MATH 252 FINAL PROJECT

BENCHMARKING STUDIES
ON
CLUSTER ANALYSIS

SUBMITTED BY

KRISHNA SINGH
SHERIN ANNA SUNNY

TABLE OF CONTENTS

1. INTRODUCTION

Cluster analysis is unsupervised learning where we don’t have information about the actual labels of the data. The end goal is to divide data into clusters of similar nature. There are multiple ways to define the similarity of clusters leading to different clustering techniques. These techniques can be broadly classified as hierarchical techniques, model-based partitioning techniques and not model based partitioning techniques. Often, the biggest challenge is to come up with the right number of clusters for a given dataset.

In this report, we are trying to do a neutral benchmarking study between different clustering techniques using both simulated data and an actual dataset. Four clustering algorithms have been selected for this analysis.

2. BACKGROUND

2.1 Clustering Algorithms

The objective of this analysis is to perform a comparative study on the following four clustering techniques:

1) K-Means
2) Gaussian mixture model
3) Partition around medoids
4) Bayesian agglomerative clustering

- K-Means

K-Means algorithm is a distance-based technique that partitions data into groups by minimizing the Euclidean distance of each point to its cluster center. This is achieved by minimizing the trace of within-group covariance matrix. The algorithm of K-Means is as follows:

1) Randomly select initial estimates for means of cluster
2) Compute the distance between all data points and the cluster centers
3) Allocate each observation to that group which has the minimum distance
4) Repeat steps 2 and 3 until the convergence is attained.

- Gaussian Mixture Model (GMM)

Mixture model clustering techniques use a probability density function to describe each of the sub-clusters/ components. In the Gaussian mixture model, a Gaussian distribution is considered for the sub-clusters. The ease of computation of the estimates of parameter makes this method popular. The parameters, in this case, are mixing proportion, mean vector and variance-covariance matrix for each component.

- Partition Around Medoids (PAM)

Partition around medoids is another distance-based clustering procedure that is similar to k-means. In PAM, clustering is achieved by minimizing the Manhattan distance between each point and its cluster medoid.

- Bayesian Agglomerative Clustering

Bayesian agglomerative clustering is a model based hierarchical clustering procedure that uses a linear model for data [1]. For data with V variables, that is to be grouped into C clusters, we define:

$Y_{ijk}=\mu+\delta_i\gamma_{ij}\theta_{ij}+\eta_{ijk}$, where, i corresponds to the continuous variable ranging from 1,….,V
j is the clusters ranging from 1,.....,C
k is the kth element in a cluster with values from 1,...,N_j

Y_{ijk} is a random variable which is data for a particular variable belonging to a particular cluster. μ is a constant overall mean, δ_i and γ_{ij} are binary random variables. θ_{ij} is a parameter which captures cluster effect for different variables. η_{ijk} is noise and is a continuous random variable independent of θ_{ij}.
Under the Bayesian method, θ_{ij} is a random variable which can have a Gaussian distribution or an asymmetric Laplace distribution. In our analysis, we have used a Gaussian distribution. The proposed model has 6 hyperparameters that are to be initialized before performing clustering. One of the biggest challenges in the Bayesian method is to come up with the correct prior distribution of the parameter. The conjugate prior to Gaussian likelihood is used in order to calculate posterior distribution [2]. In addition to performing clustering, this technique also performs variable selection thereby identifying the subset of variables that will be useful for clustering.

2.2 Functions and Package used:

For the cluster analysis, we used R software. The following functions and packages were used:

a) R Packages: mnormt, pgmm, EMMIXskew, cluster, mixture, Bclust
b) R functions: rmnorm, rdmsn, gpcm, Bclust, pam, k means**,** ARI, silhouette

2.3 Dataset:

For our analysis, we took simulated data and a real dataset into consideration. The entire analysis is based on continuous data and the details are as follows:

Simulation design:

The dataset is simulated from the normal distribution with 15 dimensions. For the purpose of our analysis, we have taken 4 scenarios in consideration. Each scenario was tested for 10 iterations. The considered scenarios are as follows:

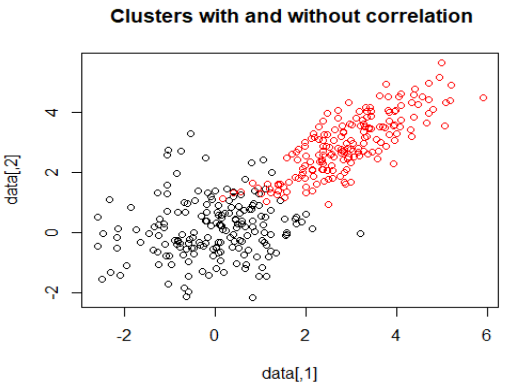

Fig.1 Simulated data with and without correlation

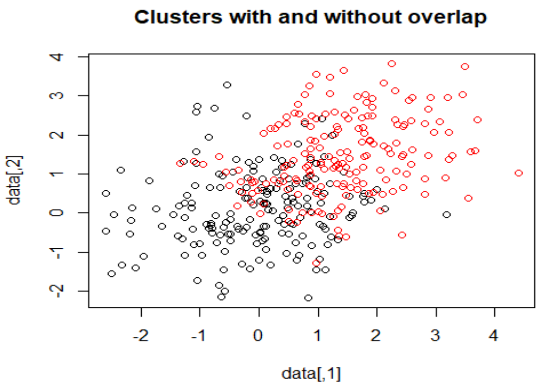

Fig.2 Simulated data with and without overlap

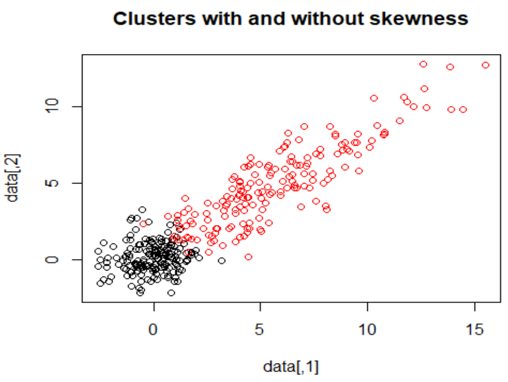

Fig 3. Simulated data with and without skewness

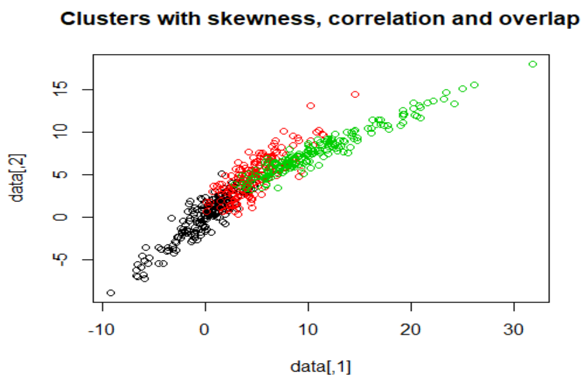

Fig 4. Simulated data with skewness, correlation, and overlap

Wine dataset:

This is a dataset with information about 27 physical and chemical properties of 178 Italian wine samples belonging to 3 different types. There are 13 chemical analyses reported about its physical and chemical properties [3].

The wine dataset is available in the package ‘pgmm’ in R.

2.4 Evaluation Metric:

To compare the performance of the four clustering techniques, we propose to use the two metrics namely Adjusted Rand Index (ARI) and Silhouette width (SW).

Adjusted Rand Index

Adjusted Rand Index is a measure of agreement between two quantities. It has a range between -1 and 1 and a value closer to 1 is preferred. We compare the cluster output from each of the methods to the true labels of the point. The ARI values for 10 iterations are averaged to obtain a mean ARI value for a specific technique. Hence, this metric measures how accurately the data is divided into various clusters. We use ARI as the primary metric for comparison.

Silhouette Width

Silhouette width provides a measure to assess cluster strength. It has a range between -1 and 1 and a value closer to 1 is preferred. Average silhouette width greater than 0.5 corresponds to a reasonable classification while values below 0.2 are considered as poor clustering. The average silhouette width values for 10 iterations are again averaged to obtain an aggregated measure for a technique.

3. RESULTS

The following table summarizes the mean ARI values for the four techniques and four scenarios

TABLE 1. ARI values

Dataset	GMM	PAM	BCLUST	K-MEANS
Correlation	0.99	0.88	0.92	0.87
Overlap	0.72	0.74	0.80	0.75
Skewness	0.99	0.55	0.64	0.33
Correlation, overlap & skewness	0.55	0.29	0.33	0.22
Wine dataset	0.89	0.42	0.44	0.41

The following table summarizes the mean silhouette width values for the four techniques and four scenarios

TABLE 2. Silhouette width values

Dataset	GMM	PAM	BCLUST	K-MEANS
Correlation	0.56	0.58	0.26	0.58
Overlap	0.27	0.32	0.32	0.33
Skewness	0.43	0.50	0.29	0.53
Correlation, overlap & skewness	0.40	0.49	0.19	0.51
Wine dataset	0.15	0.25	0.41	0.26

From the results of the simulation study, we find that the Gaussian mixture model performs well in almost all cases. For the three cases of simulations, including clusters with and without correlation, clusters with and without skewness, and clusters with a combination of correlation, overlap, and skewness: GMM has the highest ARI values followed by Bclust further followed by PAM and then K-means.The ARI values on the wine dataset also follow the same order. However, for the case with overlap, we see that Bclust has significantly higher performance than the remaining three algorithms. The poor results on the simulation design with correlation, overlap, and skewness are expected since the clusters are mixed with very less separation.

For all 4 simulation data, we find that k-means and PAM give the highest value for average silhouette width. However, for the wine dataset, Bclust algorithm has a significantly higher silhouette value compared to other algorithms. It is also interesting to note that for the wine data, on the one hand, GMM gives the lowest silhouette width but on the other hand it gives the highest ARI value.

4. CONCLUSION

Based on the results from our experiment, we can conclude that among the four clustering techniques namely Gaussian mixture models, partition around medoids, Bayesian agglomerative clustering and k-means clustering, the Gaussian mixture model provides better results under the various scenario. Bayesian agglomerative model tends to have an edge for data with overlapping clusters. Also, the results indicate that model-based techniques overperform the distance based techniques in accurately identifying cluster.

5. FUTURE RESEARCH

For the purpose of our project, we have restricted the data-simulation for fewer cases and 15 variables. The analysis can further be extended with different scenarios of simulation by increasing the number of variables, changing the underlying distribution of sampling, etc.
Our analysis broadly concentrates on 4 cluster algorithm such as GMM, PAM, Bclust and K-Means. This analysis can be further explored with different clustering algorithms.

6. REFERENCES

[1] V.P. Nia and A. C. Davison, “High Dimensional Bayesian Clustering with Variable Selection: The R package Bclust,” *J. of Statistical Software.*, vol.47, no. 5, Apr. 2012. [Online]. Available: https://www.jstatsoft.org/article/view/v047i05/v47i05.pdf

[2] K. A. Heller and Z. Ghahramani, “Bayesian Hierarchical Clustering,” in *Proceedings of 22nd Int. Conf. on Machine Learning*, Bonn, Germany, August, 2005, pp. 297-304. [Online]. Available:https://www2.stat.duke.edu/~kheller/bhcnew.pdf

[3] J. M. Santos and M. Embrechts, “On the Use of the Adjusted Rand Index as a Metric for Evaluating Supervised Classification,” in *Int. Conf. on Artificial Neural Networks.*

[4] https://rdrr.io/cran/rattle.data/man/wine.html

APPENDIX

```
# Load required libraries
library(mixture)
library(cluster)
library(bclust)
library(mnormt)
library(EMMIXskew)
library(mclust)
# Generate simulation data from gaussian distribution with different parameter values
### Scenario 1 : Clusters with and without correlation
set.seed(123)
resARI1=matrix(0,10,4)
ressil1=matrix(0,10,4)
lab=c(rep(1,180),rep(2,180))
sig=matrix(0.8,15,15)
diag(sig)=1
start<-Sys.time()
for(i in 1:10){
  c1=rmnorm(180,mean=rep(0,15),diag(15))
  c2=rmnorm(180,mean=rep(3,15),varcov=sig)
  data=rbind(c1,c2)
  #plot(data,col=lab)
  t1= gpcm(data,G=2,mnames="VVV")
  t2= pam(data,k=2)
  t3= bclust(data,transformed.par = c(0, -10, log(25), 0, 0, 0))
  t4= kmeans(data,centers=2,nstart=10)
  resARI1[i,1]=ARI(t1$map,lab)
  resARI1[i,2]=ARI(t2$clustering,lab)
  resARI1[i,3]=ARI(t3$optim.alloc,lab)
  resARI1[i,4]=ARI(t4$cluster,lab)
  sil1<-silhouette(x=t1$map,dist=dist(data))
  ressil1[i,1]=mean(sil1[,3])
  sil2<-silhouette(x=t2$clustering,dist=dist(data))
  ressil1[i,2]=mean(sil2[,3])
  sil3<-silhouette(x=t3$optim.alloc,dist=dist(data))
  ressil1[i,3]=mean(sil3[,3])
  sil4<-silhouette(x=t4$cluster,dist=dist(data))
  ressil1[i,4]=mean(sil4[,3])
}
# Compute mean ARI for 10 iterations
res1<-apply(resARI1,2,mean)
resw1<-apply(ressil1,2,mean)
```

```
end<-Sys.time()
print(end-start)
### Scenario 2: Clusters with and without overlap
set.seed(123)
resARI2=matrix(0,10,4)
ressil2=matrix(0,10,4)
lab=c(rep(1,180),rep(2,180))
sig=matrix(0.3,15,15)
diag(sig)=1
start<-Sys.time()
for(i in 1:10){
  c1=rmnorm(180,mean=rep(0,15),diag(15))
  c2=rmnorm(180,mean=rep(1.5,15),varcov=sig)
  data=rbind(c1,c2)
  #plot(data,col=lab)
  t1= gpcm(data,G=2,mnames="VVV")
  t2= pam(data,k=2)
  t3=bclust(data,transformed.par = c(0, -10, log(25), 0, 0, 0))
  t4= kmeans(data,centers=2,nstart=10)
  resARI2[i,1]=ARI(t1$map,lab)
  resARI2[i,2]=ARI(t2$clustering,lab)
  resARI2[i,3]=ARI(t3$optim.alloc,lab)
  resARI2[i,4]=ARI(t4$cluster,lab)
  sil1<-silhouette(x=t1$map,dist=dist(data))
  ressil2[i,1]=mean(sil1[,3])
  sil2<-silhouette(x=t2$clustering,dist=dist(data))
  ressil2[i,2]=mean(sil2[,3])
  sil3<-silhouette(x=t3$optim.alloc,dist=dist(data))
  ressil2[i,3]=mean(sil3[,3])
  sil4<-silhouette(x=t4$cluster,dist=dist(data))
  ressil2[i,4]=mean(sil4[,3])
}
# Compute mean ARI for 10 iterations
res2<-apply(resARI2,2,mean)
resw2<-apply(ressil2,2,mean)
end<-Sys.time()
print(end-start)
 ### Scenario 3: Clusters with and without skewness
set.seed(123)
resARI3=matrix(0,10,4)
ressil3=matrix(0,10,4)
lab=c(rep(1,180),rep(2,180))
```

```
start<-Sys.time()
for(i in 1:10){
  c1=rmnorm(180,mean=rep(0,15),diag(15))
  c2=rdmsn(180,15,mean=rep(1.5,15),cov=diag(15),del=c(5,4,1.5,2 ,-3, -3, 2, 5, 2, 2, -4, -3,
1,5,5))
  data=rbind(c1,c2)
  #plot(data,col=lab)
  t1= gpcm(data,G=2,mnames="VVV")
  t2= pam(data,k=2)
  t3=bclust(data,transformed.par = c(0, -10, log(25), 0, 0, 0))
  t4= kmeans(data,centers=2,nstart=10)
  resARI3[i,1]=ARI(t1$map,lab)
  resARI3[i,2]=ARI(t2$clustering,lab)
  resARI3[i,3]=ARI(t3$optim.alloc,lab)
  resARI3[i,4]=ARI(t4$cluster,lab)
  sil1<-silhouette(x=t1$map,dist=dist(data))
  ressil3[i,1]=mean(sil1[,3])
  sil2<-silhouette(x=t2$clustering,dist=dist(data))
  ressil3[i,2]=mean(sil2[,3])
  sil3<-silhouette(x=t3$optim.alloc,dist=dist(data))
  ressil3[i,3]=mean(sil3[,3])
  sil4<-silhouette(x=t4$cluster,dist=dist(data))
  ressil3[i,4]=mean(sil4[,3])
}
# Compute mean ARI for 10 iterations
res3<-apply(resARI3,2,mean)
resw3<-apply(ressil3,2,mean)
end<-Sys.time()
print(end-start)
#####  Scenario 4: Clusters with skewness, correlation and overlap
set.seed(123)
resARI4=matrix(0,10,4)
ressil4=matrix(0,10,4)
lab=c(rep(1,180),rep(2,180),rep(3,180))
start<-Sys.time()
for(i in 1:10){
  sig=matrix(0.3,15,15)
diag(sig)=1
  c1=rdmsn(180,15,mean=rep(3,15),cov=sig,del=c(-4,-4,1.5,0 ,0, 0, 2, 2, 0, 0, 2, 0, 0,1,1))
  c2=rdmsn(180,15,mean=rep(1.5,15),cov=diag(15),del=c(4,4,1.5,0 ,0, 0, 2, 2, 0, 0, 2, 0, 0,3,2))
  sig=matrix(0.7,15,15)
diag(sig)=1
```

```
  c3=rdmsn(180,15,mean=rep(4.5,15),cov=sig,del=c(8,4,1.5,2 ,-3, -3, 2, 5, 2, 2, -4, -7, 1,5,5))
  data=rbind(c1,c2,c3)
  plot(data,col=lab)
  t1= gpcm(data,G=2,mnames="VVV")
  t2= pam(data,k=2)
  t3= bclust(data,transformed.par = c(0, -10, log(25), 0, 0, 0))
  t4= kmeans(data,centers=2,nstart=10)
  resARI4[i,1]=ARI(t1$map,lab)
  resARI4[i,2]=ARI(t2$clustering,lab)
  resARI4[i,3]=ARI(t3$optim.alloc,lab)
  resARI4[i,4]=ARI(t4$cluster,lab)
  sil1<-silhouette(x=t1$map,dist=dist(data))
  ressil4[i,1]=mean(sil1[,3])
  sil2<-silhouette(x=t2$clustering,dist=dist(data))
  ressil4[i,2]=mean(sil2[,3])
  sil3<-silhouette(x=t3$optim.alloc,dist=dist(data))
  ressil4[i,3]=mean(sil3[,3])
  sil4<-silhouette(x=t4$cluster,dist=dist(data))
  ressil4[i,4]=mean(sil4[,3])
}
# Compute mean ARI for 10 iterations
res4<-apply(resARI4,2,mean)
resw4<-apply(ressil4,2,mean)
end<-Sys.time()
print(end-start)

 ### WINE DATASET
 data(wine,package="pgmm")
wine_data=as.matrix(wine[,2:28])
wine_lab=wine[,1]
 t1=gpcm(wine_data,G=3,mnames ="VVV")
 t2= pam(wine_data,k=3)
 t3= bclust(wine_data,transformed.par = c(0, -10, log(1.5), 0, 0, 0))
 t4= kmeans(wine_data,centers=3,nstart=10)
 resARI5_gpcm=ARI(t1$classification,wine_lab)
 resARI5_pam=ARI(t2$clustering,wine_lab)
 resARI5_bclust=ARI(t3$optim.alloc,wine_lab)
 resARI5_kmeans=ARI(t4$cluster,wine_lab)
 sil1<-silhouette(x=t1$classification,dist=dist(wine_data))
 ressil4_gpcm=mean(sil1[,3])
 sil2<-silhouette(x=t2$clustering,dist=dist(wine_data))
 ressil4_pam=mean(sil2[,3])
```

```
sil3<-silhouette(x=t3$optim.alloc,dist=dist(wine_data))
ressil4_bclust=mean(sil3[,3])
sil4<-silhouette(x=t4$cluster,dist=dist(wine_data))
ressil4_kmeans=mean(sil4[,3])
```