}[\phi'(\mathcal{X})|U^c;(t=t^*,u=u^*)] \leq \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\left(\Xi + \frac{u^*-t^*}{u^*}\phi(\mathcal{X}_u)\right).$$

Combining terms, we have that

$$\begin{aligned}
&\mathbb{E}[\phi'(\mathcal{X});(t=t^*,u=u^*)] \\
&\quad \leq \Xi + \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\left(\frac{u^*-t^*+1}{u^*}\phi(\mathcal{X}_u)\right) + \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\left(\frac{u^*-t^*}{u^*}\phi(\mathcal{X}_u)\right) \\
&\quad = \Xi + \frac{u^*-t^*}{u^*}\phi(\mathcal{X}_u) + \frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{\phi(\mathcal{X}^{(u)})}\frac{1}{u^*}\phi(\mathcal{X}_u) \\
&\quad = (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))(1+H_{t^*-1}) + \frac{u^*-t^*}{u^*}\phi(\mathcal{X}_u) + \frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})}\frac{\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)})}{u^*}
\end{aligned}$$

Because $\frac{\phi(\mathcal{X}_u)}{\phi(\mathcal{X}^{(u)})} \leq 1$ and $\phi(\mathcal{X}_c \cap \mathcal{X}^{(u)}) \leq \phi(\mathcal{X}_c) \leq \phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u)$, we can combine the last and first terms in the final inequality to yield

$$\mathbb{E}[\phi'(\mathcal{X}); (t = t^*, u = u^*)] \leq (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))(1 + H_{t^*-1} + \frac{1}{u^*}) + \frac{u^* - t^*}{u^*}\phi(\mathcal{X}_u).$$

Finally, since $H_{t^*-1} + \frac{1}{u^*} \leq H_{t^*-1} + \frac{1}{t^*} = H_{t^*}$, we have that

$$\mathbb{E}[\phi'(\mathcal{X}); (t = t^*, u = u^*)] \leq (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))(1 + H_{t^*}) + \frac{u^* - t^*}{u^*}\phi(\mathcal{X}_u),$$

as desired. ∎

Theorem 4.3.7 *Suppose our story about how the supervision occurs holds. Let* $C = \emptyset$. *For each label* ℓ *that we have supervised exemplars of, add the centroid of the supervised data labeled* ℓ, *say* c_ℓ *to* C. *Suppose that* $card(C) = G$. *Let* n_{ℓ_j} *be the number of supervised exemplars with label* ℓ_j *for* $j = 1, 2, \ldots, G$. *Then, we have* $u = k - G$ *uncovered clusters. Add* $t = u$ *new centers using* D^2 *weighting ignoring the supervised points. The expectation of the resulting potential,* ϕ', *is then*

$$\mathbb{E}[\phi'(\mathcal{X})] \leq 8\phi_{OPT}(\mathcal{X})(2 + \log(k - G)).$$

Proof Note that after supervision, $\mathcal{X}_c = \bigcup_{j=1}^{G} \mathcal{X}_{\ell_j}$, where $\mathcal{X}_{\ell_j}$ is an optimal cluster which we have covered through supervision and the double subscripts ℓ_j are to account for a possible relabeling. Hence, $\mathcal{X}_u = \mathcal{X} \setminus \bigcup_{j=1}^{G} \mathcal{X}_{\ell_j}$. Therefore, $\phi(\mathcal{X}_c) = \sum_{j=1}^{G} \phi(\mathcal{X}_{\ell_j})$,

and $\phi_{OPT}(\mathcal{X}_u) = \phi_{OPT}(\mathcal{X}) - \sum_{j=1}^{G} \phi_{OPT}(\mathcal{X}_{\ell_j})$. Then, applying Lemma 4.3.6 with $u = t = k - G$, we have

$$\begin{aligned}
&\mathbb{E}[\phi'(\mathcal{X}); (t = k - G, u = k - G)] \\
&\qquad \leq (\phi(\mathcal{X}_c) + 8\phi_{OPT}(\mathcal{X}_u))(1 + H_{k-G}) \\
&\qquad = \left(\sum_{j=1}^{G} \left(\phi(\mathcal{X}_{\ell_j}) - 8\phi_{OPT}(\mathcal{X}_{\ell_j}) \right) + 8\phi_{OPT}(\mathcal{X}) \right) (1 + H_{k-G}).
\end{aligned}$$

Applying Lemma 4.3.4 to each $\phi(\mathcal{X}_{\ell_j})$, we have

$$\mathbb{E}[\phi'(\mathcal{X})] \leq \left(\sum_{j=1}^{G} \left(-7 + \frac{n - n_{\ell_j}}{n_{\ell_j}(n-1)} \right) \phi_{OPT}(\mathcal{X}_{\ell_j}) + 8\phi_{OPT}(\mathcal{X}) \right) (1 + H_{k-G}).$$

Finally, using the fact that $H_{k-G} \leq 1 + \log(k - G)$, we have our result. ∎

The end result is a modest improvement over that of [5] that scales with the level of supervision. The final inequality in the proof is tighter than the result stated in the theorem, since the factor of 8 could be lower depending on the contributions of the supervised clusters in the optimal clustering.

4.4 Numerical Experiments

4.4.1 Performance Measures

We use several measures for each experiment. First, we use the cost, or the potential function with the final centers. For comparing to the theoretical bound, we also use the fraction of optimal cost, where "optimal" is derived by taking the centroids for each class as determined by the ground truth labels. Next, we use the number of Lloyds iterations until convergence.

Finally, we will use the Adjusted Rand Index (ARI) [26], which is an index that compares how closely two partitions agree. The ARI is the Rand index, the ratio of number of agreements between two partitions, after adjusting for chance. It is essentially chance at 0, meaningless < 0, and perfect at its maximal value, unity. Since ground truth labels are available for our datasets, we can compare them to the partitions yielded from the output of the algorithms in Section 4.4.3. Thus, a large ARI value indicates good clustering performance as determined by fidelity to the ground truth partition.

4.4.2 Data

We showcase our algorithm on three datasets (cf. Figure 4.2 for depiction). The first, `Gaussian Mixture` was inspired by both [5, 8]. We drew $k = 24$ centers from

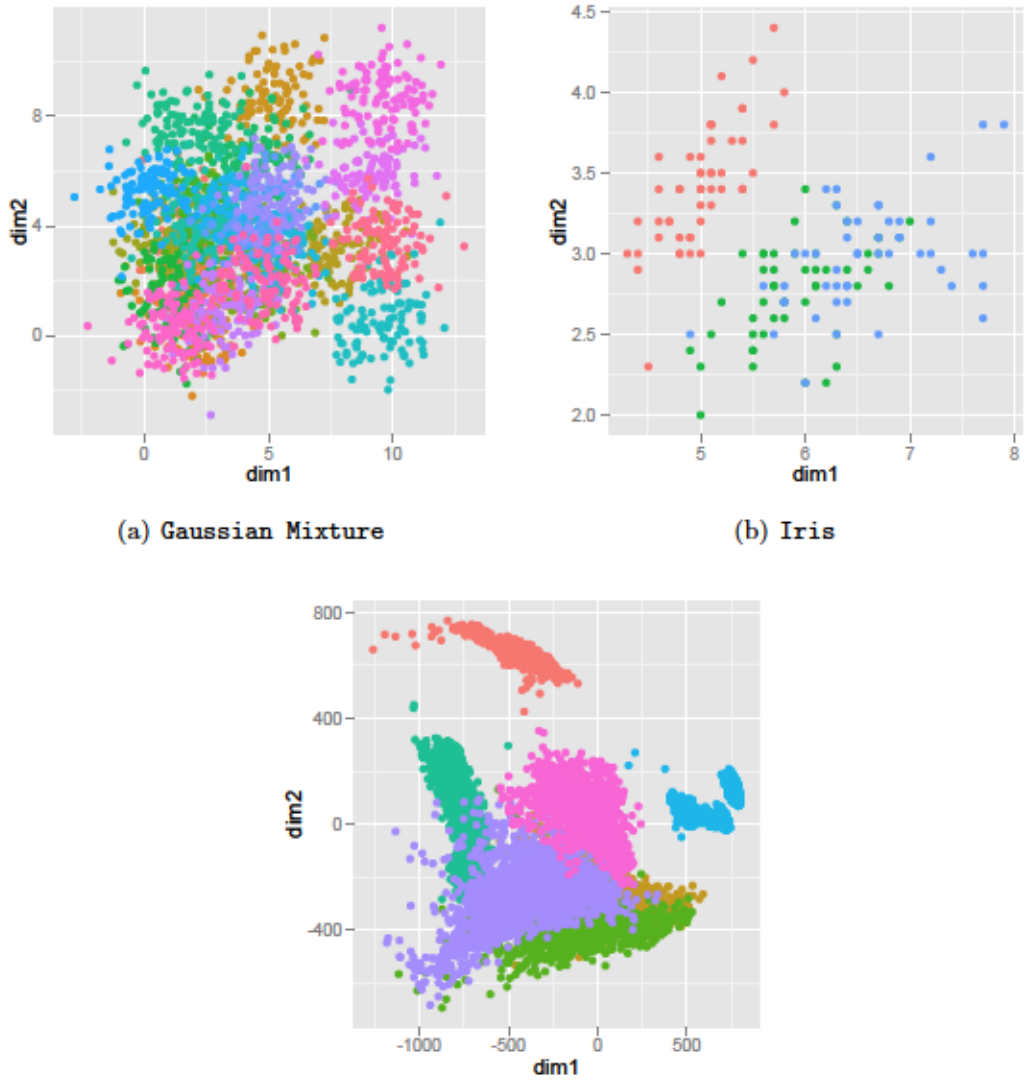
the 15-dimensional hypercube with side length of 10. For each center c_ℓ, we drew $n_\ell = 100$ points from a multivariate Guassian with mean c_ℓ and identity covariance. This dataset is remarkable because it is easy to cluster by inspection (at least with larger side-length, as in the original papers) yet is difficult for Lloyd's algorithm when initialized with bad centers. For our chosen side length, it is not easy to cluster by eye. Note that the supervision story (where centroids of the class labels correspond to best centers) is likely to hold for most realizations of the data.

The next two datasets are real data for which the assumption that the labels match up with minimum cost clusters is not met. The second dataset is the venerable `Iris` dataset [19], which uses $d = 4$ variables to describe $k = 3$ different classes of flowers. While this dataset is old, it is nonetheless difficult for `k-means` to handle from a clustering standpoint. This fact is widely known; indeed, even the Wikipedia page for the `Iris` dataset has a rendering of `k-means` failing on it.[55] We compared the ARI for this dataset and the `Gaussian Mixture` dataset while varying the ratio of side length of the hypercube to standard deviation ($\Sigma = \sigma I$ with $\sigma = 1$ fixed), and we found that the datasets were roughly equivalent for side length around 3.25. This is under one percent of the side length and 10^{-32} times the volume of the `norm25` dataset [8] that our `Gaussian Mixture` dataset is based on. Thus, we observe that the `Iris` dataset is harder to cluster than the synthetic dataset.

The third dataset, `Hyperspectral`, is a Naval Space Command HyMap hyperspectral dataset representing an aerial photo a runway and background terrain of the

Figure 4.1: Image corresponding to the `Hyperspectral` dataset as seen in Figure 2 of [45]. Each pixel can be classified according to what it represents.

airport at Dalgren, Virginia as originally seen in [45] (cf. Figure 4.1 for a depiction of the location). Each pixel in the photo is associated with $d = 126$ features representing different spectral bands (e.g. visible and infrared). We took the first six principal components to form a dataset with $n = 14748$ data in $\mathbb{R}^6$, as chosen by the minimum number of dimensions to capture $> 97.5\%$ of the total variance. The first two principal components are depicted in Fig 4.2. The $k = 7$ classes are the identities of each pixel (i.e. runway, pine, oak, grass, water, scrub, and swamp). Based on the ARI scores presented in the forthcoming results section, this dataset is only a little easier to cluster than `Iris`.



(a) `Gaussian Mixture`

(b) `Iris`

(c) `Hyperspectral`

Figure 4.2: First two dimensions of the datasets (one realization for `Gaussian Mixture`). Because `Gaussian Mixture` has 13 more dimensions than are shown here, clustering it is considerably easier than this figure would imply. Note, however, that we have overlapping classes (as denoted by the colors) in all datasets.

4.4.3 Algorithms

For both datasets, we apply several algorithms: `ss-k-means++`, `Constrained-KMeans`, and `Constrained-KMeans` algorithm initialized at the true class centroids. We consider the latter algorithm as an approximation to the optimal solution. Also, because the "++" versions of the algorithms supposedly improve the initialization, we consider both `ss-k-means++` (without Lloyds) and `Constrained-KMeans` (without Lloyds) to verify this improvement. `Constrained-KMeans` and `Constrained-KMeans` (without Lloyds) use a random sample of the unsupervised data weighted uniformly for the remaining initial centers (after using centroids of the labeled points). The algorithms without Lloyds use their respective initialization strategy to choose initial centers then move straight to class assignment without updating the initial centers. We consider these algorithms as "initialization only" methods for this reason.

4.4.4 Results

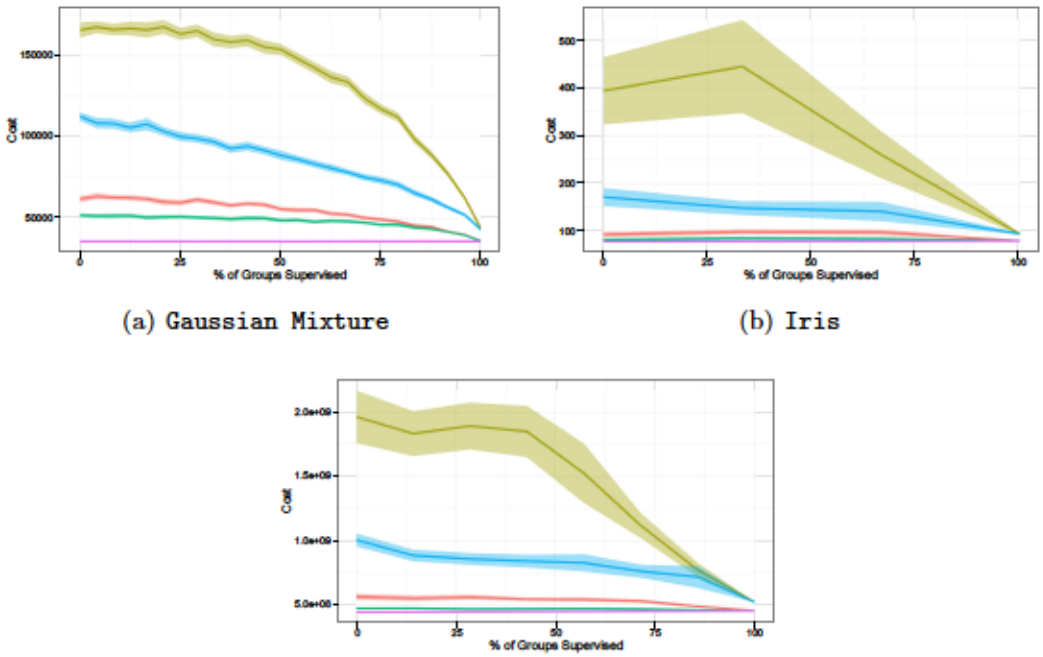
We vary the supervision level from 0% to 100%, where we add supervised classes and sample 5, 5, and 50 datapoints per class to label for `Guassian Mixture`, `Iris`, and `Hyperspectral`, respectively. Note that this is percent of clusters which have exemplars and not percent of all points which are labels. Also, at 100% supervision, `ss-k-means++` and `Constrained-KMeans` are the same, since there are no additional centers to choose. We did not allow the supervised data to change cluster assignment,

so the approximation to the optimal can change with the level of supervision and with different supervised data chosen . We set k equal to the true number of groups (24 for `Gaussian Mixture`, 3 for `Iris`, and 7 for `Hyperspectral`). We used 100 Monte Carlo replicates at each level of supervision.

Figure 4.3 shows the cost as the level of supervision changes. We observe the cost decreases with more supervision. Also, we see the same relative performances of the algorithms, with the ++ version outperforming the benchmark. Observe that the approximation to the optimal solution is the best. Figure 4.4 depicts the theoretical bound. All algorithms are below the bound (in expectation).

Figure 4.5 shows the number of iterations before Lloyd's converges. We can see that improved selection of initial centers by D^2 weighted randomization leads to fewer iterations before convergence. We expected this; Arthur and Vassilvitskii [5] observed a similar phenomenon with no supervision. More supervision did not seem to affect the number of iterations until very high levels (near 100%). For the real world datasets, we can see that the approximation to the optimal algorithm required more than one iteration to converge, indicating that the centroids of the true class labels do not match with the locally minimal cost solutions. This means that the conditions for the supervision in our proofs do not hold for this dataset. Nevertheless, both cost and ARI improve with additional supervision.

Figure 4.6 shows the ARI for all algorithms. Note that supervision improves the ARI, as expected. Also, `ss-k-means++` generally outperforms `Constrained-KMeans`.

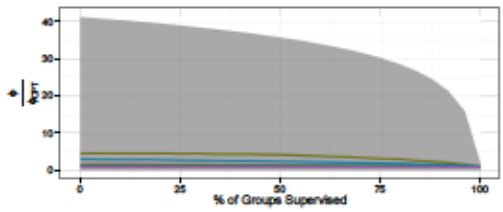


(a) `Gaussian Mixture` (b) `Iris`

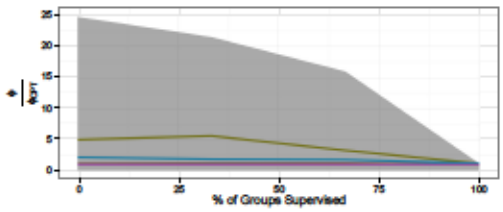
(c) `Hyperspectral`

Figure 4.3: Cost (value of the potential) shown as a function of the level of supervision for 100 Monte Carlo replicates. Shading indicates ± two standard deviations. Colors indicate algorithm:
gold: `Constrained-KMeans` (without Lloyds iterations);
blue: `ss-k-means++` (without Lloyds iterations);
red: `Constrained-KMeans`;
green: `ss-k-means++`; and
pink: `Constrained-KMeans` initialized at true centroids of labels.

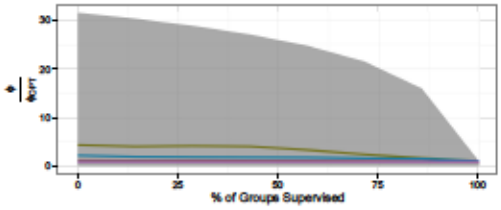
The same observation holds for the initialization only versions as well. Remarkably, the true centroids and Lloyd's algorithm is outperformed by the initialization only methods on the `Iris` and `Hyperspectral` datasets at 100% supervision for the ARI metric. This is due to the fact that the true classes do not correspond to the minimum cost solution, which is what Lloyd's iterations would improve (apparently at the cost of ARI).



(a) `Gaussian Mixture`



(b) `Iris`



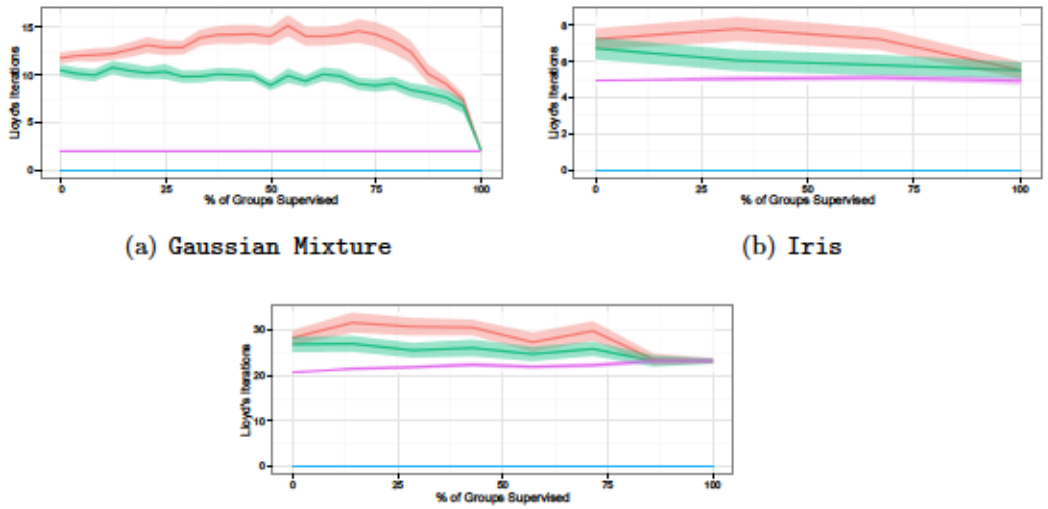
(c) `Hyperspectral`

Figure 4.4: Fractional cost (value of the potential over an estimate of the optimal) plotted as a function of the level of supervision for 100 Monte Carlo replicates. Shading around the lines indicates ± two standard deviations. The shaded region is the region corresponding to the theoretical cost in expectation from Section 4.3. Colors indicate algorithm:
gold: `Constrained-KMeans` (without Lloyds iterations);
blue: `ss-k-means++` (without Lloyds iterations);
red: `Constrained-KMeans`;
green: `ss-k-means++`; and
pink: `Constrained-KMeans` initialized at true centroids of labels.

4.5 Conclusions

In this chapter, we present a natural extension of `k-means++` and `Constrained-KMeans`. Then, we prove the corresponding bound on the expectation of the cost under some conditions on the supervision. No assumptions are made about the distribution of the data. Finally, we demonstrated that on three datasets judicious supervision and good starting center selection heuristics improve clustering performance, cost, and iteration count.

figure

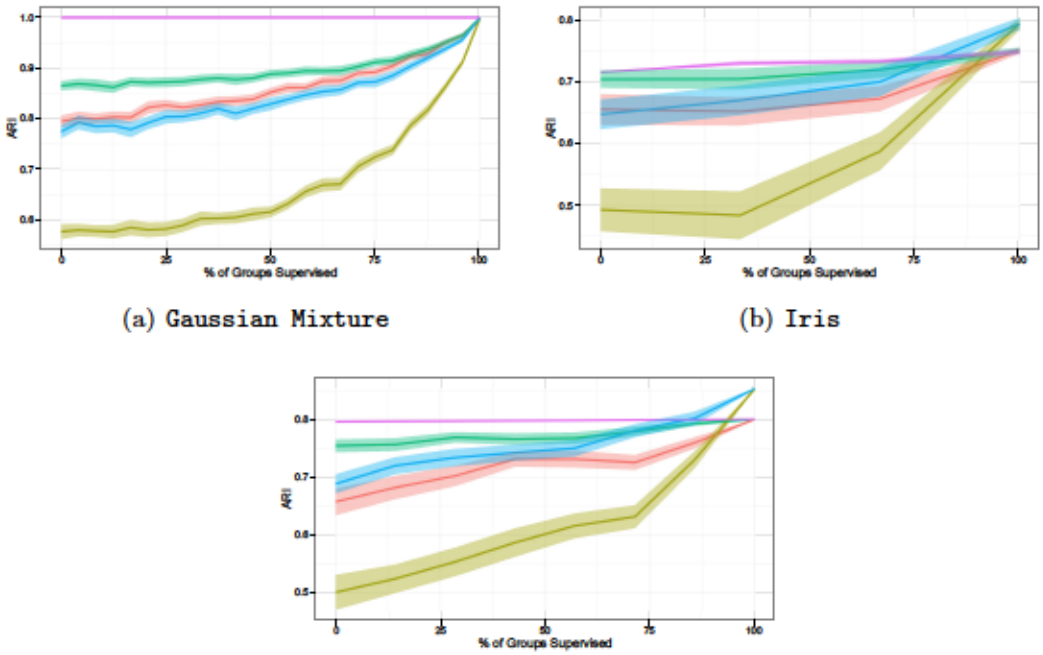


(a) `Gaussian Mixture` (b) `Iris`

(c) `Hyperspectral`

Figure 4.5: Lloyd's iterations before convergence plotted as a function of the level of supervision for 100 Monte Carlo replicates. Shading indicates ± two standard deviations. Colors indicate algorithm:
gold: `Constrained-KMeans` (without Lloyds iterations);
blue: `ss-k-means++` (without Lloyds iterations);
red: `Constrained-KMeans`;
green: `ss-k-means++`; and
pink: `Constrained-KMeans` initialized at true centroids of labels.

Possible future theoretical work includes incorporating the advances set forth in the extensions to the original `k-means++` paper. For example, we could produce semi-supervised versions of `k-means#` [8] and `k-means||` [4] with commensurately improved bounds. Relaxing the constraints to the pairwise cannot-link and must-link constraints as in [54] is also desirable, because the assumption of exogenously provided hard labels is often untenable. Other assumptions that would be nice to relax would be the equal cluster shapes and cluster volume implicit in `k-means` clustering.



(a) `Gaussian Mixture`

(b) `Iris`

(c) `Hyperspectral`

Figure 4.6: Average ARI shown as a function of the level of supervision for 100 Monte Carlo replicates. Shading indicates ± two standard deviations. Green beats red. Colors indicate algorithm:
gold: `Constrained-KMeans` (without Lloyds iterations);
blue: `ss-k-means++` (without Lloyds iterations);
red: `Constrained-KMeans`;
green: `ss-k-means++`; and
pink: `Constrained-KMeans` initialized at true centroids of labels.

Chapter 5

Applications

I'm not crazy about reality, but it's still the only place to get a decent meal.

Groucho Marx

In this chapter, we present two applications of semi-supervised model based clustering. The first experiment section is based on our work in [58].

5.1 Identifying Fly Behaviotypes

In one of the motivating applications for this work, classes of neurons in *Drosophilia* larvae are controlled using optogenetics (cf. [1] regarding optogenetics). In [52], they observe the reactions of the affected larvae to stimuli in high-throughput behav-

ioral assays. The goal is to determine which classes of neurons cause similar changes in behavior when deactivated.

We initially collected data on $n = 37780$ larvae grouped into $b = 2062$ dishes. By changing the optogenetic procedure, $\ell = 11$ known lines are created, pbd1, 38a1, 61d0, ppk1, 11f0, pbd2, 38a2, pbd3, ppk2, iav1, and 20c0. Of these lines, we discarded the larvae in pbd3 and ppk2 because they had less than 40 larvae each. Further, we discarded all larvae with an unknown line. After curating the data, we now have $n = 7730$ larvae. Each larva is observed while responding to various stimuli. Vogelstein *et al.* [52] expand on the methods of [44] and [45] and describe how the observations are embedded into $\mathbb{R}^d$, where here $d = 30$. We use the method presented in [61] to select the elbow of the scree plot to further reduce the data to $d = 14$ dimensions. We did not perform any additional feature selection.

For each Monte Carlo replicate, we use a small subset of the data where the line was known (101 randomly chosen animals from each of the 9 remaining lines) along with $m = 0, 1, 2, \ldots 8$ pre-labeled data randomly chosen from the 101 animals in each line. We wrote an R package entitled `ssClust` to perform semi-supervised GMM with similar options to the popular Mclust software, which is an R package for GMM [25]. We cluster the points using both `ssClust` and Mclust. Then, we compute the ARI against the line type for both methods.

We observe that the initialization strategy of using `ss-k-means++` instead of hierarchical clustering results in a significant improvement to the ARI even with no

supervision. We find that a single animal per line significantly improves the clustering results, as expected (cf. Figure 5.1). Further, we can see that there are diminishing returns on additional supervision starting at 3 supervised examples per line.

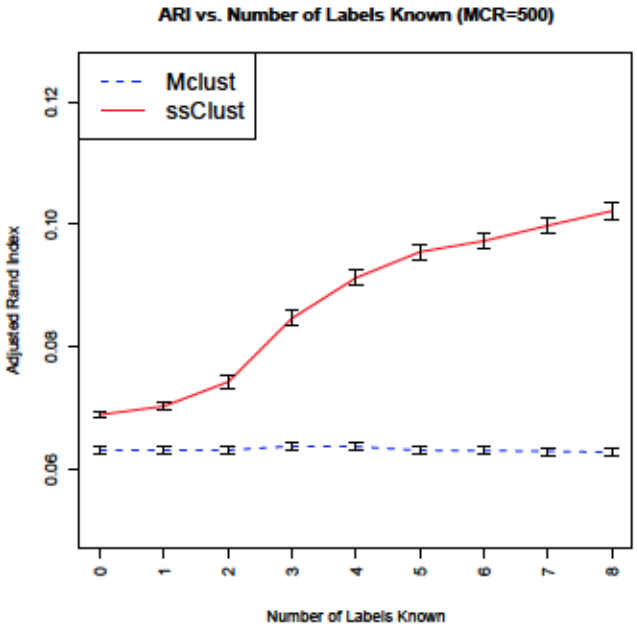


Figure 5.1: Average value of the ARI for 500 Monte Carlo replicates for experiment 5.1. Error bars are ± 2 standard errors.

5.1.1 Differentiating Lines

Vogelstein *et al.* [52] posed and answered questions of the form, "Is line X different than line Y (in terms of behaviotypes)?" As an illustrative example, we will perform a similar analysis comparing the ppk1 and pbd1 lines but will additionally incorporate some supervision. Here, our interpretation of the clusters will shift from lines to behaviotypes. Our proposed procedure for distinguishing between lines is as follows:

(1) Sample 101 animals from each line

(2) Label 3 animals from each line according to a labeling strategy (see below)

(3) Cluster the animals using `ssClust` and Mclust into between 2 and 12 clusters using the parameterizations EEE, VVV, VII, and EII.

(4) Collect the results and construct empirical probability of cluster membership for each line

(5) Compute the Hellinger distance between the two lines to be compared and store this as statistic H

(6) Simulate the distribution of H under the null hypothesis that the lines are the same by permuting the labels and computing the Hellinger distance H_i for $i = 1, 2, \ldots, B$ for some large integer B.

(7) Return an empirical p-value based on steps (5) and (6).

In item (2) we did not specify a labeling strategy in detail. We propose a strategy that is reasonably realistic to execute for our particular dataset. Vogelstein *et al.* [52] used a hierarchical clustering scheme in which the first few layers were visually identifiable by watching the worms. Thus, by using their labels from an early layer (layer 2, with 4 clusters total), we have a plausible level of supervision for a human to have performed. Specifically, we sample at random a label from the true labels among a line with weights proportional to the counts of each label in that line. Using that

label, we sample 3 worms of that label in that line to be the supervised examples. Next, for the other line, we sample a different label, and 3 examples with that true label.

We see that based on all three labeling strategies that both `ssClust` and Mclust are able to corroborate the results from [52] even with small amounts of data: ppk1 and pbd1 are statistically different (p-value $\approx 1e-4$ for both for 500 MC replicates).

We now show that `ssClust` can answer these questions "sooner" than Mclust. That is, the p-value ($pVal$) will be below the significance level with fewer unsupervised examples. To quantify this concept, we introduce the "answering time" for algorithm A:

$$\tau_A := \min_{n \in \mathbb{N}} \left\{ n : 2 \leq n \leq 30 \text{ and } \prod_{q=n}^{30} \mathbb{1}\{pVal\left(\mathcal{D}_k\right) \leq \alpha\} = 1 \right\},$$

where here we use the notation

$$\mathcal{D}_k := \{X_{1,k}, \ldots, X_{q,k}, (X_{99,k}, Y_{99,k}), \ldots, (X_{101,k}, Y_{101,k}) : k \in \{1, 2\}\}$$

to be the labeled and unlabeled data.

The p-value constraint bears some explanation; it says that we require a significant p-value for all datasets at least as large as with n unsupervised examples per line. Note that here we assume our datasets are nested and that they all use the same supervised examples. Since the answering time will be dependent on the datasets

used, we perform a Monte Carlo simulation with 500 random sequences of datasets and report the answering times for both `ssClust` and Mclust (cf. Figure 5.2). The median answering time for `ssClust` is significantly lower than for Mclust (p-value = 4.7e-12 for paired Wilcoxon signed-rank test).

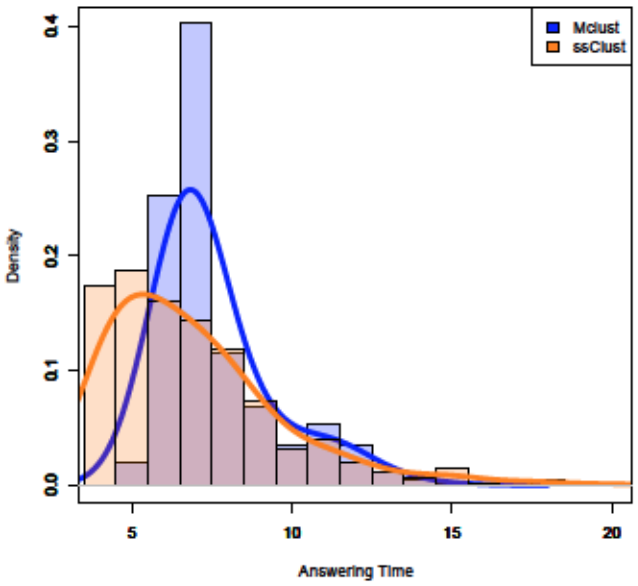


Figure 5.2: Frequency distribution of answering times for 5.1.1 given 500 MC replicates.

5.2 Vertex Nomination

Consider a graph $G = (V, E)$. Suppose that there is a labeling function on the vertices, say

$$\ell : V \to C,$$

where $C = \{\text{red, blue, yellow, } \dots\}$ is a set of colors on the vertices and **red** is the color of interest. In vertex nomination problems, the structure of the graph is known (i.e. G), but the images of only some (or none) of the vertices under the mapping ℓ are known. The task is to nominate ambiguously colored vertices based on their relative certainty of being red. Coppersmith [14] performs a recent review on vertex nomination in the literature.

In our formulation of the vertex nomination problem, we can view it as a semi-supervised clustering problem on graphs with only one cluster being of interest. In order to apply the methods explicated in previous chapters of this thesis, we need to produce real valued features for each vertex. Spectral methods have long been used to embed graphs in $\mathbb{R}^d$ with the aim of clustering (e.g. [17, 28, 42]) . For an excellent tutorial on spectral clustering, see [53]. Using clustering for finding interesting vertices in graphs through latent positions has been studied in other ways; for example, in [30], we directly infer latent positions during a streaming doubly stochastic process. Actually using the nomination list to modify the graph to fit some goal is a subject of further study; in [39], we simulated the effects of a sequence of graph modifications on expected terrorist attacks.

Fishkind *et al.* [20] compare spectral clustering to a likelihood based and to a gold standard canonical clustering method on a special type of random graphs, stochastic block models (SBM). In random graph models, V is generally fixed, and the set E is a random variable. In a stochastic block model, there are a number of blocks (colors),

much smaller than the number of vertices. Each vertex is in a block. Every pair of vertices independently has an edge with probability related only to their block memberships. Formally, if there are $K = card(C)$ blocks, $\Lambda \in \mathbb{R}^{K \times K}$ is the link probability matrix such that

$$P\left[(u,v) \in E | \ell(u) = i, \ell(v) = j\right] = \Lambda_{i,j}.$$

Spectral clustering to recover the blocks in SBM graphs has been shown to be efficacious. [47]

The Random Dot Product Graph (RDPG) model is a related random graph model. Each vertex $v \in V$ has a latent position $x_v \in \mathbb{R}^d$, where d is generally unknown. Instead of a link probability matrix dependent on blocks, the link probabilities are based on the latent positions. Suppose $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a kernel function and $g : \mathbb{R} \to [0,1]$ is a link function. Then,

$$P\left[(u,v) \in E\right] = g(\mathcal{K}(x_u, x_v)).$$

If an SBM has as positive semi-definite link probability matrix, then it is a special case of a RDPG model in which the latent positions of the vertices are fully specified by their block memberships. RDPG models are well described by spectral embedding. [51, 50] Hence, spectral embedding on SBM graphs is often appropriate. [49] Then, since the estimated locations in the latent space are strongly related to

the block membership, a clustering in the latent space generally recovers block membership nicely. [32, 21] Finally, Athreya *et al.* [6] show that the scaled eigenvectors (i.e. spectral embedding) of RDPG are approximately distributed as a mixture of Gaussians. Hence, using model-based clustering in this setting is applicable.

In some vertex nomination applications, there may be partially labeled data with additional information indicating that some of the vertices are red and some are not red. For notational purposes, let the colors correspond to natural numbers. Only red, the color of interest, need have an explicit correspondence: red = 1; further, we will often refer to nodes in block 1 as being red. Thus, the aforementioned constraints may be that $\ell(v_1) = 1$ and $\ell(v_2) \neq 1$. The second piece of a-priori knowledge is clearly weaker than label information, as $\ell(v_2)$ could be any other number (e.g. 2, 3, 4). We propose a model-based method that leverages the known constraints to provide a confidence that a vertex is interesting. The overview of the `ssVN` method is as follows:

1. ASE and elbow finder to get vertices as points in $\mathbb{R}^d$
2. `ss-k-means++` for initialization
3. GMM with partially supervised data using Algorithm 5
4. Use the modified BIC of Section 3.2

First, we use Adjacency Spectral Embedding (ASE) on the adjacency matrix of the observed graph to create d features for each vertex. To obtain the ASE, we take

the SVD of the adjacency matrix, $A = U\Sigma V^T$, and after choosing an appropriate dimension d, set

$$\hat{X} = V_d S_d^{\frac{1}{2}},$$

where $V_d \in \mathbb{R}^{n \times d}$ is the first d columns of V and $S_d^{\frac{1}{2}}$ is the $d \times d$ diagonal matrix with diagonal equal to square roots of the top d eigenvalues of A. Proximity in this latent space indicates similarity. We choose d with the elbow-finding scheme of [61] if the rank of the block probability matrix is unknown. On the other hand, if it is known, we set d equal to that. Thus, we will use $\hat{X}$ to assign a cluster (color or block) label to each vertex, where only the first cluster is particularly of interest, but a variable number of other clusters are allowed in order to better explain the structure in the latent space. We used our R package `ssClust` to perform the clustering steps.

5.2.1 Incorporation of Constraints

Now, we assume that we have additional information regarding the cluster membership of some of the vertices *a-priori*. Let Z_i be the hidden block membership, $\ell(v_i)$. If we know that some of the vertices are in our block of interest, we can set $Z_i = 1$ for those data, so that $X_i \sim f_{\theta_1}$. Further we know that some of the vertices are not in our block of interest. Decompose $\{1, 2, \ldots n\} = T + T' + T^C$, where

$$T = \{i : X_i \text{ is known to be from block 1}\},$$

$$T' = \{i : X_i \text{ is known not to be from block 1}\},$$

and

$$T^c = \{1, 2, \ldots n\} - T - T'.$$

Then, we can decompose the likelihood of the complete data as

$$\begin{aligned}\mathcal{L}(\theta)_{X,Z} = &\sum_{i \in T} f_1(x_i) \\ &+ \sum_{i \in T'} \sum_{k=2}^{G} Z_{i,k} \log\left(\frac{\pi_k}{1-\pi_1} f_k(x_i)\right) \\ &+ \sum_{i \in T^c} \sum_{k=1}^{G} Z_{i,k} \log(\pi_k f_k(x_i))\end{aligned} \tag{5.1}$$

The update equations for various constrained forms of Gaussian density functions are available in [48]. Using these equations, we look for the Z_i maximizing the complete loglikelihood. Note that we actually obtain posterior probabilities of block memberships, and thus return only soft labels for most vertices.

5.2.2 Synthetic Data Example

To demonstrate the effectiveness of partially labeled GMM using smart initialization, we performed an experiment wherein there were 3 blocks, 20 examples of block 1, and an increasing number of examples of vertices known not to be in block 1 (call these softly labeled vertices). The details of the simulation match the medium-sized

Algorithm 5: Semi-supervised Gaussian Mixture Modeling

Input: $\mathcal{X}$ (n datapoints in $\mathbb{R}^d$)
T (set of indices of data known to be in block 1)
T' (set of indices of data known not to be in block 1)
$g_i \in [i]^n$ (class of auxiliary cluster assignments into i clusters for $i = 2, 3, \ldots, G_{max}$)
M (set of models to consider)

Output: $m^* \in M$ (chosen model)
$\ell : \mathcal{X} \to \{1, 2, \ldots, k\}$ (cluster assignment mapping, where k is as according to m^*)

foreach $m \in M$ **do**
  Let G be the number of clusters according to m.
  Let f_k be the k^{th} component of the mixture according to m.
  Form the complete loglikelihood using equation (5.1)
  Maximize the complete loglikelihood by using the E-M algorithm using g_G as the initial class labels and starting with the M-step.
  Let $\mathcal{L}_m$ be the value of the maximized loglikelihood.
  Let $\ell_{m,j} = \text{argmax}_{k \in [G]} \pi_k f_k(x_j)$ for $j \notin T \cup T'$.
  Let $\ell_{m,j} = \text{argmax}_{k \in \{2,3,\ldots,G\}} \pi_k f_k(x_j)$ for $j \in T'$.
  Let $\ell_{m,j} = 1$ for $j \in T$.
  Let $BIC_m = 2\mathcal{L}_m - d_m \log(n)$.
Let $m^* = \text{argmax}_{m \in M} BIC_m$.
return m^*, ℓ_{m^*}

experiment in Section 7 of [20].

Let $\Lambda_1 = \begin{bmatrix} .5 & .3 & .4 \\ .3 & .8 & .6 \\ .4 & .6 & .3 \end{bmatrix}$ and $\Lambda_2 = \begin{bmatrix} .5 & .5 & .5 \\ .5 & .5 & .5 \\ .5 & .5 & .5 \end{bmatrix}$.

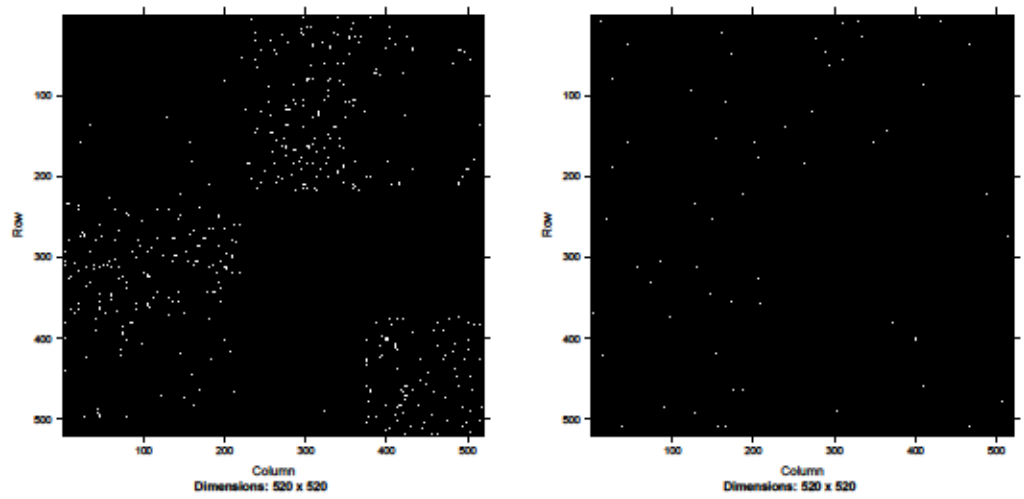
The block link probability matrix is given as

$$\Lambda = .3\Lambda_1 + .7\Lambda_2,$$

where the first matrix is regarded as the signal and the second as noise which serves to occlude the distinctiveness of the blocks. Figure 5.3 depicts example realizations of a pure (signal-only) and occluded SBMs as used in this experiment. Finally, we varied the number of softly labeled (i.e. known not to be from block 1) vertices from $0, 5, 10, \ldots, 100$.

Due to the fact that vertex nomination entails submitting a list of vertices in order of likelihood of being red, we would like to use different performance metrics than in our other experiments. Suppose that x_i is the i^{th} vertex in the nomination list. For example, x_1 is the vertex estimated to be most likely to be red. Note that we will exclude the 20 known red vertices from all performance metrics. Recall that $\ell(x_i)$ is the block number of vertex i.

First, we will use mean average precision at k (for k equal to the number of vertices in block 1 that have not been pre-labeled), which which for c Monte Carlo replicates is $m(k) = \frac{1}{c}\sum_{i=1}^{c}\left(\frac{1}{k}\sum_{j=1}^{k}\mathbb{1}\{\ell(x_i) = 1\}\right)$. Figure 5.4 shows the mean average pre-



(a) $B = \Lambda_1$ (b) $B = \Lambda$

Figure 5.3: Realizations of SBM with their respective inter-block probability matrices, B.

cision as a function of the number of partially labeled vertices for $c = 100$ Monte Carlo replicates. As expected, more partially labeled vertices improves performance. Compare this to the unlabeled table of mean average precision values in Section 7 of [20], where the mean average precision of the most comparable method (ASE + `k-means` without supervision) is 0.7330.

Next, we show the probability of $\ell(x_i) = 1$ for most of the vertices whose block is unknown. After 200, we would like to see the numbers drop rapidly. Figure 5.5 depicts these probabilities with three different levels of partial supervision (i.e. more vertices are known not to be red). Note that the curve becomes more sigmoidal with increased supervision, indicating improving performance.

For comparison, refer to Figure 5.6, which is the same plot for 100 Monte Carlo

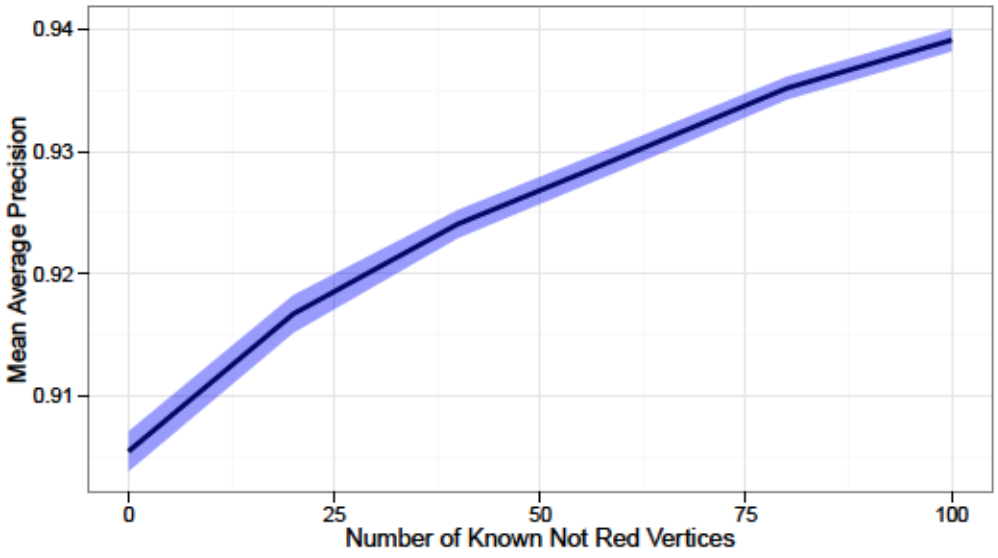


Figure 5.4: Mean average precision as a function of the number of vertices known not to be red. Shading represents $\pm$ 2 standard errors.

replicates with two different algorithms. The first is a canonical sampling scheme which estimates the posterior prbablitlity of block membership in the natural MCMC extension of the canonical method of [20]. We set the number of samples and number of burn ins to 10^4. The other is `ssVN` without using semi-supervised GMM after the `ss-k-means++` initialization. Note that using GMM improves performance drastically. The comparison method implementations are in MATLAB. Thus, while the simulation details are the same, the runtimes are incomparable.

5.2.3 Youtube Dataset

In this section, we demonstrate the efficacy of our algorithms on a real data application with dubious block-model structure.

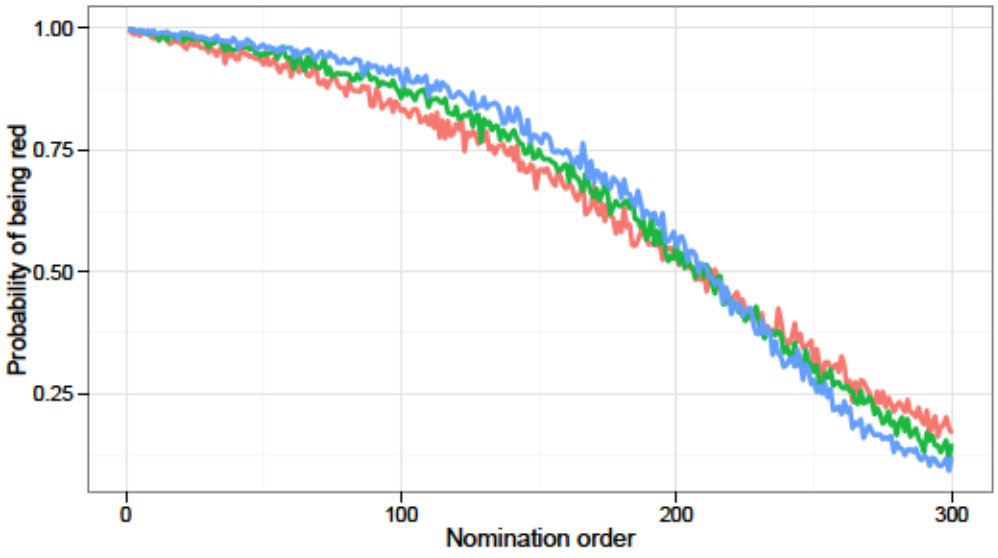


Figure 5.5: Probability of the vertices appearing in the nomination list of being in the block of interest. Higher numbers for low values on the abscissa are better. Only 200 vertices are in the block of interest. Color indicates the level of partial supervision:
red: 0 partially supervised vertices
green: 40 partially supervised vertices
blue: 100 partially supervised vertices

We downloaded the original data from [57], who curated the scrapped data of [40] to include users in the top 5000 "highest quality" communities on the popular cat video website **Youtube.com**. By considering users as nodes and their friendships as links, we can form an undirected graph with up to 1134890 nodes and 2987624 edges. We trim the full graph to only include users from the top 5000 communities, resulting in 15151 nodes. We further trim the graph to include only the top 5000 nodes of highest degree.

We now perform two final operations that trim down the number of nodes and

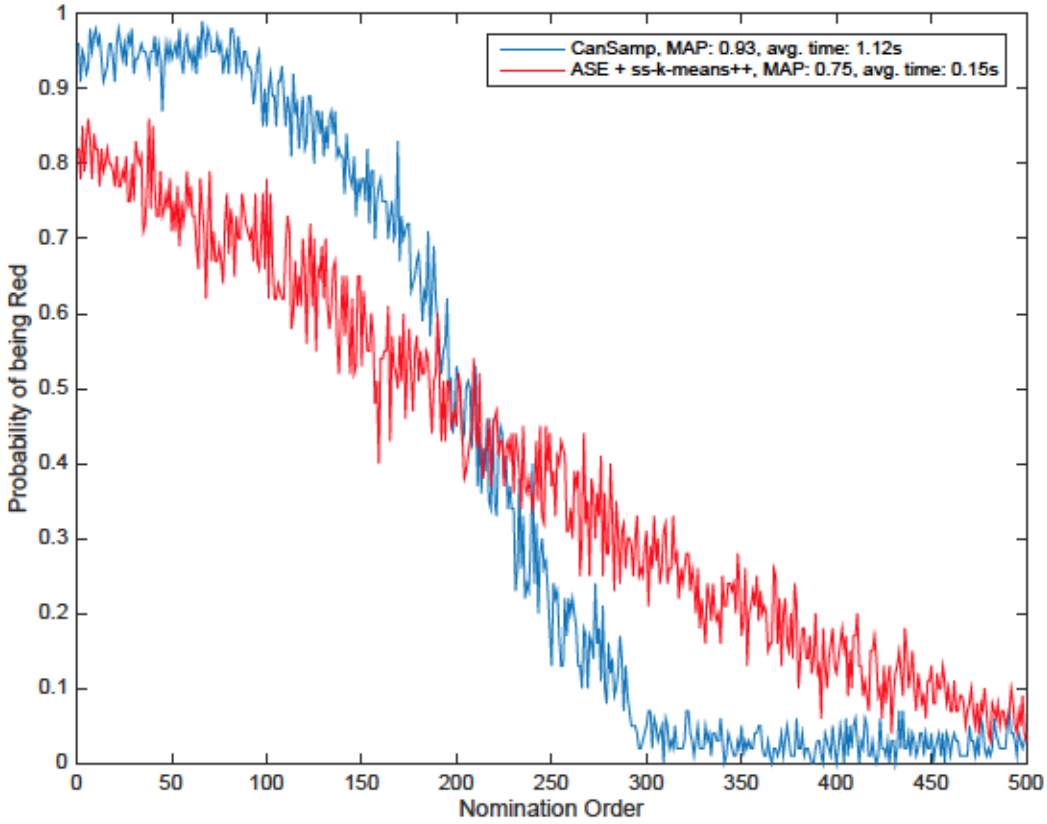


Figure 5.6: Probability of the vertices appearing in the nomination list of being in the block of interest. Higher numbers for low values on the abscissa are better. Only 200 vertices are in the block of interest. We provided no supervision of the type "known not red."

edges significantly. Due to this, we call this dataset YoutubeEasy. Next, we delete nodes that are part of communities of size less than 20. Finally, the nodes of 0 degree in the resulting graph are dropped (since thousands of nodes are deleted, their edges contribute to some of those in the top 5000). The final graph is of size 420 nodes and 422 edges.

We tabulate node membership in the top 5000 communities; community "46" was the largest, with 62 members. We deem this the red community, and all nodes with

membership in the red community are red. Every other node is blue.

We take 20 red members and 20 blue members as seeds. With these "ground truth" labels, we can then proceed to apply canonical sampling and `ssVN`. Due to how sparse the graph was, we often sampled seeds that could not estimate the inter-block link probabilities. Thus, we used the empirical Λ from the ground truth labels. This is unrealistic, but so is how sparse we made this graph. As a final detail, noting that the canonical sampling scheme is a MCMC method in which we can set the burn-in and/or number of samples, we estimated the number of samples and burn-ins such that the run-times (reported in seconds) are approximately equal for both methods (samples=burn-ins=3.5×10^4).

With 100 Monte Carlo replicates using uniformly random seeds, we see that `ssVN` outperforms canonical sampling in terms of mean average precision (cf. 5.7). We observed that one seed in particular, the vertex of the highest degree, is important to include. Thus, we repeated the experiment conditioning on this vertex being included always as a seed and once again never being included as a seed (cf. Figures 5.8 and 5.9). Remarkably, `ssVN` always does well, but canonical sampling is more dependent on the seeding.

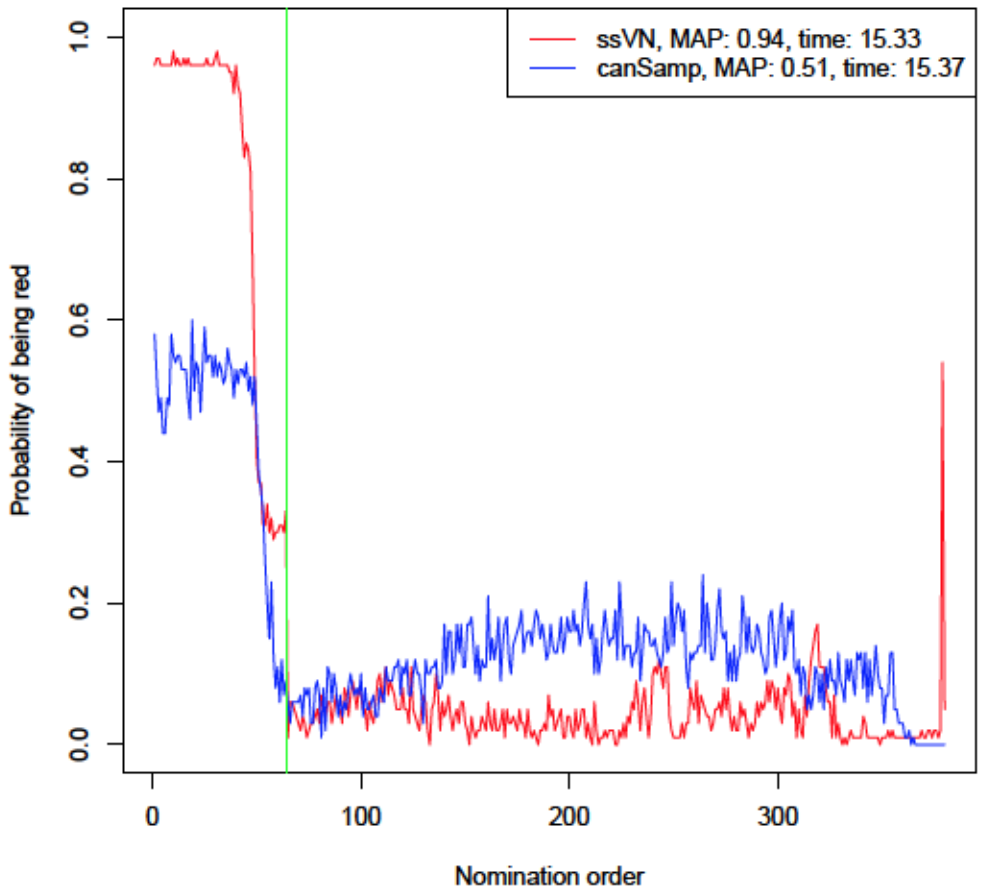


Figure 5.7: Average probability of being red for each position in the nomination list for YoutubeEasy data. Seeds are random.

5.3 Conclusions

In this chapter, we present two interesting applications to perform semi-supervised clustering in. First, we identified behaviotypes in a fly larvae dataset earlier than a non-supervised method. Meaningful behavioral groups as derived from a similar method are used for supervision information, and more finely detailed groups are

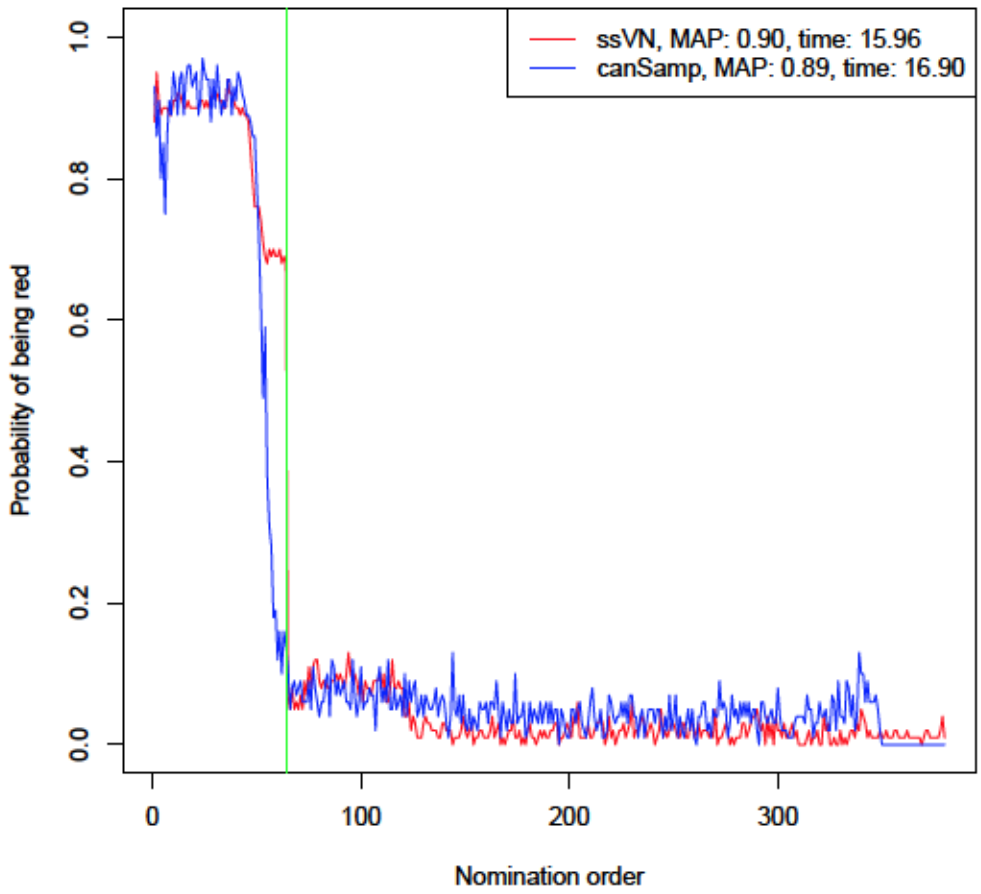


Figure 5.8: Average probability of being red for each position in the nomination list for YoutubeEasy data. Seeds must include vertex of highest degree.

discovered subsequently.

Second, we briefly introduce the topic of vertex nomination and perform a simulation study on a relatively challenging generative distribution. We demonstrate how an additional, weaker type of supervision can help to identify members of the block of interest. Also, we showed results from two similar methods; canonical clustering is superior in accuracy with good seeding and worse with bad seeding. ASE

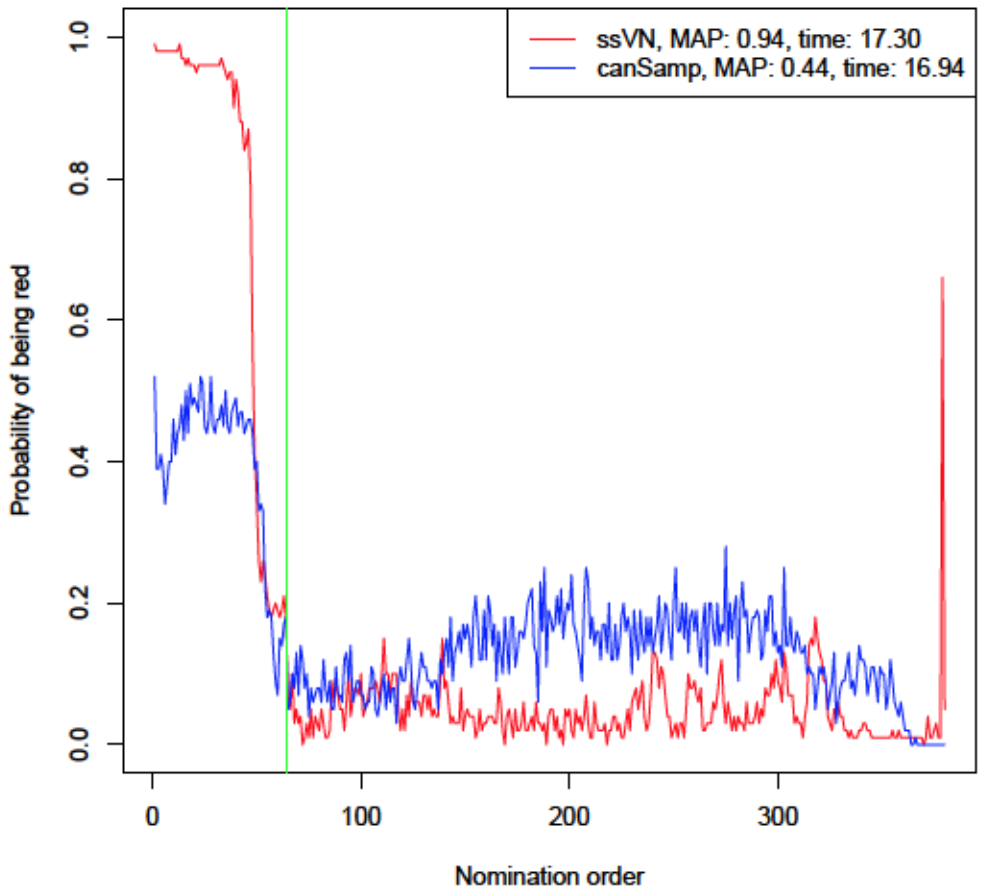


Figure 5.9: Average probability of being red for each position in the nomination list for YoutubeEasy data. Seeds never include vertex of highest degree.

+ `ss-k-means++` is strictly worse in accuracy (albeit faster as it does not involve parameter estimation from GMM).

Chapter 6

Conclusions

Pretty good is great!

Carey E. Priebe

In this final chapter, we summarize the entire thesis and suggest some future work.

6.0.1 Summary

This dissertation focuses on the general problem of semi-supervised clustering, where the goal is to partition a collection of objects into groups using some exogenous information regarding to which groups some of those objects belong. In Chapter 1, we begin by briefly explaining this problem's relationship to the other common machine learning tasks of classification, semi-supervised learning, and clustering.

After explicating the role of the problem in the field, we gave some gentle introductions to the common clustering algorithms of hierarchical clustering, `k-means`, and

model-based clustering in Chapter 2. We expounded upon the most famous modifications other researchers have made to these algorithms to handle the semi-supervised case.

In Chapter 3, we focus on model-based semi-supervised clustering. We re-derive the complete log-likelihood, E-step, and M-step for semi-supervised Gaussian Mixture Modeling, which has previously been done in the literature. We then give a derivation of a modified BIC that does not penalize for supervised data, possibly allowing for additional complexity to be chosen. We note that this derivation was $O(1)$ equivalent to the standard BIC and discussed how such a difference can matter in the finite sample case. We gave some general analysis of the probabilities and followed this with an illustrative toy example where we can tweak several parameters to exhibit that a $O(1)$ difference in information criteria can lead to mistakes with varying probabilities. Finally, we tie our analysis of $O(1)$ differences in information criteria to semi-supervised clustering through a simulation study, where we demonstrate that choosing to penalize less leads to statistically significantly improved performance over the standard BIC in our example.

In Chapter 2, we note that hierarchical clustering does not lend itself to precisely our setting; for this reason, we do not use it for any of our experiments despite it being used in the software upon which our R package, `ssClust`, is based (i.e. `Mclust`) for initialization purposes. Thus, we require another algorithm to warmstart our semi-supervised model-based clustering implementation. For this reason, in Chap-

ter 4, we introduce the semi-supervised k-means++ (`ss-k-means++`) algorithm and compute theoretical bounds on its optimality under certain supervision assumptions. We include some experiments comparing the performance of (`ss-k-means++`) to some similar semi-supervised k-means algorithms and noted improved performance under a variety of metrics that scaled with the level of supervision.

Armed with our initialization strategy (i.e. using the labels from `ss-k-means++`), we are finally prepared for applications tying the whole method together. In Chapter 5, we begin with a clustering problem dealing with behavioral groups in fly larvae induced by different optogenetic lobotimizations arising in a recent Science paper. We use some supervision derived from their hierarchical clustering and observe that we can detect differences in lines faster than the non-supervised method. Further, we are able to reconstruct line information relatively well for this challenging problem.

Finally, we introduce the general problem of vertex nomination of interesting vertices in random graphs as another practical application. Using the literature to justify our feature generation and clustering strategies, we lay out a procedure for vertex nomination. We use the simulation scheme of a recent paper on vertex nomination, which includes several competing strategies, in order to compare our empirical results with theirs. We do pretty well and are pretty fast.

6.0.2 Future Work

Pairwise constraints are sometimes used instead of hard label information. In fact, pairwise constraints can completely specify the latter. Thus, moving our semi-supervised setting to use pairwise constraints would make it more general. If we were to adapt our algorithms to this more general setting, we would be able to apply it to richer datasets with a larger variety of supervised information.

Our analysis in Chapter 4 is based on the seminal work on randomized D^2 weighting k-means algorithms; since then, there have been advancements to these algorithms that improve performance and/or scalability substantially. It seems likely that analogous results for the semi-supervised case could be derived in the semi-supervised setting.

Finally, we have not, but would like to characterize when our semi-supervised vertex nomination procedure is better than the competing methods in the literature, as performance appears to vary based on the signal in the graph and number of vertices. In particular, the "gold standard" method of canonical sampling as analyzed in [20] is very slow, but can naturally be halted early. By varying allowable run-time, we could compare practical implementations of canonical sampling to our method on more interesting datasets.

Bibliography

[1] Method of the year 2010. *Nat Meth*, 8(1):1–1, 01 2011. URL `http://dx.doi.org/10.1038/nmeth.f.321`.

[2] A. W. Bowman A. Azzalini. A look at some data on the old faithful geyser. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(3):357–365, 1990. ISSN 00359254, 14679876. URL `http://www.jstor.org/stable/2347385`.

[3] Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, pages 15–28. Springer, 2009.

[4] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In *Advances in Neural Information Processing Systems*, pages 10–18, 2009.

[5] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on*

Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[6] Avanti Athreya, Vince Lyzinski, David J Marchette, Carey E Priebe, Daniel L Sussman, and Minh Tang. A central limit theorem for scaled eigenvectors of random dot product graphs. *arXiv preprint arXiv:1305.7388*, 2013.

[7] Korinna Bade and Andreas Nürnberger. Creating a cluster hierarchy under constraints of a partially known hierarchy. In *SDM*, volume 8, pages 13–24. SIAM, 2008.

[8] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7): 622–633, 2012.

[9] Sugato Basu, Arindam Banerjee, and Raymond Mooney. Semi-supervised clustering by seeding. In *Proceedings of 19th International Conference on Machine Learning*. Citeseer, 2002.

[10] Sugato Basu, Mikhail Bilenko, and Raymond J Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM, 2004.

BIBLIOGRAPHY

[11] Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics*, volume I. Prentice-Hall, Inc., 2001.

[12] DE Boekee and HH Buss. Order estimation of autoregressive models. In *Proceedings of the 4th Aachen colloquium: Theory and application of signal processing*, pages 126–130, 1981.

[13] Gilles Celeux and Gérard Govaert. Gaussian parsimonious clustering models. *Pattern recognition*, 28(5):781–793, 1995.

[14] Glen Coppersmith. Vertex nomination. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(2):144–153, 2014.

[15] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[16] Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.

[17] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

[18] Thomas Finley and Thorsten Joachims. Supervised k-means clustering. Technical Report 1813-11621, Cornell University, 2008.

[19] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[20] D. E. Fishkind, V. Lyzinski, H. Pao, L. Chen, and C. E. Priebe. Vertex nomination schemes for membership prediction. *Ann. Appl. Stat.*, 9(3):1510–1532, 09 2015. doi: 10.1214/15-AOAS834. URL `http://dx.doi.org/10.1214/15-AOAS834`.

[21] Donniell E Fishkind, Daniel L Sussman, Minh Tang, Joshua T Vogelstein, and Carey E Priebe. Consistent adjacency-spectral partitioning for the stochastic block model when the model parameters are unknown. *SIAM Journal on Matrix Analysis and Applications*, 34(1):23–39, 2013.

[22] Edward Forgey. Cluster analysis of multivariate data: Efficiency vs. interpretability of classification. *Biometrics*, 21(3):768–769, 1965.

[23] Chris Fraley. Algorithms for model-based gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, 1998.

[24] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97 (458):611–631, 2002.

[25] Chris Fraley, Adrian E. Raftery, T. Brendan Murphy, and Luca Scrucca. mclust version 4 for r: Normal mixture modeling for model-based clustering, classifica-

tion, and density estimation. Technical Report 597, University of Washington, Department of Statistics, June 2012.

[26] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[27] Harry Joe. Generating random correlation matrices based on partial correlations. *Journal of Multivariate Analysis*, 97(10):2177–2189, 2006.

[28] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515, 2004.

[29] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pages 3–24, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press. ISBN 978-1-58603-780-2. URL `http://dl.acm.org/citation.cfm?id=1566770.1566773`.

[30] Nam H Lee, Jordan Yoder, Minh Tang, and Carey E Priebe. On latent position inference from doubly stochastic messaging activities. *Multiscale Modeling & Simulation*, 11(3):683–718, 2013.

[31] Stuart P Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[32] Vince Lyzinski, Daniel L. Sussman, Minh Tang, Avanti Athreya, and Carey E. Priebe. Perfect clustering for stochastic blockmodel graphs via adjacency spectral embedding. *Electron. J. Statist.*, 8(2):2905–2922, 2014. doi: 10.1214/14-EJS978. URL `http://dx.doi.org/10.1214/14-EJS978`.

[33] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[34] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. *Theoretical Computer Science*, 442:13–21, 2012.

[35] Cathy Maugis, Gilles Celeux, and M-L Martin-Magniette. Variable selection in model-based clustering: A general variable role modeling. *Computational Statistics & Data Analysis*, 53(11):3872–3882, 2009.

[36] Cathy Maugis, Gilles Celeux, and Marie-Laure Martin-Magniette. Variable selection for clustering with gaussian mixture models. *Biometrics*, 65(3):701–709, 2009.

[37] Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, New York, 2004.

[38] Donald R McNeil. *Interactive data analysis: a practical primer*. John Wiley & Sons, 1977.

[39] Jonathan Mellon, Jordan Yoder, and Daniel Evans. Undermining and strengthening social networks through network modification. *arXiv preprint arXiv:602.06461*, 2016.

[40] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proceedings of the 5th ACM/Usenix Internet Measurement Conference (IMC'07)*, San Diego, CA, October 2007.

[41] Thomas Brendan Murphy, Nema Dean, and Adrian E Raftery. Variable selection and updating in model-based discriminant analysis for high dimensional data with food authenticity applications. *The annals of applied statistics*, 4(1):396, 2010.

[42] Alex Pothen, Horst D Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM journal on matrix analysis and applications*, 11(3):430–452, 1990.

[43] Kevin Pouslen. How a math genius hacked okcupid to find true love, January 2014. URL `http://www.wired.com/2014/01/how-to-hack-okcupid/`.

[44] Carey E. Priebe. Olfactory classification via interpoint distance analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):404–413, 2001.

[45] Carey E Priebe, David J Marchette, and Dennis M Healy Jr. Integrated sensing

and processing decision trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):699–708, 2004.

[46] Adrian E Raftery and Nema Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101(473):168–178, 2006.

[47] Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, pages 1878–1915, 2011.

[48] Noam Shental, Aharon Bar-Hillel, Tomer Hertz, and Daphna Weinshall. Computing gaussian mixture models with em using side-information. In *Proc. of workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, 2003.

[49] Daniel L Sussman, Minh Tang, Donniell E Fishkind, and Carey E Priebe. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107(499):1119–1128, 2012.

[50] Daniel L Sussman, Minh Tang, and Carey E Priebe. Consistent latent position estimation and vertex classification for random dot product graphs. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(1):48–57, 2014.

[51] Minh Tang, Daniel L Sussman, Carey E Priebe, et al. Universally consistent

vertex classification for latent positions graphs. *The Annals of Statistics*, 41(3): 1406–1430, 2013.

[52] Joshua T Vogelstein, Youngser Park, Tomoko Ohyama, Rex A Kerr, James W Truman, Carey E Priebe, and Marta Zlatic. Discovery of brainwide neural-behavioral maps via multiscale unsupervised structure learning. *Science*, 344 (6182):386–392, 2014.

[53] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[54] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, volume 1, pages 577–584, 2001.

[55] Wikipedia. Iris flower data set — wikipedia, the free encyclopedia, 12 2015. URL `https://en.wikipedia.org/w/index.php?title=Iris_flower_data_set&oldid=678872226`.

[56] Daniela M Witten and Robert Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490), 2010.

[57] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, 2015.

[58] Jordan Yoder and Carey E Priebe. A model-based semi-supervised clustering methodology. *arXiv preprint arXiv:1412.4841*, 2014.

[59] Jordan Yoder and Carey E Priebe. Semi-supervised k-means++. *arXiv preprint arXiv:1602.00360*, 2016.

[60] Li Zheng and Tao Li. Semi-supervised hierarchical clustering. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 982–991. IEEE, 2011.

[61] Mu Zhu and Ali Ghodsi. Automatic dimensionality selection from the scree plot via the use of profile likelihood. *Computational Statistics & Data Analysis*, 51 (2):918–930, 2006.

Vita

Jordan Yoder was born in Nashville, TN. At the time, he did not like country music, but does now. He attended Hillsboro Comprehensive High School until graduating in 2005, at which point he moved to Baltimore, MD to get his Mathematics B.A. degree from Goucher College. He graduated with honors and *summa cum laude* in 2009. Next, he worked as an actuary for GEICO in Chevy Chase, MD until 2011, when he began attending the graduate school Johns Hopkins University in Baltimore, MD in the department of Applied Mathematics and Statistics. He recently placed third in a tractor tournament.