Hierarchical Clustering

## How do we define close?

Distance or similarity

* Continuous – Euclidean distance
* Continuous – correlation similarity
* Binary – Manhattan distance

## Distance Calculation

dist()

The input can be a numeric matrix/data frame. It views each row as a point. It calculates the distances between every pair of rows (points).

## Hierarchical Clustering- hclust

> distxy <- dist(dataFrame)

> hClustering <- hclust(distxy)

> plot(hClustering)

Then you can cut the tree at some level to decide how many clusters you want to produce.

## Prettier Dendrograms

<http://rpubs.com/gaston/dendrograms>

## A script here produces pretty dendrograms:

myplclust <- function( hclust, lab=hclust$labels, lab.col=rep(1,length(hclust$labels)), hang=0.1,...){

## modifiction of plclust for plotting hclust objects \*in colour\*!

## Copyright Eva KF Chan 2009

## Arguments:

## hclust: hclust object

## lab: a character vector of labels of the leaves of the tree

## lab.col: colour for the labels; NA=default device foreground colour

## hang: as in hclust & plclust

## Side effect:

## A display of hierarchical cluster with coloured leaf labels.

y <- rep(hclust$height,2)

x <- as.numeric(hclust$merge)

y <- y[which(x<0)]

x <- x[which(x<0)]

x <- abs(x)

y <- y[order(x)]

x <- x[order(x)]

plot( hclust, labels=FALSE, hang=hang, ... )

text( x=x, y=y[hclust$order]-(max(hclust$height)\*hang), labels=lab[hclust$order], col=lab.col[hclust$order], srt=90, adj=c(1,0.5), xpd=NA, ... )

}

## How to use this script:

> dataFrame <- data.frame(x=x,y=y)

> distxy <- dist(dataFrame)

> hClustering <- hclust(distxy)

> myplclust(hClustering, lab=rep(1:3, each=4), lab.col = rep(1:3,each=4))

## How to Merge Points Together?

One issue in hierarchical clustering is how to merge two points together.

1. Complete-linkage clustering: Calculate the distance between two farthest points in the two clusters.
2. Average-linkage clustering

## Heat Map – heatmap()

A heat map is a false color image (basically [image](http://127.0.0.1:46886/help/library/stats/help/image)(t(x))) with a dendrogram added to the left side and to the top. Typically, reordering of the rows and columns according to some set of values (row or column means) within the restrictions imposed by the dendrogram is carried out.

K-means Clustering

## Main Idea of K-means Clustering

* A partitioning approach
  + Fix a number of clusters
  + Get “centroids” of each cluster
  + Assign things to closest centroid
  + Recalculate centroids
* Requires
  + A defined distance metric
  + K
  + An initial guess as to cluster centroids
* Produces
  + Final estimate of cluster centroids
  + An assignment of each point to clusters

## kmeans() in R

Important parameters: x, centers, iter.max, nstart

> dataFrame <- data.frame(x,y)

> kmeansObj <- kmeans(dataFrame,centers=3)

How to Plot

> plot(x,y, col=kmeansObj$cluster, pch=19,cex=2)

> points(kmeansObj$centers, col=1:3, pch=3, cex=3, lwd=3)

> names(kmeansObj)

[1] "cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss" "size" "iter"

[9] "ifault"

Notes of K-means

* K-means requires a number of clusters
  + Pick by eye/intuition
  + Pick by cross validation/information theory..
* K-means is not deterministic
  + Different number of clusters
  + Different number of iterations

Principal Components Analysis and Singular Value Decomposition

Matrix Data

> set.seed(12345)

> par(mar=rep(0.2,4))

> dataMatrix <- matrix(rnorm(400), nrow=40)

> image(1:10, 1:40, t(dataMatrix)[, nrow(dataMatrix):1])

This will shows a heat-map like image.

image(): Creates a grid of colored or gray-scale rectangles with colors corresponding to the values in z. This can be used to display three-dimensional or spatial data aka *images*. This is a generic function.

t(): Transpose a matrix.