Cluster Analysis on Khan datasets using K-means VS Hierarchical clustering

Rohan Patil and Momina Tariq[1]

May 9, 2016

[1] Dept. of Bioinformatics and Computational Biology, George Mason University

Introduction

The research topic of our project is based on the following research question and hypothesis.

Research Question:

Which clustering method is the most perferable for analyzing gene data? why?

Hypothesis:

K-means clustering is most commonly used and hence we hypothesize that it would be the most preferred one. However, we would reproduce our own clusters using both Hierarchical and K-means clustering methods, then validate cluster solutions, compare their performances and analyze which one is more preferable.

Contents

This report of our research project is devided in following sections:

- Background and objectives,
- Computational methods,
  - Data Preparation,
  - Hierarchical Clustering,
  - K-means Clustering,
  - Model-Based Clustering,
  - Plotting cluster solutions,
  - Validating cluster solutions,
- Results and Discussion,
- Conclusions and a brief description of how these conclusions could be tested using biochemical or genetic techniques.
- References

Background

- Khan data set uses cDNA microarrays containing 6567 clones of which 3789 were known genes and 2778 were ESTs to study the expression of genes in of four types of small round blue cell tumours of childhood (SRBCT).
- Gene expression profiles from both tumor biopsy and cell line samples were obtained and are contained in this dataset
- Sources -
  - *This data were originally reported in:* Khan J, Wei J, Ringner M, Saal L, Ladanyi M, Westermann F, Berthold F, Schwab M, Antonescu C, Peterson C, and Meltzer P. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine, v.7, pp.673-679, 2001.
  - *The data were also used in:* Tibshirani RJ, Hastie T, Narasimhan B, and G. Chu. Diagnosis of Multiple Cancer Types by Shrunken Centroids of Gene Expression. Proceedings of the National Academy of Sciences of the United States of America, v.99(10), pp.6567-6572, May 14, 2002.

Khan Datasets

This interesting data set offers two interesting items:

- Train: data.frame of 2308 rows and 63 columns. The training data set of 63 arrays and 2308 gene expression values
- Test: data.frame of 2308 rows and 20 columns. The test data set of 20 arrays and 2308 genes expression values

For each tissue sample, gene expression measurements are available. The data set consists of training data, xtrain and ytrain, and testing data, xtest and ytest

Cluster Analysis

- Much of the history of cluster analysis is concerned with developing algorithms that were not too computer intensive.
- A problem which often arises in Bioinformatics is to find genes which have similar expression patterns
- In general, cluster analysis also known as unsupervised learning consists of several methods for discovering a subset of data points, such as genes which form a group under some observable similarity criteria, such as gene expression.

Objectives

Our main objectives are to perform different clustering methods and compare thier performances against Khan datasets.
These methods can be divided into:

- K-means clustering,
- Hierarchical clustering, and
- Model-Based Clustering

K-means Cluster Analysis

- One of the oldest method of cluster analysis
- First step- specify the number of clusters (k)
- The process begins by choosing k observations to serve as centers for the clusters
- Then, the distance from each of the other observations is calculated for each of the k clusters, and observations are put in the cluster to which they are the closest.
- Recalculation of centers
- The process continues until no observations switch clusters

Hierarchical Agglomerative Clustering

- It starts out by putting each observation into its own separate cluster
- It then examines all the distances between all the observations and pairs together the two closest ones to form a new cluster
- One less cluster than there are observations
- Agglomerative Hierarchical cluster analysis is provided in R through the hclust function

Data Preperation

- This step is done to remove or estimate missing data and rescale variables for comparability
- A generic function called scale() was applied on our data set
- Principal component analysis was also done to summarize our set with a smaller number of representative variables that collectively explain most of the variability in our original set.

Data Preperation

Prior to clustering data, you may want to remove or estimate missing data and rescale variables for comparability.

```r
## For Khan train data
naTrain = na.omit(Khan$xtrain)  # Omit Missing Values
stdTrain = scale(naTrain)  # Data Standardization
distTrain = dist(stdTrain)  # Euclidian Distance
## For Khan test data
naTest = na.omit(Khan$xtest)  # Omit Missing Values
stdTest = scale(naTest)  # Data Standardization
distTest = dist(stdTest)  # Euclidian Distance

## Perform principal components analysis using prcomp with scale=TRUE.

pr.out.train = prcomp(Khan$xtrain, scale. = TRUE)  # for training set
pr.out.test = prcomp(Khan$xtest, scale. = TRUE)  # for test set
```

Hierarchical Clustering

- Tree-like visual representation of the observations, called a dendrogram.
- plot function
- The term hierarchical refers to the fact that clusters obtained by cutting the dendrogram at a given height

Hierarchical Clustering

```
## Perform hierarchical clustering using average linkage clustering.
hc.avg.train = hclust(distTrain, method ="average")
hc.avg.test = hclust(distTest, method ="average")
## Re-perform hierarchical clustering and cut the tree into 4 clusters.
hc.out.train = hclust(distTrain)
hc.cutree.train = cutree(hc.out.train, 4)
table(hc.cutree.train)  # For Khan train data

## hc.cutree.train
##  1  2  3  4
## 34  8 19  2

hc.out.test = hclust(distTest)
hc.cutree.test = cutree(hc.out.test, 4)
table(hc.cutree.test)  # For Khan test data

## hc.cutree.test
##  1  2  3  4
## 10  2  5  3
```

K-Means Clustering

- Unsupervised learning algorithm
- Partitioning of our data set into K distinct, non-overlapping clusters
- First the desired number of K clusters are specified

K-Means Clustering

```r
# Perform k-means clustering on all data with k=4.
set.seed(0)
## Khan train data
km.out.train = kmeans(stdTrain, 4)
km.clusters.train = km.out.train$cluster
## Khan test data
km.out.test = kmeans(stdTest, 4)
km.clusters.test = km.out.test$cluster

# Re-perform k-means clustering on only 1st 4 Principal Components.
set.seed(0)
km1st4.out.train = kmeans(stdTrain[,1:4], 4)
km1st4.out.test = kmeans(stdTrain[,1:4], 4)
```

Model-Based Clustering

- mclust is a contributed R package for model-based clustering, classification, and density estimation based on finite normal mixture modelling
- It provides functions for parameter estimation via the EM algorithm
- The Mclust( ) function in the mclust package selects the optimal model according to BIC for EM initialized by hierarchical clustering for parameterized Gaussian mixture models

Model-Based Clustering - Training set

```
# For Khan train data
mc.train = Mclust(Khan$xtrain,Khan$ytrain)
summary(mc.train)

## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EEI (diagonal, equal volume and shape) model with 4 components
##
##  log.likelihood  n    df      BIC     ICL
##       -108678.4 63 11543 -265181 -265181
##
## Clustering table:
##  1  2  3  4
## 17 22 16  8

MDA = MclustDA(Khan$xtrain[,1:4], Khan$ytrain)
```

Model-Based Clustering - Test set

```
# For Khan test data
mc.test = Mclust(Khan$xtest,Khan$ytest)
summary(mc.test)

## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 3 components:
##
##  log.likelihood  n   df       BIC       ICL
##        -32808.4 20 9236 -93285.38 -93285.38
##
## Clustering table:
##  1  2  3
##  5 10  5

# MclustDA(Khan$xtest[,1:4], Khan$ytest)
```

Principal Components Analysis, scaled TRUE

```
## Provide a pairs plot on the first 2 principal components.
Cols=function(vec){ cols = rainbow(length(unique(vec)))
  return(cols[as.numeric(as.factor(vec))])}
par(mfrow=c(1,2))
plot(pr.out.train$x[,1:2], col=Cols(Khan$ytrain),
     main="Fig.1.1: Plotting 1st 2 Principal Components of training set"
plot(pr.out.test$x[,1:2], col=Cols(Khan$ytest),
     main="Fig.1.2: Plotting 1st 2 Principal Components of test set")
```

Fig.1.1: Plotting 1st 2 Principal Components of trainin

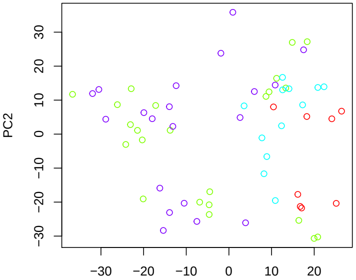
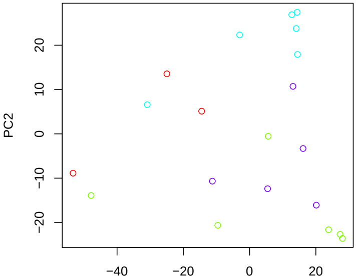


Fig.1.2: Plotting 1st 2 Principal Components of test s



PVE

```
pve=100*pr.out.train$sdev^2/sum(pr.out.train$sdev^2)
par(mfrow=c(1,2))
plot(pve, type="o", ylab="PVE", xlab="Principal Component",
     col =" blue ", main="Fig.1.3: the PVE of each principal component")
plot(cumsum(pve), type="o", ylab="Cumulative PVE", xlab="Principal Compo
     col =" brown3 ", main="Fig.1.4: cumulative PVE of the principal com
```

Fig.1.3: the PVE of each principal component

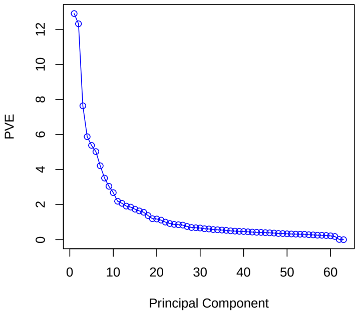
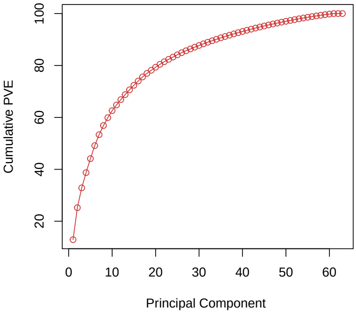


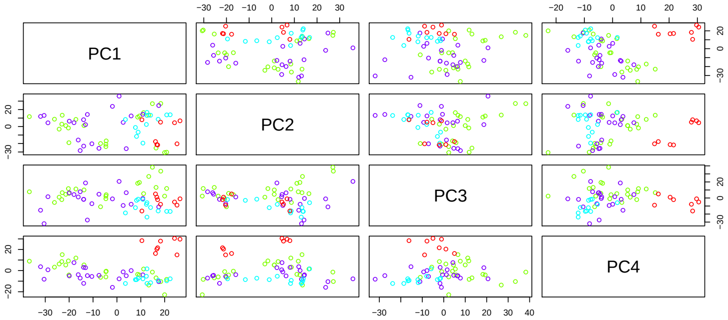
Fig.1.4: cumulative PVE of the principal component



Pairs plot for 1st 4 Principal Components of training set

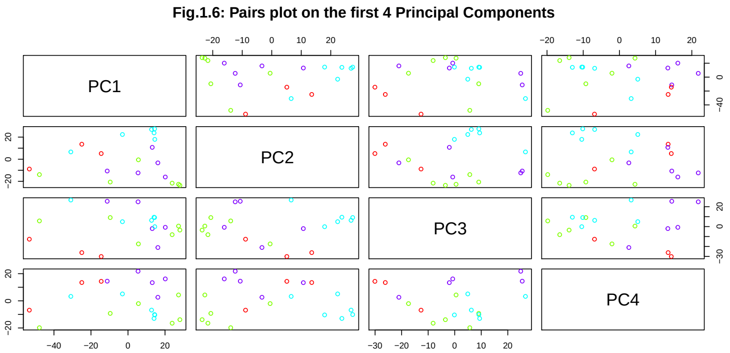
```r
pairs(pr.out.train$x[,1:4], col=Cols(Khan$ytrain),
      main="Fig.1.5: Pairs plot on the first 4 Principal Components")
```

Fig.1.5: Pairs plot on the first 4 Principal Components



Pairs plot for 1st 4 Principal Components of test set

```
pairs(pr.out.test$x[,1:4], col=Cols(Khan$ytest),
      main="Fig.1.6: Pairs plot on the first 4 Principal Components")
```



Plot hierarchical clusters with average linkage

```
par(mfrow=c(1,2))
plot(hc.avg.train, labels=Khan$ytrain, main="Fig.2.1: Average Linkage
     clustering of training set", xlab="", sub="", ylab ="")
plot(hc.avg.test, labels=Khan$ytest, main="Fig.2.2: Average Linkage
     clustering of test set", xlab="", sub="", ylab="")
```

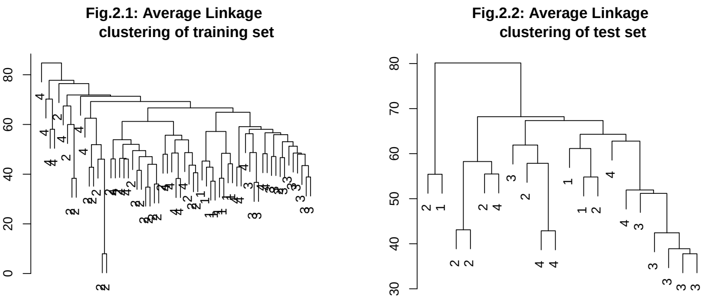


Fig.2.1: Average Linkage clustering of training set

Fig.2.2: Average Linkage clustering of test set

Plot hierarchical clusters with a line cutting the tree of 4 groups

```r
par(mfrow=c(1,2))
plot(hc.out.train, labels = Khan$ytrain, main="Fig.2.3: Cluster Dendogra
     with a line cutting the tree of training set"); # abline(h=89.5, co
rect.hclust(hc.out.train, k=4, border="red")
plot(hc.out.test, labels = Khan$ytest,main="Fig.2.4: Cluster Dendogram
     with a line cutting the tree of test set"); # abline(h=73.5, col ='
rect.hclust(hc.out.test, k=4, border="red")
```

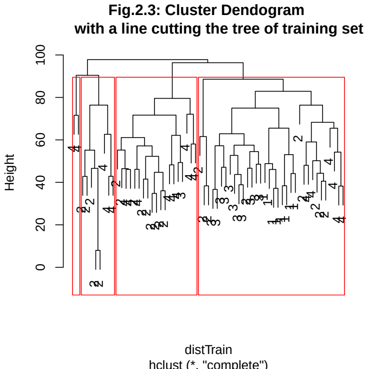


Fig.2.3: Cluster Dendogram with a line cutting the tree of training set

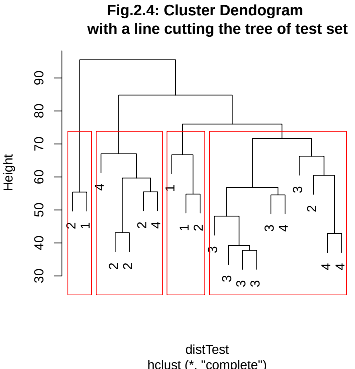
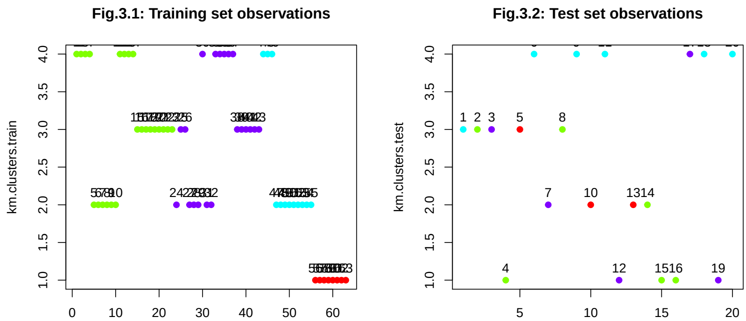


Fig.2.4: Cluster Dendogram with a line cutting the tree of test set

Plot K-means clusters on all principal components

```
par(mfrow=c(1,2))
plot(km.clusters.train, col=Cols(Khan$ytrain), pch=19,
     main="Fig.3.1: Training set observations")
text(km.clusters.train, row.names(km.clusters.train), pos=3)

plot(km.clusters.test, col=Cols(Khan$ytest), pch=19,
     main="Fig.3.2: Test set observations")
text(km.clusters.test, row.names(km.clusters.test), pos=3)
```



Plot K-means clusters on 1st 4 principal components

```
par(mfrow=c(1,2))
plot(km1st4.out.train$cluster, col=Cols(Khan$ytrain), pch=19,
     main="Fig.3.3: Training set observations")
text(km1st4.out.train$cluster, row.names(km1st4.out.train$cluster), pos=

plot(km1st4.out.test$cluster, col=Cols(Khan$ytest), pch=19,
     main="Fig.3.4: Test set observations")
text(km1st4.out.test$cluster, row.names(km1st4.out.test$cluster), pos=3)
```

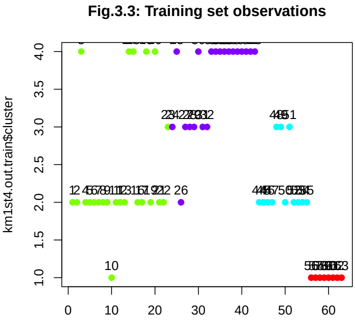


Fig.3.3: Training set observations

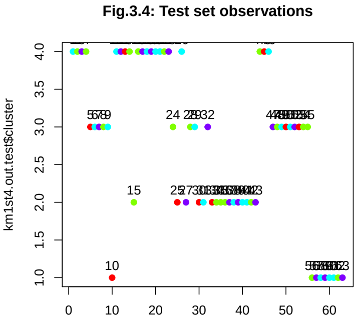
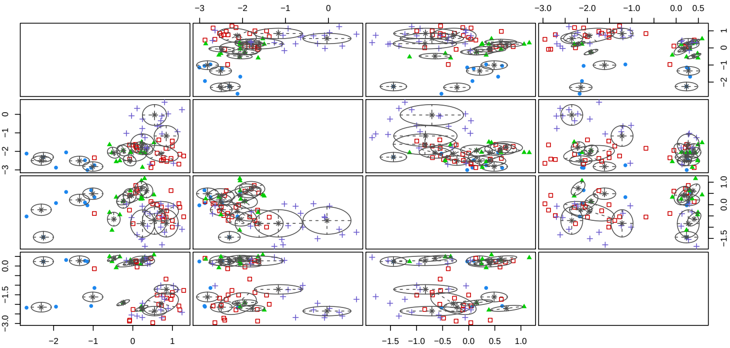


Fig.3.4: Test set observations

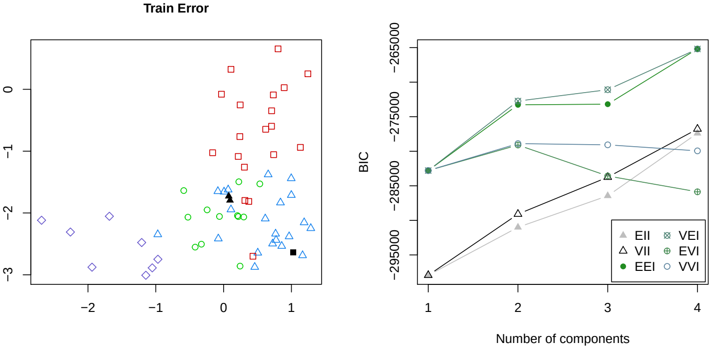
Plot Model-based clusters for training

```
plot(MDA, what = "scatterplot")
```



Plot Model-based clusters for error and BIC

```
par(mfrow=c(1,2));plot(MDA, what = "error");plot(mc.train, what = "BIC")
```



Validate Hierarchical Cluster Solution

Table of hierarchical clustering results based on cut of 4 groups

```
table(hc.cutree.train, Khan$ytrain)  # For Khan train data

##
## hc.cutree.train  1  2  3  4
##               1  8  9 11  6
##               2  0  5  0  3
##               3  0  9  1  9
##               4  0  0  0  2

table(hc.cutree.train == Khan$ytrain)

##
## FALSE  TRUE
##    47    16

mean(hc.cutree.train == Khan$ytrain)

## [1] 0.2539683
```

Validate Hierarchical Cluster Solution

```
table(hc.cutree.test, Khan$ytest)  # For Khan test data

##
## hc.cutree.test 1 2 3 4
##              1 0 1 6 3
##              2 1 1 0 0
##              3 0 3 0 2
##              4 2 1 0 0

table(hc.cutree.test == Khan$ytest)

##
## FALSE  TRUE
##    19     1

mean(hc.cutree.test == Khan$ytest)

## [1] 0.05
```

So using hierarchical clustering, 25.39% of the training observations are correctly classified and 5% of the test observations are correctly classified.

Validate K-means Cluster Solution

Table of k-mean clustering results with k = 4 clusters

```
table(km.clusters.train, Khan$ytrain)  # For Khan train data

##
## km.clusters.train 1 2 3 4
##                 1 8 0 0 0
##                 2 0 6 9 6
##                 3 0 9 0 8
##                 4 0 8 3 6

table(km.clusters.train == Khan$ytrain)

##
## FALSE  TRUE
##    43    20

mean(km.clusters.train == Khan$ytrain)

## [1] 0.3174603
```

Validate K-means Cluster Solution

```
table(km.clusters.test, Khan$ytest)  # For Khan test data

##
## km.clusters.test 1 2 3 4
##                1 0 3 0 2
##                2 2 1 0 1
##                3 1 2 1 1
##                4 0 0 5 1

table(km.clusters.test == Khan$ytest)

##
## FALSE  TRUE
##    17     3

mean(km.clusters.test == Khan$ytest)

## [1] 0.15
```

So using K-means clustering, 31.75% of the training observations are correctly classified and 15% of the test observations are correctly classified.

Results and Discussion

Comparing performances of K-means and Hierarchical clustering

```
table(km.clusters.train, hc.cutree.train)  # For training set

##                    hc.cutree.train
## km.clusters.train  1  2  3  4
##                  1  8  0  0  0
##                  2  9  8  4  0
##                  3  0  0 15  2
##                  4 17  0  0  0

table(km.clusters.train == hc.cutree.train)

##
## FALSE  TRUE
##    32    31

mean(km.clusters.train == hc.cutree.train)

## [1] 0.4920635
```

Results and Discussion

```
table(km.clusters.test, hc.cutree.test)  # For test set

##                  hc.cutree.test
## km.clusters.test 1 2 3 4
##                1 1 0 4 0
##                2 0 0 1 3
##                3 3 2 0 0
##                4 6 0 0 0

table(km.clusters.test == hc.cutree.test)

##
## FALSE  TRUE
##    19     1

mean(km.clusters.test == hc.cutree.test)

## [1] 0.05
```

We see that the four clusters obtained using these 2 mthods are somewhat different. K-means clustering contains only 49.2% portion of training observations and just 5% of test observations assigned to Hierarchical clustering.

Conclusions

- It seems from the comparison of their performances that K-means clustering could do better classification than Hierarchical clustering. Athough the classification accuracy wasn't so great in either cases due to the moderate size of dataset, the clusters were successfully obtained in both cases. And this could be improved with bigger sizes of datasets or more classes.
- One down side of k-means is that, if you rearrange your data, it's very possible that you'll get a different solution every time you change the ordering of your data. This makes the procedure somewhat unattractive if you don't know exactly how many clusters you should have in the first place.

Brief Description

How could the conclusions of our analyses be tested using biochemical or genetic techniques?

- Monitoring global gene-expression levels by cDNA microarrays provides an additional tool for elucidating tumor biology as well as the potential for molecular diagnostic classification of cancer. Currently, classification and clustering tools using gene-expression data have not been rigorously tested for diagnostic classification of more than two categories.
- Other approaches that share the parametric nature of artificial neural networks and have been utilized to classify gene-expression profiles include Support Vector Machines. Thus far, these other methods have not been fully explored to extract the genes or features that are most important for the classification performance and which also will be of interest to cancer biologists.
- Our method identifies genes related to tumor histogenesis, but includes genes that may not normally be expressed in the corresponding mature tissue.

References

- Games, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An Introduction to Statistical Learning with applications in R, www.StatLearning.com, Springer-Verlag, New York.
- Anderberg, M. R. (2014). Cluster analysis for applications: probability and mathematical statistics: a series of monographs and textbooks (Vol. 19). Academic press.
- Hair, J. F., Black, W. C., Babin, B. J., Anderson, R. E., and Tatham, R. L. (2006). Multivariate data analysis (Vol. 6). Upper Saddle River, NJ: Pearson Prentice Hall.
- Suzuki, R., and Shimodaira, H. (2006). Pvclust: an R package for assessing the uncertainty in hierarchical clustering. Bioinformatics, 22(12), 1540-1542.
- Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 24(7), 881-892.