Fall 2017

Final Project Report: Hubness and Some Questions

Yimeng Li and Vega Bharadwaj *CS 584 Data Mining*

1 Introduction

The number of k-occurences of point x from dataset D is called its hubness score. High hubness score indicates that this point is close to plenty of other points in the dataset. In a regular clustering algorithm, e.g. K-Means, a cluster medoid is also the point closest to all the points in the cluster. Since both of them, a cluster medoid and a high hubness point, share the same attribute, we believe it's highly possible that high hubness points are more likely to become cluster medoids comparing to points with a lower hubness score. In this project, we unveil the relationship between high hubness points and cluster medoids on both synthetic and real-world datasets.
Besides, low hubness scores also indicates that a point is far away from the majority of its peers, which is big sign of being an outlier or a noise. We happened to learn an algorithm called DBSCAN during the semester, which is quite effective in finding data outliers using density information. The second problem we are trying to solve is to explore the relation between low hubness points and DBSCAN-detected noise points on both synthetic and real-world datasets.

2 Related Work

Hubness has been observed in several fields of applications involving sound and image data during recent years (Aucouturie and Pachet, 2007; Doddington et al., 1998; Hicklin et al., 2005) and, in addition, Jebara et al. Hubness phenomenon in the construction of the neighborhood graph of learning (Tony Jebara et al 2009) [2]. Amina M et al. designed a hubness-based algorithm by introducing hubs into the k-means algorithm (Amina M et al 2015) [3]. Although people does not give much attention to the phenomenon of hubness in data clustering, the k-nearest-neighbor list is widely used in many clusters. The k-nearest-neighbor list computes the density estimate by the volume used to observe the space determined by the k nearest neighbors. Density-based clustering methods generally rely on this density estimation. The main goal of density-based clustering algorithms is to find high-density areas separated by low-density areas [4]. In high-dimensional space, this is often difficult to estimate because data is usually sparse. It is also important to choose the appropriate neighborhood size, because too small or too large k values may cause the density-based approach to fail. The k-nearest-neighbor list is often used to construct k-NN graphs and so it's used in graph clustering.

3 Solutions

To explore the relation between hubs and clusters, we did experiments on both synthetic datasets and real-world datasets. All the experiments are finished using MATLAB.

First part of the experiment is performed on the synthetic datasets. We run K-Means on each gaussian mixture. The K-Means algorithm is initialized by using K-Means++. In each iteration, we measure the distance from each cluster centroid to its closest neighbor and to the strongest hub inside the cluster, and scaled by the average intracluster distance. Both minimal and maximal distance among all the clusters are computed. We run the whole process several times and compute the average minimal and maximal distances to eliminate the noise brought in by K-Means random initialzation. In the end we compare the average minimal distance to a medoid and to a strong hubness point to see if they are the same point. And of course we also compare the maximal distances between them. Second part of the experiment is done on the real-world datasets. Before we running the real algorithms on these datasets, we first do some data preprocessing. For each point in a dataset, we normalize its attributes values across the whole dataset using minmax normalization. Then we go through the same computing process on each real dataset as we do on the gaussian mixtures.
Even though we implement all the experiments by ourselves, it's worth mentioning how we find the strongest hubness point in each cluster. We first call $knnsearch()$ function to find the nearest neighbors of each point in the cluster. Then we call $unique()$ function to pick out points that are the neighbors of other points. $histc()$ function is called to figure out the number of times each point being the nearest neighbor of other points. The point with the largest number is the strongest hubness point in the dataset. To compute the intraclass distance, we call $pdist2()$ function to compute the euclidean distance from each point in the cluster to the centroid. We sum up these distances and divide it by the number points in the cluster to get the intraclass distance. The cluster medoid is found by seeing the point which has the smallest distance to the cluster centroid.
To explore the relation between hubs and DBSCAN outliers, experiments are also finished using both synthetic datasets and real-world datasets.First part of the experiment is done on the synthetic datasets. We run DBSCAN on each gaussian mixture to find the outliers. The eps value, the radius of each core circle, is set up manually by reading the kdist graph of the whole dataset. Then we compute the hubness score of all the points in the dataset. For the DBSCAN detected noise points, we check if their hubness scores are two standard deviations lower than the average hubness value of all the points. Second part of the experiment is done on the real-world datasets. We still need to manually set up the eps value. Then we do the same comparison between the outliers hubness scores and the average hubness value.
How we find the strongest hubness point in the dataset is the same as what we do when we find the relationship between hubs and cluster medoids. The difference is that in the hubs and medoids experiment, we find the strongest hubness point of each cluster. The hubness point is computed by finding each point's nearest neighbors inside the cluster. But in the hubs and DBSCAN noise points experiment, each point's hubness score is decided by each point's nearest neighbors over the whole dataset, rather than just points in the same cluster. To compute the average and standard deviation of the hubness scores, we call MATLAB functions $mean()$ and $std()$ respectively.

4 Experiments

4.1 Data

Two types of data are including synthetic and real-world data sets.
For synthetic datasets, we create 7 random generated gaussian mixtures by using $gmdistribution()$ from MATLAB. All the gaussian mixtures are generated from a given list of dimensions: 2, 5,

10, 20, 30, 50, 100 and are created from 10 Gaussian generators. So each mixture has 10 clusters. When we create the mixtures, the gaussian distribution mean is uniformly chosen from $[lower\_bound^{\mu}, upper\_bound^{\mu}]^{dimension}$, $lower\_bound^{\mu} = -20$, $upper\_bound^{\mu} = 20$ and the standard deviations is also randomly chosen from $[lower\_bound^{\sigma}, upper\_bound^{\sigma}]^{dimension}$, $lower\_bound^{\sigma} = 2$, $upper\_bound^{\sigma} = 5$. And we randomly generate 10000 points for each mixture.
For the real world datasets, we take several datasets from UCI dataset repository. And they are described in an ascending order of their dimension.

- Iris Dataset. It is perhaps the best known database to be found in the pattern recognition literature. It has 150 instances and the data dimension is 4 and it has 3 classes. It can be downloaded from https://archive.ics.uci.edu/ml/datasets/iris.
- Abalone Dataset. The data comes from predicting the age of abalone from physical measurements. It has 4177 instances and the data dimension is 8 and it also has 8 classes. It can be downloaded from https://archive.ics.uci.edu/ml/datasets/abalone.
- Breast Cancer Wisconsin (Prognostic) Dataset. Each record represents follow-up data for one breast cancer case. It has 198 instances and the data dimension is 32 and it has 2 classes. It can be downloaded from https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Prognostic).
- Sonar Dataset. The data is obtained by bouncing sonar signals off a metal cylinder at various angles and under various conditions. It has 208 instances and the data dimension is 60 and it has 2 classes. It can be downloaded from http://archive.ics.uci.edu/ml/datasets/connectionist+bench+(sonar,+mines+vs.+rocks).
- Hill-Valley Data Set. Each record represents 100 points on a two-dimensional graph. When plotted in order as the Y co-ordinate, the points will create either a Hill or a Valley. It has 606 instances and the data dimension is 100 and it has 2 classes. It can be downloaded from http://archive.ics.uci.edu/ml/datasets/hill-valley.

4.2 Experiment Setup

To make sure that the implementation computing hubness scores is correct, we replicate the experiment settings mentioned in [1] to see if I get similar results.
To evaluate our experiments on hubs distribution across clusters, we will draw some plots to show how the maximal distance and minimal distance evolve through iterations. Data dimensions below 10 are used to illustrate low-dimensional behavior while dimensions above 10 are used to illustrate high-dimensional behavior. The neighborhood size is set to 10 across all the datasets.
To set up the DBSCAN *eps* parameter, we read the kdist graph, a graph recording the distance of each point to its 10th nearest neighbor. Table 1 shows the *eps* value of each dataset.

4.3 Experimental Results

To prove that the synthetic datasets are created correctly, we draw out the point distribution of the 2-dim gaussian mixture on Figure 1. It's clear to see that there are 10 clusters there, even though sometimes 2 of them might be very close to each other. It's also worth mentioning that all

Synthetic Datasets	Eps Value	Real Datasets	Eps Value
2-dim gm	1.0	Iris	0.28
5-dim gm	3.2	Abalone	0.17
10-dim gm	5.8	WPBG	1.2
20-dim gm	9.3	Sonar	2.2
30-dim gm	12.0	Hill and Valley	0.8
50-dim gm	16.5		
100-dim gm	24.6		

Table 1: Datasets and DBSCAN *eps* Parameter

the cluster centroids x-coordinates are inside $[-20, 20]$ and y-coordinates are also inside $[-20, 20]$. This is because we set up the gaussian distribution means to be inside $[-20, 20]$. This also proves that our implementation to generate synthetic data is correct.

Let's move on to the first part of the whole project, exploring the relation between hubness points and cluster medoids. Figure 2 shows the results on low-dimensional synthetic datasets. The dimension of the points in these datasets are below or equal to 0. Apparently, both the minimal and maximal distances from the cluster centroids to the cluster medoids and from the cluster centroids to the strongest hubness points are still very different, even though the two lines are getting closer as the dimension increases. This means that for low dimensional synthetic data, the cluster medoid and the strongest hubness point in the cluster are still two different points. However, when it comes to high-dimensional data, things change. Figure 3 shows the results on high-dimensional synthetic data. When the dimension increases to 30, the distance from the cluster centroid to the cluster medoid and the distance from the cluster centroid to the strongest hubness point are the same. This means that under high-dimensional condition, the strongest hubness point is the cluster medoid in a cluster.

Above is the analysis towards synthetic data. When we do experiments using real-world datasets, we get different results. Figure 4 and Figure 5 show the results on both low-dimensional and high-dimensional real world datasets. It's not difficult to find that no matter how the data dimension increases, the distance from cluster centroid to cluster medoid and the distance from cluster centroid to strongest hubness point are still different. This means that the strongest hubness point is not the cluster medoid when it comes to real-world datasets.

Let's move on to the second part of the whole project, exploring the relation between low hubness points and DBSCAN detected noise points. Figure 6 shows the kdist graph we use when we set up the DBSCAN *eps* parameter. The kdist graph shows the number of points whose distance to their 10th nearest neighbors is below a value. We want to make sure that nearly 1% points over the whole dataset will be selected as noise points when we run DBSCAN. The final *eps* value for each dataset is shown by Table 1. Table **??** shows the experiment results on synthetic datasets. When the dimension of data is 2 or 5, there are over 50% of the DBSCAN detected noise points, whose hubness value is two standard deviations below the hubness average over the whole dataset. When the dimension grows to more than 5, the hubness value is no more an indicator of whether a point is an outler. No DBSCAN detected points have hubness values 2 standard deviations below the average. So we can use low hubness value as an efficient indicator to find outliers when we deal with low dimensional synthetic data. Things are very different when we work on real-world datasets. Table 5 shows the experiment results. It turns out when the data dimension is below 100, there are no DBSCAN detected noise points whose hubness value is two standard deviations below the average. When we work on Hill Dataset, whose data dimension is 100, there are 75% noise points whose hubness points satisfy the outlier threshold. It seems that the result on real world datasets is totally the opposite of what we have from synthetic datasets. Considering the difference of dataset size between synthetic datasets and real-world datasets, there are 10,000 points in a synthetic dataset while there are usually below 500 points in a real-world dataset. We can't say that the conclusion we have from synthetic datasets is totally wrong. But we think it's fair to say we can never use low hubness score as an indicator to see if a point is an outlier.

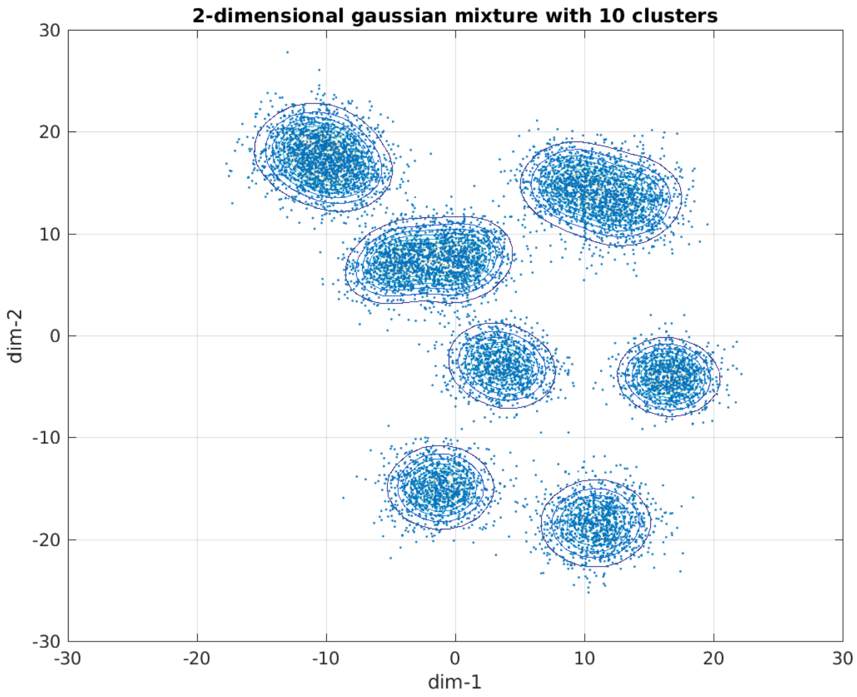


Figure 1: Example Gaussian Mixture

5 Conclusion

6 Contribution

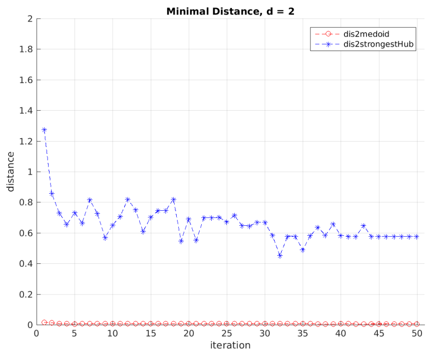
Yimeng Li worked on exploring the relation between hign-hubness points and cluster modoids. He also finished exploring the relation between low-hubness points and DBSCAN-detected noise points. All the things, including slides, experiments and report about these stuff are finished by him.

Synthetic Datasets	N_a of Noise Points	N_b of Noise Points below Average	Percentage $\frac{N_b}{N_a}$
2-dim gm	157 (from 10^4 points)	97	61.78%
5-dim gm	121 (from 10^4 points)	95	78.51%
10-dim gm	120 (from 10^4 points)	0	0%
20-dim gm	121 (from 10^4 points)	0	0%
30-dim gm	129 (from 10^4 points)	0	0%
50-dim gm	110 (from 10^4 points)	0	0%
100-dim gm	117 (from 10^4 points)	0	0%

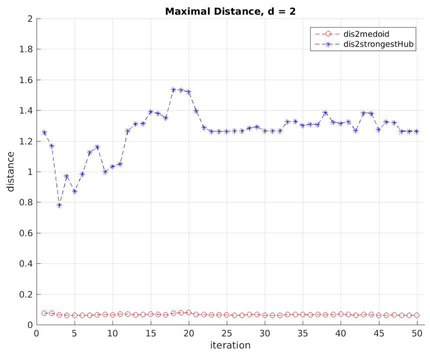
Table 2: Compare DBSCAN noise points hubness score to the average on synthetic datasets

Real Datasets	N_a of Noise Points	N_b of Noise Points below Average	Percentage $\frac{N_b}{N_a}$
Iris	2 (from 150 points)	0	0%
Abalone	29 (from 4177 points)	0	0%
Wpbg	4 (from 198 points)	0	0%
Sonar	5 (from 208 points)	0	0%
Hill	4 (from 606 points)	3	75%

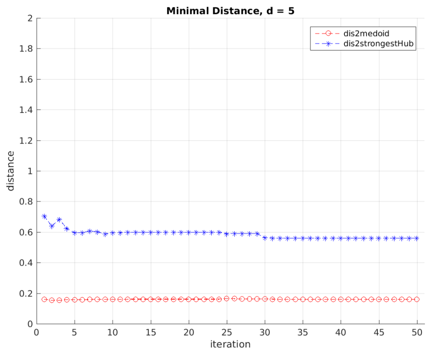
Table 3: Compare DBSCAN noise points hubness score to the average on real datasets



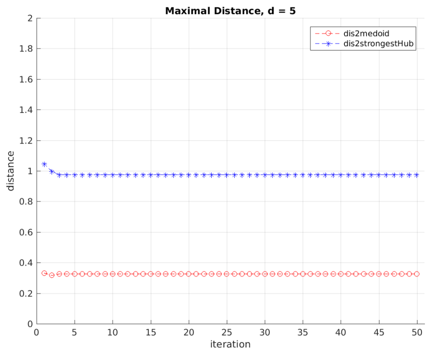
(a) Minimal dis, d = 2



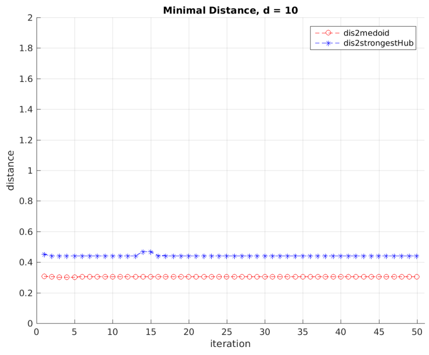
(b) Maximal dis, d = 2



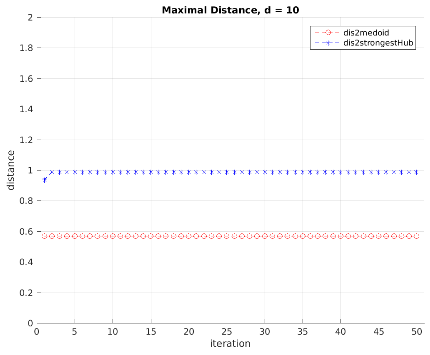
(c) Minimal dis, d = 5



(d) Maximal dis, d = 5

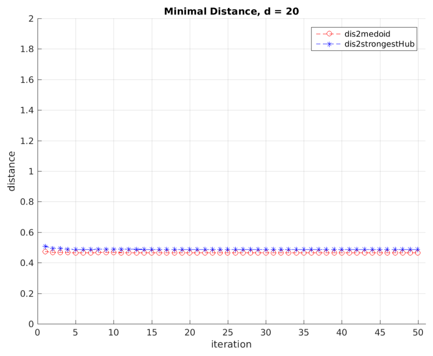


(e) Minimal dis, d = 10

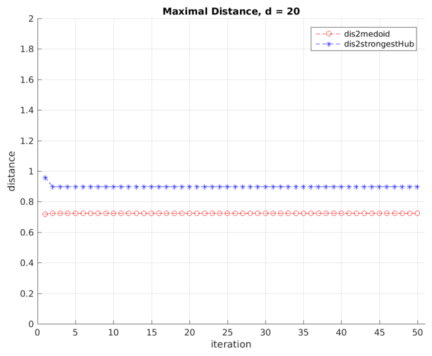


(f) Maximal dis, d = 10

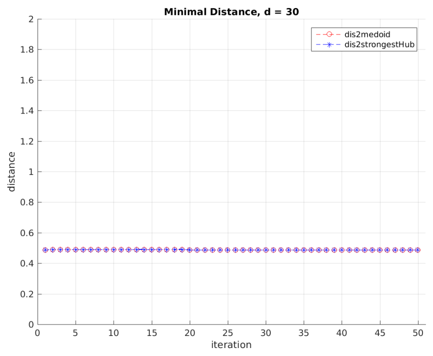
Figure 2: low-dim synthetic data results



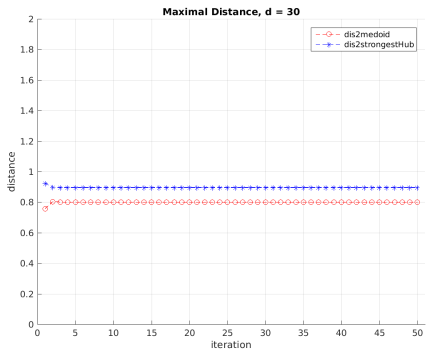
(a) Minimal dis, d = 20



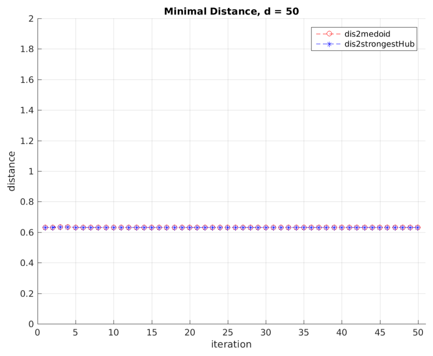
(b) Maximal dis, d = 20



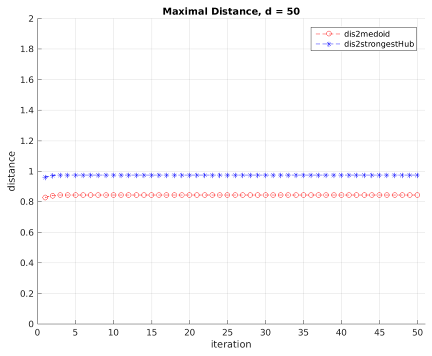
(c) Minimal dis, d = 30



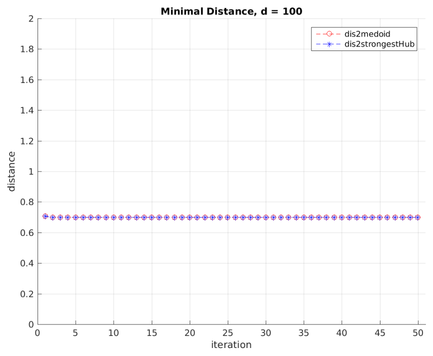
(d) Maximal dis, d = 30



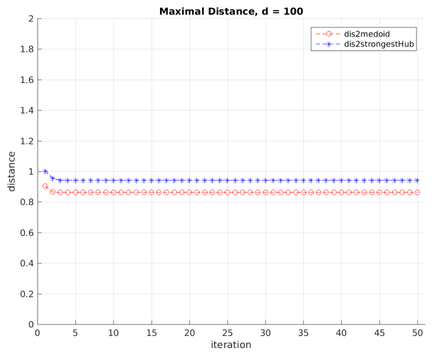
(e) Minimal dis, d = 50



(f) Maximal dis, d = 50

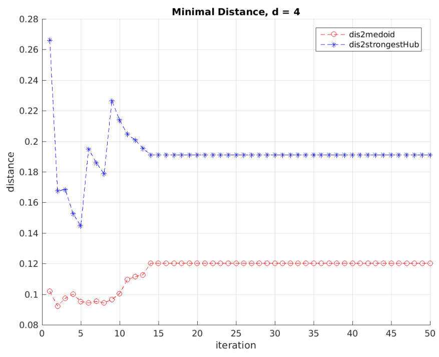


(g) Minimal dis, d = 100

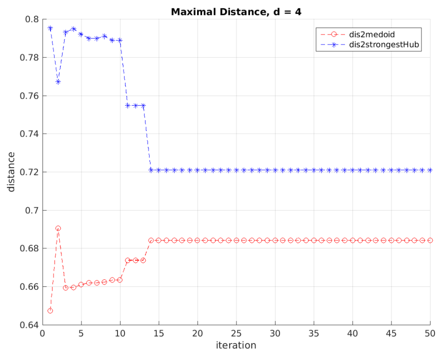


(h) Maximal dis, d = 100

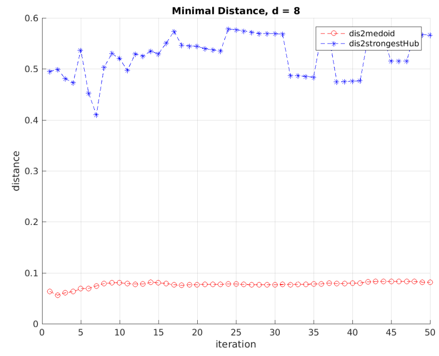
Figure 3: high-dim synthetic data results



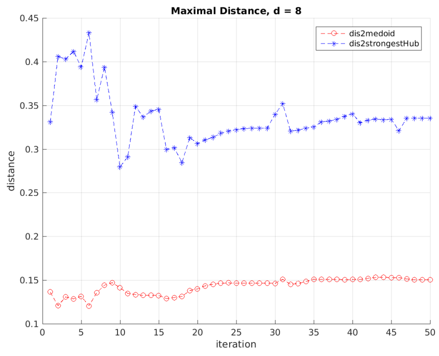
(a) Minimal dis, Iris Dataset



(b) Maximal dis, Iris Dataset

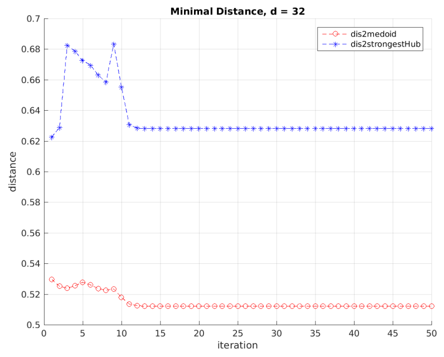


(c) Minimal dis, Abalone Dataset

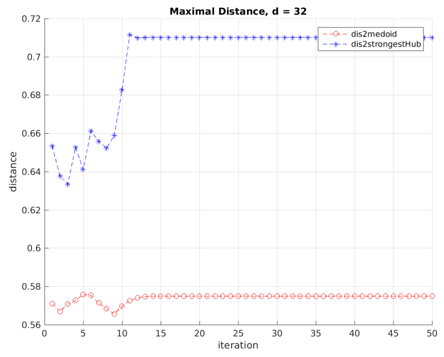


(d) Maximal dis, Abalone Dataset

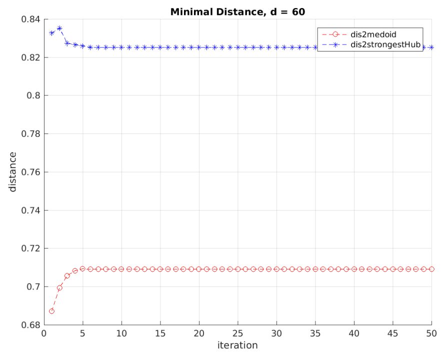
Figure 4: low-dim real data results



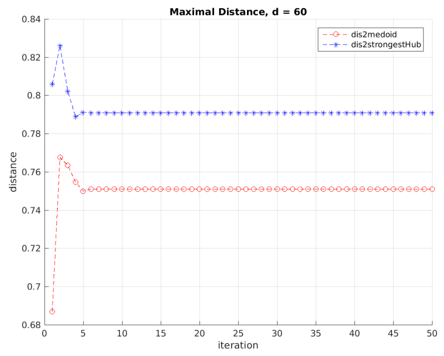
(a) Minimal dis, Wpbc Dataset



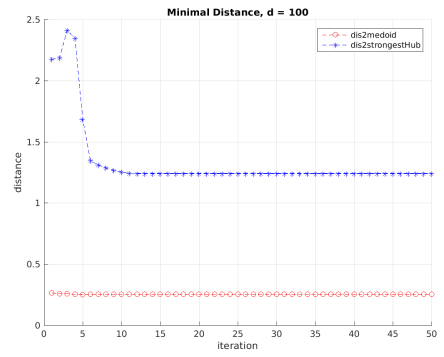
(b) Maximal dis, Wpbc Dataset



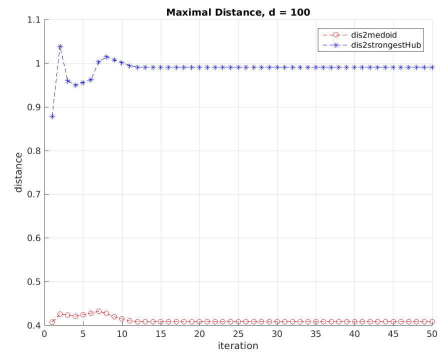
(c) Minimal dis, Sonar Dataset



(d) Maximal dis, Sonar Dataset

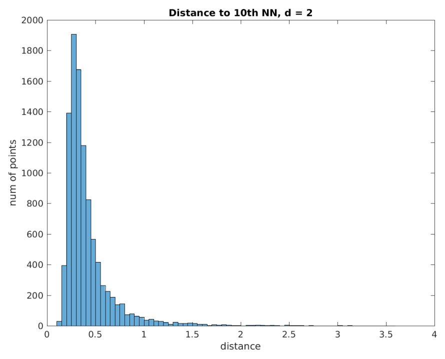


(e) Minimal dis, Hill Dataset

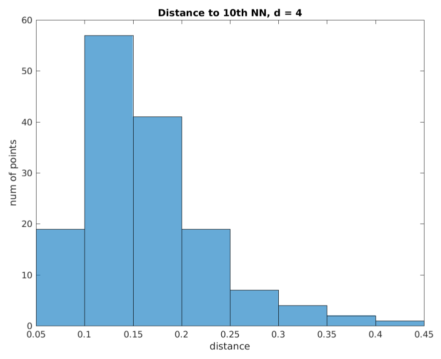


(f) Maximal dis, Hill Dataset

Figure 5: high-dim real data results



(a) Synthetic Dataset, d = 2



(b) Iris Dataset, d = 4

Figure 6: kdist graph of low dimensional data