Clustering In Many Dimensions: Comparison Between Subspace Clustering And A Baseline Clustering Algorithm

Rachael S. Shudde
Department of Statistics
Texas A&M University
College Station, TX 77840
rachaelshudde@tamu.edu

Kristyn Pantoja
Department of Statistics
Texas A&M University
College Station, TX 77840
kristynp@stat.tamu.edu

Salih Kilicli
Department of Mathematics
Texas A&M University
College Station, TX 77840
salihkilicli@math.tamu.edu

Antoni Virós i Martin
Department of Aerospace Engineering
Texas A&M University
College Station, TX 77840
aviros@tamu.edu

Abstract

Current unsupervised machine learning techniques are ill-suited for problems in high dimensions for many reasons. In this paper, we will focus on clustering techniques and what causes them to fail in high dimensions. Problems in high dimensions often include many features that are irrelevant or redundant that may mask clusters spread across many subspaces of the data. Additionally, data points in high dimensions often become equally spaced, so using normal methods of measuring distance often fail to find clusters. Problems like these present the need for clustering across various subspaces of data. This paper will review CLIQUE, a clustering method for high dimensional data, and compare it to the better known k-means method.

1 Review of current subspace clustering methods

Clustering methods are increasingly being used to find patterns and clusters in high dimension data. Clustering on real world data is an extremely difficult problem, and current methods prefer dimension reduction and feature selection. CLIQUE, the flagship subspace clustering algorithm, tries to choose subsets of dimensions where clusters appear, while k-means, on the other hand, does not rely on dimension reduction techniques, though it is not uncommon to pair it with PCA or other dimension reduction methods.

This paper will examine two questions: when is subspace clustering necessary, and how does subspace clustering perform compared to the more popular k-means method?

1.1 Background: Need for subspace clustering

The development of subspace clustering methods was motivated by the desire to overcome the curse of dimensionality. Distance-based methods such as k-means work well when the dimension of the data is small. However, as the dimension of the data grows, the data points start to become equidistant from each other, rendering such methods less effective. This idea is well illustrated from an example on simulated data from [5], the original CLIQUE publication. The data in Figure 1 is spread across

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

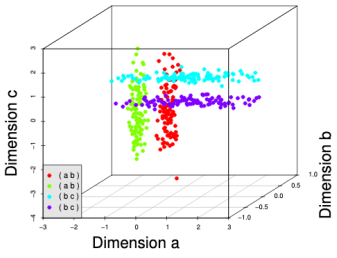
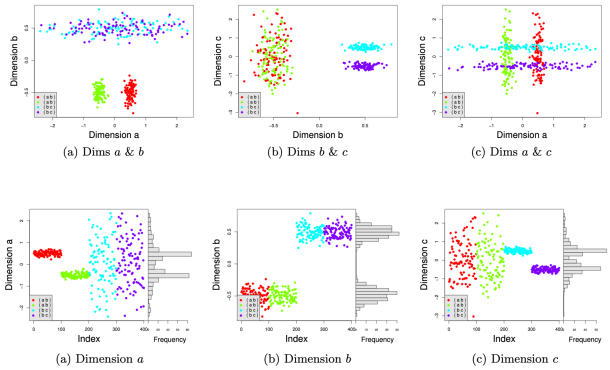


Figure 1: Sample dataset [5]



(a) Dims a & b (b) Dims b & c (c) Dims a & c

(a) Dimension a (b) Dimension b (c) Dimension c

Figure 2: Data from Figure 1 plotted in two dimensions (top) and one dimension (bottom) [5]

three dimensions, with each cluster spread across two dimensions. However, when you remove a dimension, as in Figure 2, it is clear that the clusters start to become mixed. Simply trying to cluster based on the full data space or a reduced data space would be insufficient for this data.

1.2 Subspace clustering review

Subspace clustering methods seek to solve this problem by looking at independent subspaces of the data. There are two main methods of unsupervised subspace clustering, top-down and bottom-up. Top-down methods try to find clusters in a higher dimension, and prune down dimensions. Bottom-up methods try to find clusters in each dimension, and combine dimensions one at a time looking for new clusters. CLIQUE is one of the earliest bottom-up search methods.

1.3 CLIQUE

CLIQUE [1] is a subspace clustering algorithm that generates clusters of maximum dimensionality in the form of Disjunctive Normal Forms (DNF) without any assumption on data ordering or distribution.

Before explaining the algorithm, there are some important definitions from [1] that we need to explain so the algorithm can be understood. First, the authors define *units* to be partitions of the d-dimensional space of the inputs into ξ intervals of equal length in 2D, or equal regions in higher dimensions. Then, they define a *dense* unit as one that contains a fraction of the total data bigger than some integer τ. Both ξ and τ are the input tuning parameters of the algorithm. Units can also be generalized to all the subspaces in the problem. Finally, they define *clusters* as maximal sets of connected dense k-dimensional units, where $k \leq d$. Two units are *connected* if they have a common face or if there is a set of units that are connected pairwise between them.

With all the definitions out of the way, we proceed to describe the algorithm. It performs three operations: 1) Identification of subspaces that contain clusters. 2) Identification of clusters. 3) Generation of a minimal description for the clusters.

The first operation uses a bottom-up approach to finding subspaces with clusters that exploits the following lemma: A k-dimensional cluster is also a cluster in any $k-1$ projection of itself. The algorithm starts with finding the set of all dense 1-dimensional units. Then, to find the 2-dimensional dense units that can be part of a cluster (or more generally, k-dimensional from $(k-1)$-dimensional), the algorithm proceeds by finding a superset of all k-dimensional dense units by joining all dense $(k-1)$-dimensional units that share the first $k-2$ dimensions. Once these are computed, the k-dimensional units with a $k-1$ projection not included in the previous dataset are removed. This last step is supported by the clustering lemma we described, as one derived property is that all k-dimensional dense units must have dense $k-1$-dimension projections. The complexity of this part is $O(c^k + mk)$, where k is the maximum dimensionality of all the dense units, m is the number of input points, and c is a constant.

Table 1: DNA microarray data contingency table, k-means clusters and cancer labels

label Prediction	BREAST	CNS	COLON	K562	LEUKEMIA	MCF7	MELANOMA
0	2	0	7	2	6	2	0
1	2	0	0	0	0	0	7
2	3	5	0	0	0	0	1

label Prediction	NSCLC	OVARIAN	PROSTATE	RENAL	UNKNOWN
0	0	0	0	0	0
1	0	0	0	0	0
2	9	6	2	9	1

The second operation is finding the clusters in the set of dense units. The way this works is by finding which units that share the dimension set are connected to each other. The authors describe it as a connected components on a graph problem, where each unit is a vertex, and there is an edge between units iff they have a common face. Thus, each connected component of this graph will be a cluster. The users use a depth-first approach to finding those, which mean the complexity of this second step is $O(2kn)$ for each subspace k, with n being the number of units in that subspace.

The third and last operation corresponds to finding minimal descriptions of the clusters. The authors mention finding the optimal cover for a set is a NP-hard problem, even for $d = 2$. Finding a cover is equivalent to finding a set of rectangles that hover over all of the cluster. Thus, they detail a greedy approximation that covers the whole cluster with maximal regions, where a region is a rectangular set of units. Once that is found, they perform a second step that minimizes the number of maximal regions such that the final set still covers the whole cluster. This gives an approximation on the order of $n \ln n$ of the optimal NP-hard solution where n is the minimum number of regions needed to cover the cluster.

2 K-means

K-means [2] is a very popular clustering algorithm that attempts to minimize the difference between each data point $x_1, \ldots, x_k$ and a set of means $m_k, k = 1, \ldots, K$, and puts each data point in cluster k based on the minimization. K-means works better in smaller dimensions, where minimizing the Euclidean distance is more applicable, so k-means on PCA data usually gives better results than k-means on the entire dataset, although the results are often less useful for real-world data.

3 Analysis on CLIQUE and k-means on three data sets

We will analyze k-means and CLIQUE on three different data sets. The submitted code markdown file has some more detailed explanation of each case. The first data set is a simulated data set from the CLIQUE python library documentation [4], while the second and third data sets are from real data: the first one is the Titanic dataset from the Kaggle competition [3] and the second one is the human tumor DNA microarray data, used in the k-means example in Elements of Statistical learning, Chapter 14, Section 14.3.8 Example: Human Tumor Microarray Data [2].

3.1 K-means and CLIQUE applied to the DNA microarray data

The DNA microarray data for human tumors has gene expressions containing 6830 genes for 64 samples. The targets are labels for the different human tumors exhibited in this sample. By applying k-means with $k = 3$, we can learn "which samples are most similar to each other in terms of their expression profiles across genes", the first question posed for this data set when it is introduced in Chapter 1 of Elements of Statistical learning [2].

Similar to the textbook, when we apply k-means to this data and create a contingency table 1 for the prediction given by k-means and the label, we observe that "the procedure is successful at grouping

Table 2: Confusion Matrix for Titanic, k-means (left) and CLIQUE (right)

label Prediction	0	1
0	416	264
1	8	26

label Prediction	0	1
0	423	290
1	1	0

together samples of the same cancer. In fact, the two breast cancers in the second cluster were later found to be misdiagnosed and were melanomas that had metastasized" (p.514) [2].

As it turns out, when we apply CLIQUE to the DNA microarray data, the algorithm does not detect any clusters unless interval size is equal to 1, resulting in one large cluster. The data is too high-dimensional for this particular bottoms-up subspace clustering algorithm to find clusters in the original, large space. We turn to another data set, the Titanic data set, which is not so high dimensional, in order to perform a more thorough investigation in the comparison of k-means and CLIQUE.

3.2 K-means and CLIQUE applied to Titanic data

The Titanic data set from [3] is composed of 12 variables: the features PassengerId (identifier for each passenger on the Titanic), Pclass (ticket class, categorical in $\{1, 2, 3\}$ for 1st class, 2nd class, and 3rd class respectively), Name, Sex (binary in $\{0, 1\}$ for male and female), Age, SibSp (number of siblings/spouses aboard), Parch (number of parents/children aboard), Ticket (ticket number), Fare (passenger fare), Cabin (cabin number), Embarked (port of embarkation), and the target Survived (binary in $\{0, 1\}$ for survived and did not survive). The features we use are Pclass, Sex, Age, SibSp, Parch, and Fare. We hope to find two clusters: one corresponding to passengers who survived (Survive = 1), and one corresponding to passengers who did not survive (Survive = 1).

When we apply k-means to the dataset, we get the confusion matrix in the left table 2. It is able to create two clusters with a little bit less than 40% misclassification rate, meaning the results are at least sensible.

Applying CLIQUE to the dataset and choosing interval size 2 in order to obtain two clusters with interpretable results, we get a misclassification rate of about 40% as well. However, the results are worse when we apply CLIQUE because it shows that all persons aboard the titanic were placed in one cluster, except for one, which is its own cluster. When we tried to resolve this issue by raising the threshold from 0 to 1, i.e. allowing for outliers, CLIQUE classified a large portion of the observations as outliers. We suspect that CLIQUE's performance may be doomed to suffer in this particular example because we are looking for a specific number of clusters, which is not consistent with the motivation for CLIQUE: to find all clusters that exist. In this sense, it is possible that CLIQUE is more amenable to data segmentation tasks than label prediction tasks.

3.3 K-means and CLIQUE applied to CLIQUE example simulated data

This data set provided by the CLIQUE library documentation is an example illustrating CLIQUE's advantage in being able to detect a variety of cluster topologies. This dataset is composed of six clusters in two dimensions: one circle in the middle of the support, a ring surrounding that circle, and four small clusters in the four corners. We apply k-means with $k = 6$ and CLIQUE with interval size 10 and threshold 0 (i.e. no noise/outliers).

From figure 3, we can tell that the k-means clusters did not detect the six clusters that the data describes, but instead splits the ring-shaped cluster into five pieces and places the four corner clusters in one of these five clusters.

CLIQUE, on the other hand, was able to find all six clusters, as is shown by 4. We can see from figure 5 that it is able to do this by finding which of the 10^2 units (in the grid created by an interval size 10 over a 2-dimensional support) are dense units.

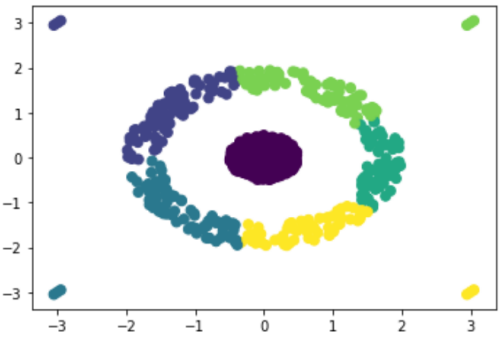


Figure 3: k-means clusters

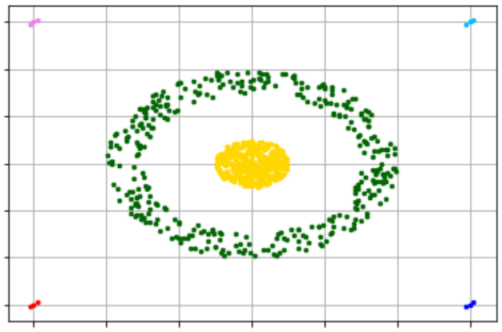


Figure 4: CLIQUE clusters

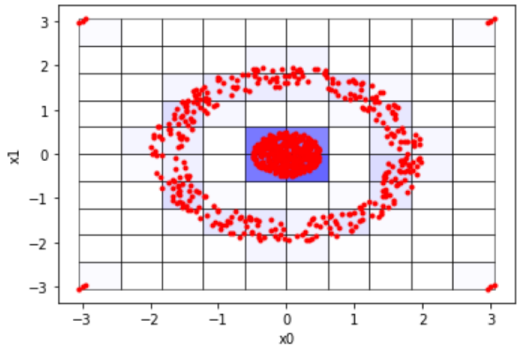


Figure 5: CLIQUE grid with interval = 10

4 Conclusion

While CLIQUE exhibited superior performance in the simulated CLIQUE example data set, it seems from our analysis that k-means performs better in higher dimensions.

From our first example application using the CLIQUE example simulated data set, we can see that CLIQUE is great at detecting clusters with unusual topologies. K-means, on the other hand, had difficulty finding the six clusters that were apparent to the human eye.

It is also worth noting that CLIQUE does not do as well as advertised in higher dimensional data because its definition of a cluster as a set of adjacent dense units still implicitly relies on Euclidean distance: dense units cannot form a cluster unless they are close (adjacent) to each other. We see this in figure 5 in how the clusters are determined in the two-dimensional space. In higher dimensions, we see this when we notice that having an interval size larger than 2 leads to fewer clusters being detected, such as in the DNA microarray data.

Overall, unsupervised clustering in high dimensions - subspace and k-means - is a difficult problem to solve. Some methods are better suited for different types of data, and different values of the tuning parameter can drastically change results. k-means and CLIQUE are both powerful methods, but both should be used carefully and with full knowledge of their weaknesses.

References

[1] Rakesh Agrawal et al. *Automatic subspace clustering of high dimensional data for data mining applications*. Vol. 27. 2. ACM, 1998.

[2] Trevor Hastie et al. "The elements of statistical learning: data mining, inference and prediction". In: *The Mathematical Intelligencer* 27.2 (2005), pp. 83–85.

[3] Kaggle. *Titanic: Machine Learning from Disaster*. URL: https://www.kaggle.com/c/titanic/overview.

[4] Andrei Novikov. "PyClustering: Data Mining Library". In: *Journal of Open Source Software* 4.36 (Apr. 2019), p. 1230. DOI: 10.21105/joss.01230. URL: https://doi.org/10.21105/joss.01230.

[5] Lance Parsons, Ehtesham Haque, and Huan Liu. "Subspace clustering for high dimensional data: a review". In: *Acm Sigkdd Explorations Newsletter* 6.1 (2004), pp. 90–105.