Report K mean

Margarita Orlova, Kenneth Ruiter, Patrick Walker

November 3, 2018

K-means classifier

In the code below we cluster data from the wine dataset, using a k-means method. Our function returns a list with 2 objects: the means of the clusters and the results of our clustering. We receive clusters with a total of 47, 62 and 69 observations. Note that the k-means function built in R produces the same result, meaning that our code is working.

```r
library(rattle) #import data
library(fpc) #import plot
data(wine)

mywine <- as.matrix(wine[,-1]) # remove the existing clustering

cluster <- function(data){ #declare function
  nrows <- length(data[,1]) #create variable with number of rows of data
  ncols <- length(data[1,]) #create variable with number of columns of data
  iterations <- 20
  set.seed(1)
  means <- data[sample(1:nrows,3),] #select rows from the original data set to serve
    #as the initial means/centers
  for(i in 1:iterations){ #for loop for each of the iterations of reassigning the means
    distances <- matrix(rep(0,3*nrows),nrow = nrows) #distances matrix initialized
    mindistances <- matrix(rep(0, 2*nrows), nrow = nrows) #mindistances matrix initialized
    for(j in 1:nrows){ #for each row
      for(k in 1:3){ #for each cluster
        distances[j,k] <- sum((data[j,]-means[k,])^2)
        #finds the difference between the row j and the mean ascribed to cluster K
      }
      mindistances[j,] <- c(min(distances[j,]), which.min(distances[j,]))
      #finds the minimum distance for each row amongst the three clusters and assigns
      #it to mindistances in the first column and the second column has the cluster
    }
    for(l in 1:3){ #for each cluster
      datarows <- data[mindistances[,2]==l,] #selects the data corresponding to
        #each of the clusters
      means[l,] <- colMeans(data[mindistances[,2]==l,])
      #the above line takes the mean of each column and
      #stores them as the new means vector
    }
  }
  return(list(means, mindistances)) #returns means and mindistances
}
results <- cluster(mywine)
results[[1]][,]
```

```
##       Alcohol    Malic      Ash Alcalinity Magnesium  Phenols Flavanoids
## [1,] 13.80447 1.883404 2.426170   17.02340 105.51064 2.867234   3.014255
## [2,] 12.51667 2.494203 2.288551   20.82319  92.34783 2.070725   1.758406
## [3,] 12.92984 2.504032 2.408065   19.89032 103.59677 2.111129   1.584032
##      Nonflavanoids Proanthocyanins    Color       Hue Dilution   Proline
## [1,]     0.2853191        1.910426 5.702553 1.0782979 3.114043 1195.1489
## [2,]     0.3901449        1.451884 4.086957 0.9411594 2.490725  458.2319
## [3,]     0.3883871        1.503387 5.650323 0.8839677 2.365484  728.3387
```

```r
sum(results[[2]][,2] == 1)
```

```
## [1] 47
```

```r
sum(results[[2]][,2] == 2)
```

```
## [1] 69
```

```r
sum(results[[2]][,2] == 3)
```

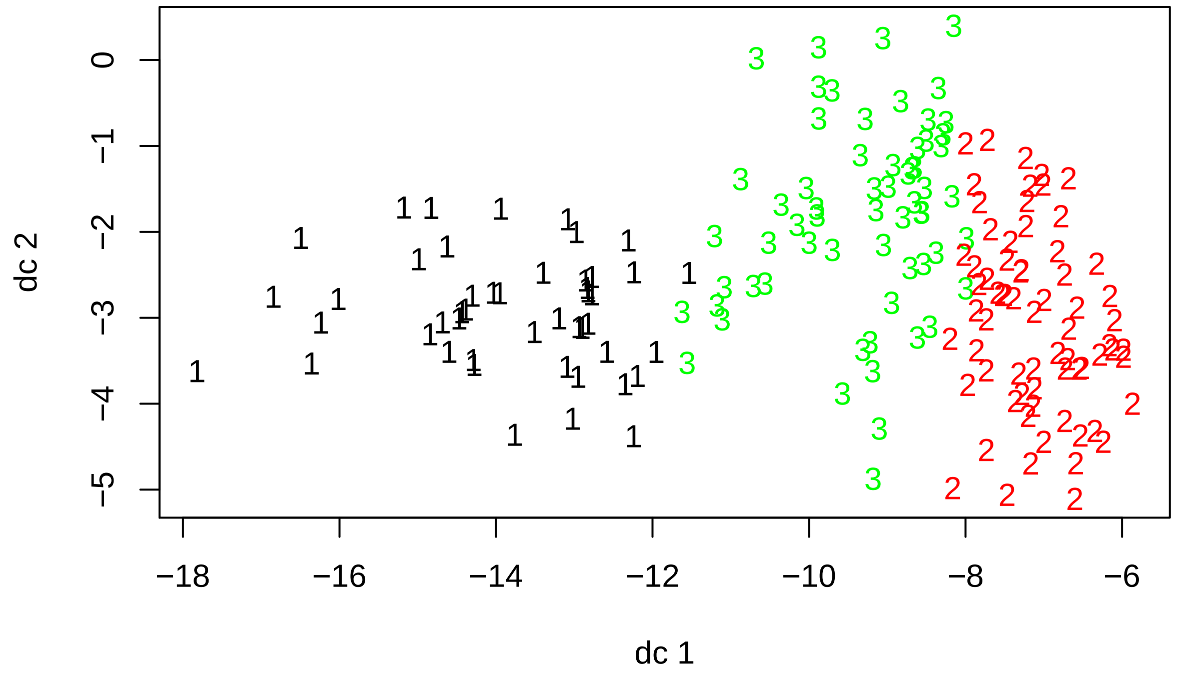
```
## [1] 62
```

```r
kmeans(wine[,-1],3,algorith="Lloyd")  #running R k-means classifier as comparison
```

```
## K-means clustering with 3 clusters of sizes 62, 47, 69
##
## Cluster means:
##    Alcohol    Malic      Ash Alcalinity Magnesium  Phenols Flavanoids
## 1 12.92984 2.504032 2.408065   19.89032 103.59677 2.111129   1.584032
## 2 13.80447 1.883404 2.426170   17.02340 105.51064 2.867234   3.014255
## 3 12.51667 2.494203 2.288551   20.82319  92.34783 2.070725   1.758406
##   Nonflavanoids Proanthocyanins    Color       Hue Dilution   Proline
## 1     0.3883871        1.503387 5.650323 0.8839677 2.365484  728.3387
## 2     0.2853191        1.910426 5.702553 1.0782979 3.114043 1195.1489
## 3     0.3901449        1.451884 4.086957 0.9411594 2.490725  458.2319
##
## Clustering vector:
##   [1] 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 2 1 1 2 2 1 2 2 2 2 2 2
##  [36] 1 1 2 2 1 1 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 1 3 1 3 3 1 3 3 1 1
##  [71] 1 3 3 2 1 3 3 3 1 3 3 1 1 3 3 3 3 3 1 1 3 3 3 3 3 1 1 3 1 3 1 3 3 3 1
## [106] 3 3 3 3 1 3 3 1 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 1 3 3 1 1 1 1 3 3 3
## [141] 1 1 3 3 1 1 3 1 1 3 3 3 3 1 1 1 3 1 1 1 3 1 3 1 1 3 1 1 1 1 3 3 1 1 1
## [176] 1 1 3
##
## Within cluster sum of squares by cluster:
## [1]  566572.5 1360950.5  443166.7
##  (between_SS / total_SS =  86.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Then we visualize our clusters to check how well are they separated. And as we see from the plot, observations are grouped adequate, however there is no distinct border between clusters.

```
plotcluster(mywine, results[[2]][,2]) #plots clusters
```



Later, we have to quantify the results of k-mean clustering. In order to do that we run following code:

```
require(data.table)
wine_cluster <- cbind(wine,results[[2]][,2]) #add variable with  cluster to wine dataset
colnames(wine_cluster)[15] <- "clusters" #name new variable
type_1 <-setkey(data.table((wine_cluster[wine_cluster$Type == 1,2:14])))
  #subgroup observation with type 1
type_2 <-setkey(data.table((wine_cluster[wine_cluster$Type == 2,2:14])))
  #subgroup observation with type 2
type_3 <-setkey(data.table((wine_cluster[wine_cluster$Type == 3,2:14])))
  #subgroup observation with type 3
cluster_1 <-setkey(data.table((wine_cluster[wine_cluster$clusters == 1,2:14])))
  #subgroup observation that are classified as claster 1
cluster_2 <-setkey(data.table((wine_cluster[wine_cluster$clusters == 2,2:14])))
  #subgroup observation that are classified as claster 2
cluster_3 <-setkey(data.table((wine_cluster[wine_cluster$clusters == 3,2:14])))
  #subgroup observation that are classified as claster 3
cluster1_type1<-na.omit(cluster_1[type_1,which=TRUE])
  #rows that in cluster 1 and originally typed as 1
cluster1_type2<-na.omit(cluster_1[type_2,which=TRUE])
  #rows that in cluster 1 and originally typed as 2
cluster1_type3<-na.omit(cluster_1[type_3,which=TRUE])
  #rows that in cluster 1 and originally typed as 3
cluster2_type1<-na.omit(cluster_2[type_1,which=TRUE])
```

```
  #rows that in cluster 2 and originally typed as 1
cluster2_type2<-na.omit(cluster_2[type_2,which=TRUE])
  #rows that in cluster 2 and originally typed as 2
cluster2_type3<-na.omit(cluster_2[type_3,which=TRUE])
  #rows that in cluster 2 and originally typed as 3
cluster3_type1<-na.omit(cluster_3[type_1,which=TRUE])
  #rows that in cluster 3 and originally typed as 1
cluster3_type2<-na.omit(cluster_3[type_2,which=TRUE])
  #rows that in cluster 3 and originally typed as 2
cluster3_type3<-na.omit(cluster_3[type_3,which=TRUE])
  #rows that in cluster 3 and originally typed as 3

method<-c(length(cluster1_type1),length(cluster1_type2),length(cluster1_type3)
          ,length(cluster2_type1),length(cluster2_type2),length(cluster2_type3)
          ,length(cluster3_type1),length(cluster3_type2),length(cluster3_type3))
  #count rows in groups cluster-type
matrix<-matrix(method,nrow = 3,ncol = 3, byrow=TRUE) #create matrix 3 by 3
colnames(matrix) <- c("Type 1","Type 2","Type 3") #set column names
rownames(matrix) <- c("Cluster 1","Cluster 2","Cluster 3") #set row names
matrix
```

```
##           Type 1 Type 2 Type 3
## Cluster 1     46      1      0
## Cluster 2      0     50     19
## Cluster 3     13     20     29
```

The output of following code is a 3 by 3 matrix that can be interpreted as follows: The intersection of row Cluster 1 and column Type 1 is the amount of observation in cluster 1 which are also type 1 in the original wine file. The sum of each row must be equal to the size of the clusters (47,62,69) and it is exactly what we observe.

As we see, cluster 1 mostly consist of type 1 wine, with only one type 2 wine. Cluster 2 can be interpreted as mostly type 2 wines, with 19 misgrouped type 3 wines. Cluster 3 is not that easy to interpret as it includes all types of wine. However, we can say that it is type 3 wine cluster.

K-means for scaled data

Using the same code, we use scaled data this time. We will not define our function again and use code above. The function returns clusters with 51, 61 and 66 observations. As we see from the plot, scaling the data improves clustering a lot. After scaling observation can be distinctly grouped with clear borders between clusters.

```
data(wine) #scale data
set.seed(0)
mywine_scaled <- as.matrix(scale(wine[,-1])) # remove the existing clustering

results <- cluster(mywine_scaled)

results[[1]][,]
```

```
##         Alcohol      Malic        Ash Alcalinity   Magnesium     Phenols
## [1,]  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724
## [2,]  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548
## [3,] -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891
##       Flavanoids Nonflavanoids Proanthocyanins      Color        Hue
## [1,]  0.97506900   -0.56050853      0.57865427  0.1705823  0.4726504
## [2,] -1.21182921    0.72402116     -0.77751312  0.9388902 -1.1615122
## [3,]  0.02075402   -0.03343924      0.05810161 -0.8993770  0.4605046
##         Dilution    Proline
## [1,]  0.7770551  1.1220202
## [2,] -1.2887761 -0.4059428
## [3,]  0.2700025 -0.7517257
```

```
sum(results[[2]][,2] == 1)
```

```
## [1] 62
```

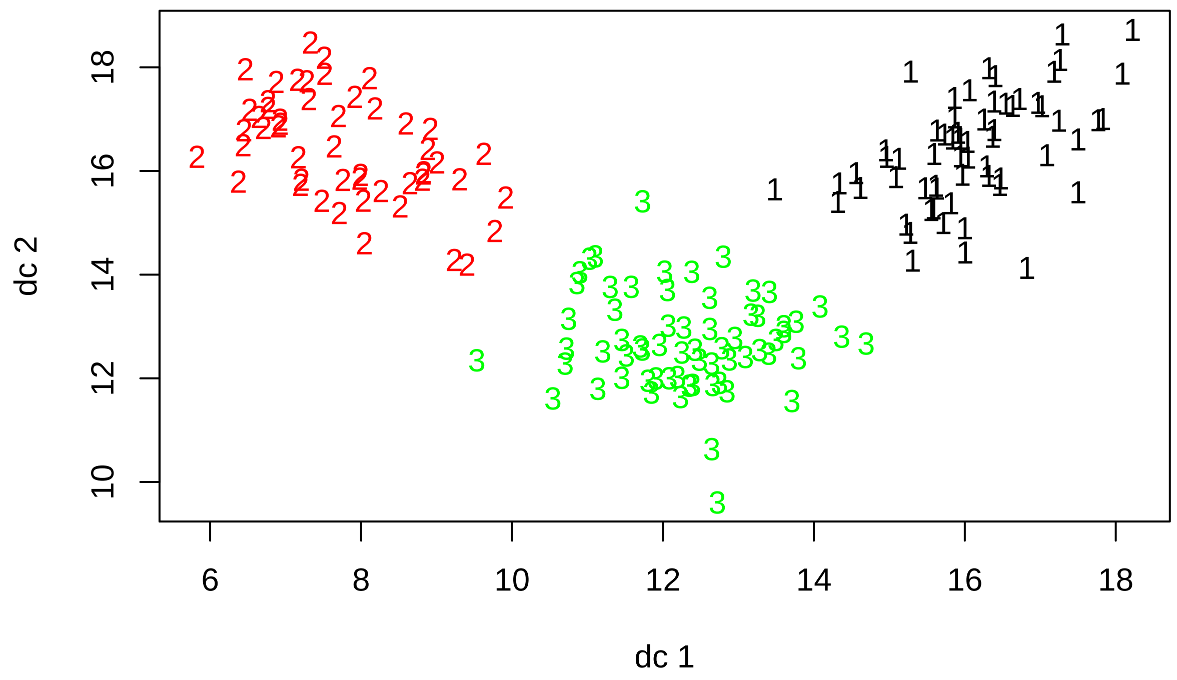
```
sum(results[[2]][,2] == 2)
```

```
## [1] 51
```

```
sum(results[[2]][,2] == 3)
```

```
## [1] 65
```

```
plotcluster(mywine, results[[2]][,2]) #plots clusters
```



Our method for checking quality of the clustering also shows improvement over unscaled data. For example, cluster 2 fully consist of type 2 wine. Cluster 1 became assosiated with type 3 wine with only 3 observations of type 2. Cluster 2 attributes type 1 wine with only 2 wrongly clustered obseration. Thus, after scaling the data, the k-means cluster algorithm wrongly classifies only 5 rows which is 3% of the total dataset.

```
##           Type 1 Type 2 Type 3
## Cluster 1     59      3      0
## Cluster 2      0      3     48
## Cluster 3      0     65      0
```

K-mean for iris dataset

In the following section we will run our previously defined function on the iris dataset and only new code will be commented, since we are running the same procedure.

Our function clusters the dataset into groups of 39, 50 and 61 observations. From the plot we can observe that cluster 1 is separated from other data, but clusters 2 and 3 are hard to distinguish.

```r
data(iris)
iris <- as.matrix(iris[,1:4]) # remove column with species
set.seed(421)

results <- cluster(iris)

results[[1]][,]
```

```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## [1,]     5.006000    3.428000     1.462000    0.246000
## [2,]     6.853846    3.076923     5.715385    2.053846
## [3,]     5.883607    2.740984     4.388525    1.434426
```

```r
sum(results[[2]][,2] == 1)
```

```
## [1] 50
```

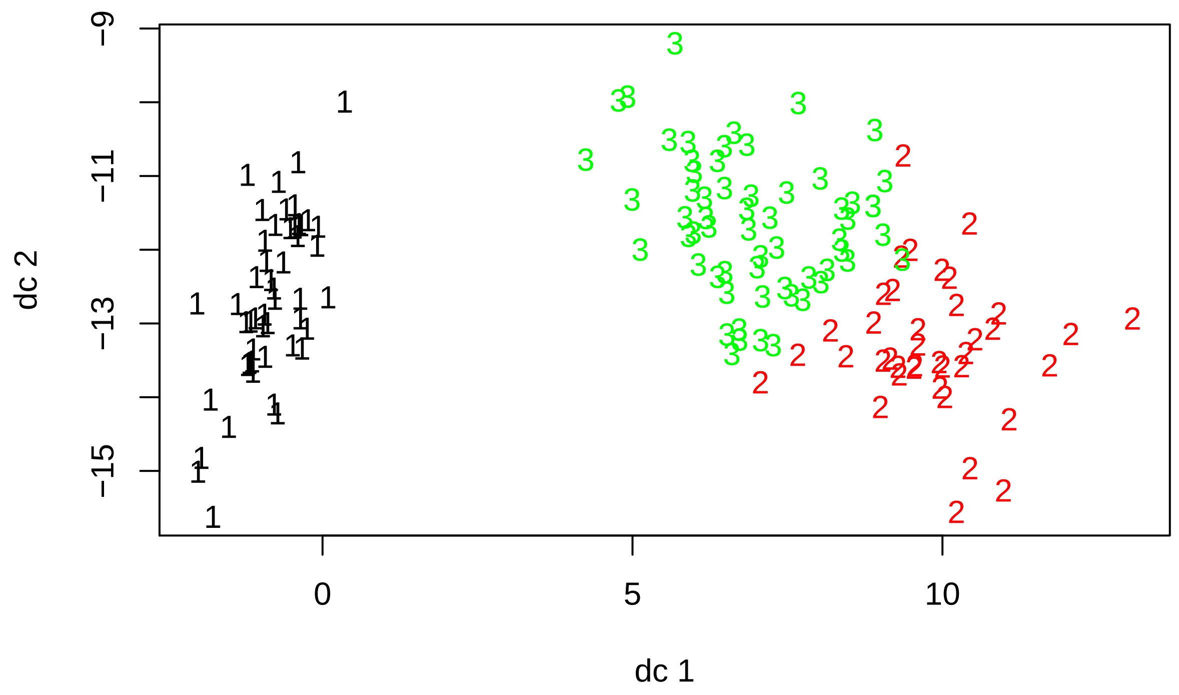
```r
sum(results[[2]][,2] == 2)
```

```
## [1] 39
```

```r
sum(results[[2]][,2] == 3)
```

```
## [1] 61
```

```r
plotcluster(iris, results[[2]][,2])
```



We can also notice that from this dataset, the k-means method clusters species 1 properly, but has more mistakes in grouping species 2 and 3 because 19 out of 150 observations were misgrouped (12,6% from total dataset).

```r
data(iris)

#Chaning properties of the data set so that our comparison method can be applied
iris <- as.matrix(iris)
iris[iris=="setosa"] <- 1 #recode variable species
iris[iris=="versicolor"] <- 2 #recode variable species
iris[iris=="virginica"] <- 3 #recode variable species
iris_clusters <- cbind(results[[2]][,2],iris) #add first column with cluster
colnames(iris_clusters)[1] <- "clusters"

iris_clusters<-as.data.frame(iris_clusters)
```

```
##           Species 1 Species 2 Species 3
## Cluster 1        50         0         0
## Cluster 2         0         3        36
## Cluster 3         0        47        16
```

K-means for iris scaled data

In the code below we first scale iris data in order to improve results of k-mean clustering. We receive groups of 50, 56, 44 observations. From the plot we can observe that again cluster 1 is separated from the majority of the data and thus clustered properly, but clusters 2 and 3 are still hard to distinguish.

```r
data(iris)
iris <- as.matrix(iris[,1:4]) # remove column with species
iris_scaled <- as.matrix(scale(iris)) #scale data
set.seed(421)

results <- cluster(iris_scaled)

results[[1]][,]
```

```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width
## [1,]   -1.0111914   0.8504137   -1.3006301  -1.2507035
## [2,]   -0.0113575  -0.8730834    0.3758167   0.3101145
## [3,]    1.1635361   0.1448178    0.9996766   1.0265628
```

```r
sum(results[[2]][,2] == 1)
```

```
## [1] 50
```

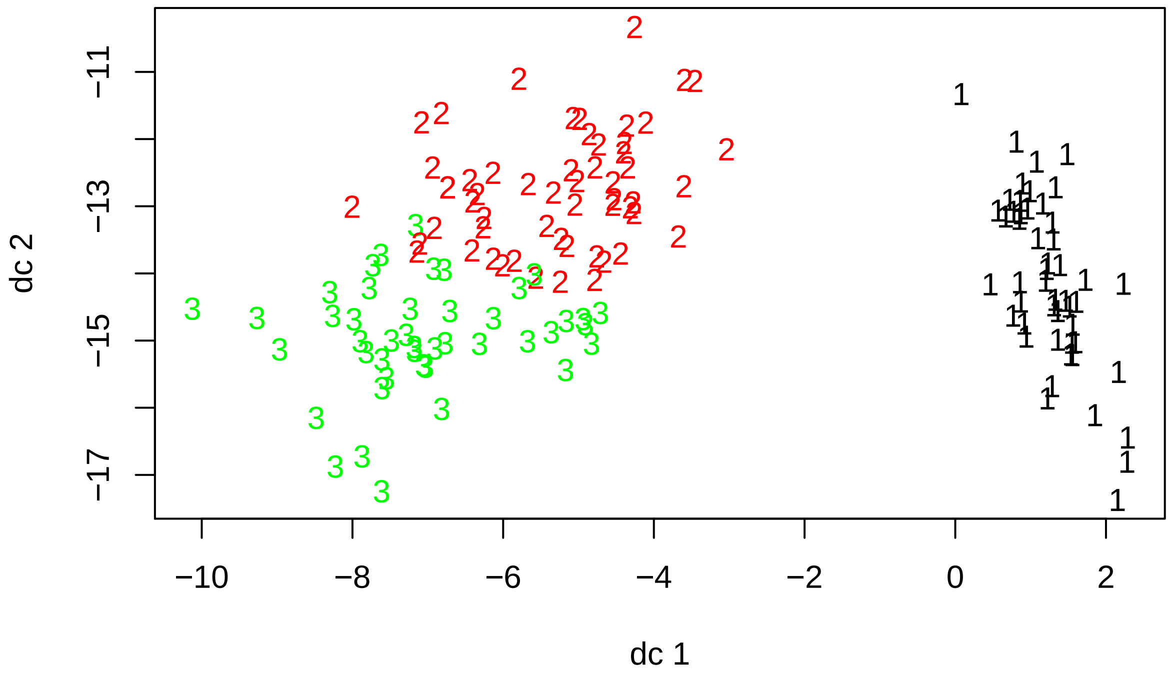
```r
sum(results[[2]][,2] == 2)
```

```
## [1] 56
```

```r
sum(results[[2]][,2] == 3)
```

```
## [1] 44
```

```r
plotcluster(iris, results[[2]][,2])
```



Corresponding with the result of the unscaled data, cluster 1 assosiated with species 1 is matched correctly, however scaling does not improve clustering for species 2 and 3. In fact, instead of 19 misgrouped observation we receive 30 (16% from dataset). Thus, we can conclude that scaling data does not always improve results of clustering and in some cases can negatively affect it.

```
##           Species 1 Species 2 Species 3
## Cluster 1        50         0         0
## Cluster 2         0        39        19
## Cluster 3         0        11        33
```