Sk-means Clustering Implementation

Saptarshi Chakraborty

5 October 2018

This file contains the codes and an example to implement --means algorithm.

The corresponding paper can be found here

S. Chakraborty, S. Das, -Means clustering with a new divergence-based distance metric: Convergence and performance analysis, Pattern Recognition Letters (2017), <https://doi.org/10.1016/j.patrec.2017.09.025>

First Load the libraries.

library(igraph)

##   
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

## The following object is masked from 'package:base':  
##   
## union

library(rgl)  
library(MASS)  
library(mlbench)

Define the s-distance

delta=function(x,y){ ## x,y > 0 are vectors  
   
 sqrt( sum( log((x+y)/2)-0.5\*(log(x)+log(y))))  
   
}

Then run the --means algorithm.

# DEFINE WHAT THE CENTER IS  
E=function(mu,Y){  
 n=dim(Y)[1]  
 s=0  
 for(i in 1:n){  
 s=s+(delta(Y[i,],mu))^2  
 }  
 return(s)  
}  
  
# KMEANS  
s.kmeans=function(X,M,itermax){  
 n=dim(X)[1]  
 d=dim(X)[2]  
 k=dim(M)[1]  
 label=numeric(n)  
 dist=numeric(k)  
   
 for(l in 1: itermax){  
 for(i in 1:n){  
 for(j in 1:k){  
 dist[j]=delta(X[i,],M[j,])  
 }  
 label[i]=which.min(dist)  
 }  
   
   
 for(i in 1:k){  
 I=which(label==i)  
 M[i,]=optim(colMeans(X[I,]),E,Y=X[I,])$par  
 }  
 }  
 return(list(label,M))  
}

# Example

We will now run the --means algorithm on the iris dataset.

data(iris) #Load the data  
X=iris   
X=data.matrix(X)  
toss=X[,5] # Since the last column contains the class information  
X=X[,-5]  
n=dim(X)[1]  
numclus=3 #Since iris has 3 classes   
sort(unique(toss))

## [1] 1 2 3

X=X-min(X)+1 # Make everything in X positive (Otherwise the distance won't be defined)

Now we run the --means algorithm on this dataset. We will only run upto 30 iterations here.

sa=sample(n,numclus)  
M=X[sa,] # Select Random Centroids  
l=s.kmeans(X,M,30)

Let’s look into the output l

l

## [[1]]  
## [1] 3 1 1 1 3 3 1 3 1 1 3 3 1 1 3 3 3 3 3 3 3 3 1 3 3 1 3 3 3 1 1 3 3 3 1  
## [36] 1 3 1 1 3 3 1 1 3 3 1 3 1 3 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
## [71] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
## [106] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
## [141] 2 2 2 2 2 2 2 2 2 2  
##   
## [[2]]  
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## [1,] 5.600626 4.010375 2.275874 1.093729  
## [2,] 7.131844 3.757482 5.747836 2.541582  
## [3,] 6.118996 4.543734 2.415284 1.177408

l has two parts. The first part (l[[1]]) contains the cluster assignments. The scond part (l[[2]]) contains the matrix of centroids i.e. the -th row of this matrix corresponds to the cluster centroid for the cluster .

Now Measure the accuracy of the clustering

compare(toss,l[[1]],method='nmi') #NMI Score

## [1] 0.6488784

compare(toss,l[[1]],method='adjusted.rand') #ARI Score

## [1] 0.4428456