Stat 432 Homework 3

Taiga Hasegawa(taigah2)

Assigned: Feb 8, 2019; Due: 11:59pm Feb 15, 2019

Question 1 (dissimilarity matrix)

[3.5 points] Consider $n = 10$ samples with $p = 5$ features (you should copy this code to generate them):

```
set.seed(1)
x = matrix(rnorm(50), 10, 5)
x
```

```
##             [,1]        [,2]        [,3]        [,4]       [,5]
##  [1,] -0.6264538  1.51178117  0.91897737  1.35867955 -0.1645236
##  [2,]  0.1836433  0.38984324  0.78213630 -0.10278773 -0.2533617
##  [3,] -0.8356286 -0.62124058  0.07456498  0.38767161  0.6969634
##  [4,]  1.5952808 -2.21469989 -1.98935170 -0.05380504  0.5566632
##  [5,]  0.3295078  1.12493092  0.61982575 -1.37705956 -0.6887557
##  [6,] -0.8204684 -0.04493361 -0.05612874 -0.41499456 -0.7074952
##  [7,]  0.4874291 -0.01619026 -0.15579551 -0.39428995  0.3645820
##  [8,]  0.7383247  0.94383621 -1.47075238 -0.05931340  0.7685329
##  [9,]  0.5757814  0.82122120 -0.47815006  1.10002537 -0.1123462
## [10,] -0.3053884  0.59390132  0.41794156  0.76317575  0.8811077
```

Calculate the dissimilarity matrix for this set of data using:

- Euclidean distance
- ℓ_1 norm distance (we covered this in class, you can also google the definition)

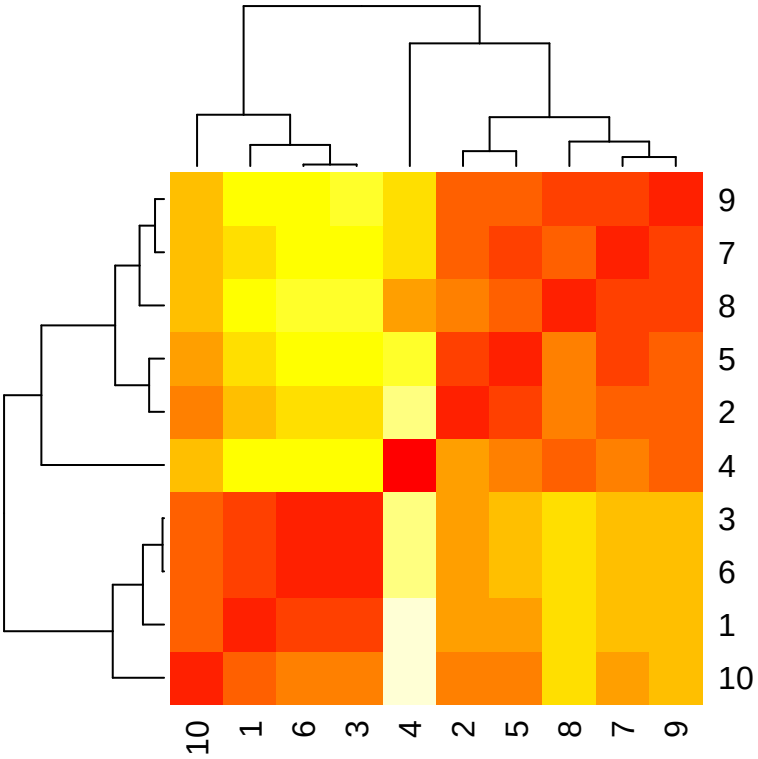
To show your result, do not print them. Use a heatmap (without reordering the columns and rows) to plot them separately. The position of each cell should match exactly the corresponding position in the dissimilarity matrix. Or more precisely, the ith row jth column in your heatmap should represent the (i, j)th element in the matrix. Moreover, the color of the (i, j)th element should be the same as the (j, i)th element due to symmetry. Figure this out by reading the `heatmap()` function documentation.

In the hierarchical clustering algorithm, we need to evaluate the distance between two groups of subjects. Consider the first 5 subjects as one group, and the rest subjects as another group. For this part, you cannot use the original data matrix `x`, only the dissimilarity matrix can be used. Do the following:

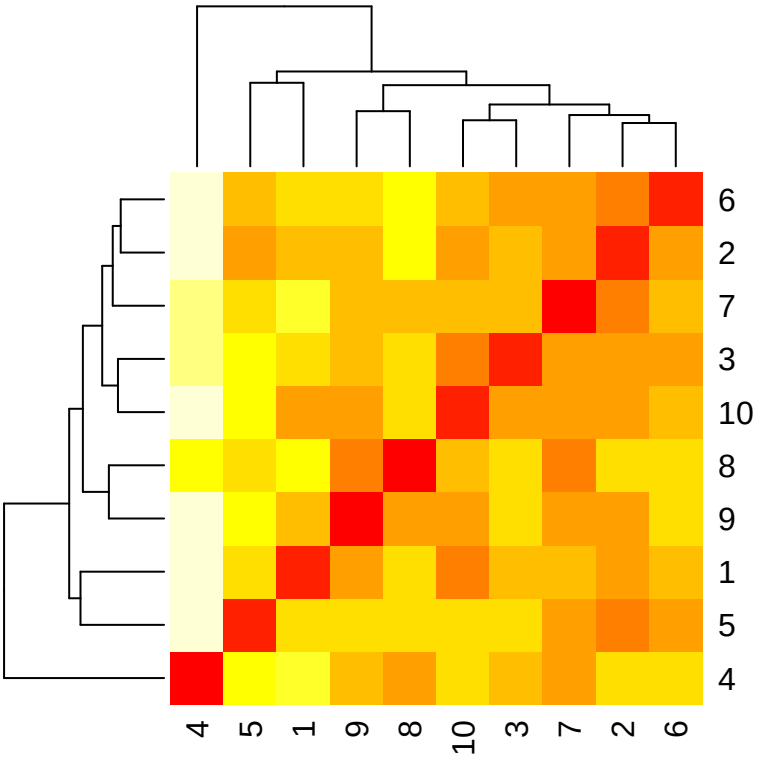
- Calculate the single linkage between the two groups using the dissimilarity matrix defined based on Euclidean distance.
- Calculate the average linkage between the two groups using the dissimilarity matrix defined based on the ℓ_1 norm distance.

```
#calculate l1 norm
l1norm=matrix(rep(0,100),10,10)
for (i in 1:10){
  for(j in 1:10){
    l1norm[i,j]=abs(x[i]-x[j])
  }
}
```

```
#plot the heatmap of l1 norm
heatmap(l1norm)
```




```
#plot the heatmap of euclidean distance
euclidean=as.matrix(dist(x,diag=TRUE,upper=TRUE))
heatmap(euclidean)
```




```
#calculate the single linkage
hclust1=hclust(dist(x), method = "single")
```

```
hclust1$height
```

```
## [1] 1.266370 1.431972 1.484476 1.549617 1.565357 1.626220 1.652735 1.728900
## [9] 3.094436
```

```
#calculate the average linkage
l1norm_dist=as.dist(l1norm)
hclust2=hclust(l1norm_dist,method="average")
hclust2$height
```

```
## [1] 0.01516023 0.08835230 0.14586445 0.20159469 0.20671950 0.34393616
## [7] 0.45546188 1.10992204 1.62564225
```

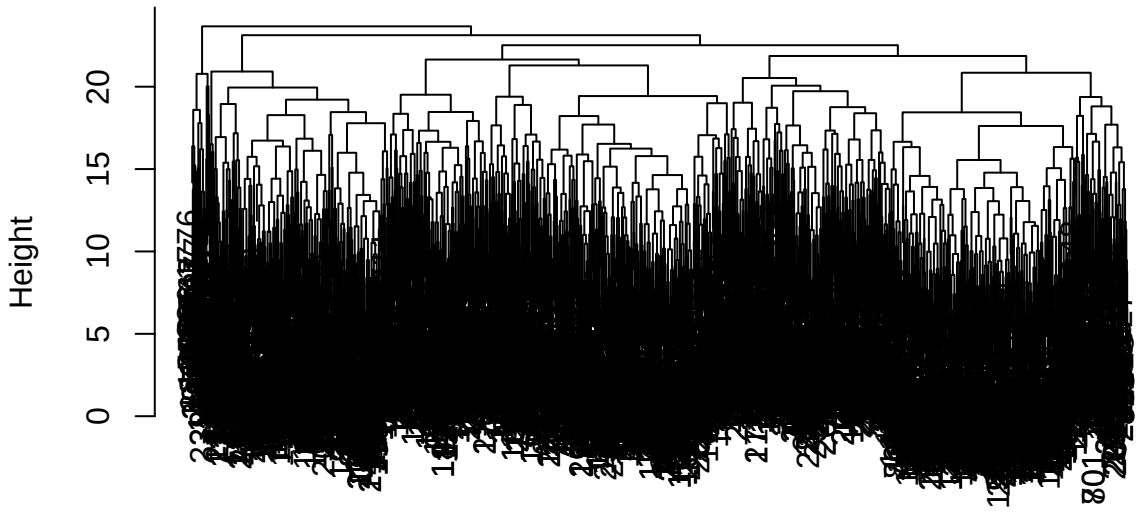
Question 2 (hierarchical clustering)

[3 points] Load the handwritten zip code digits data from the `ElemStatLearn` package. There are two datasets: `zip.train` and `zip.test`. Take only the observations with true digits 2, 4, 6 and 8 from the training data.

```
    library(ElemStatLearn)
    even.train.x = zip.train[zip.train[,1] %in% c(2, 4, 6, 8), -1]
    even.train.y = as.factor(zip.train[zip.train[,1] %in% c(2, 4, 6, 8), 1])
```

Run a hierarchical clustering (with the default Euclidean distance function `dist()`) to the training dataset `even.train.x` and plot the dendrogram. Read the documentation of the `cutree()` function, and use it to define 4 clusters from your hierarchical clustering tree. Does this clustering result match the true digits well? Produce some summaries regarding how well they match. For example, the `table()` function. Now, change the linkage method to `single` and `average`, which method seems to match the true labels the best if they all just use 4 clusters? You can use `table(cutree(hclust_fit),true_labels)` to compare results.

```
#hierarchical with complete linkage
hclust3=hclust(dist(even.train.x))
plot(hclust3)
```




```
#summary of complete linkage
hclust.cut <- cutree(hclust3, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1 334   8 635 120
##          2  21 467   4   4
##          3 371 177  24 376
##          4   5   0   1  42
```

4 and 6 were classified well but 2 and 8 were not.

```
#hierarchical with single linkage
hclust4=hclust(dist(even.train.x),method="single")
hclust.cut <- cutree(hclust4, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1 729 650 664 542
##          2   0   1   0   0
##          3   2   0   0   0
##          4   0   1   0   0
```

This couldn't classify all digits at all.

```
#hierarchical with average linkage
hclust5=hclust(dist(even.train.x),method="average")
hclust.cut <- cutree(hclust5, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1 728 652 663 523
##          2   0   0   1  19
##          3   1   0   0   0
##          4   2   0   0   0
```

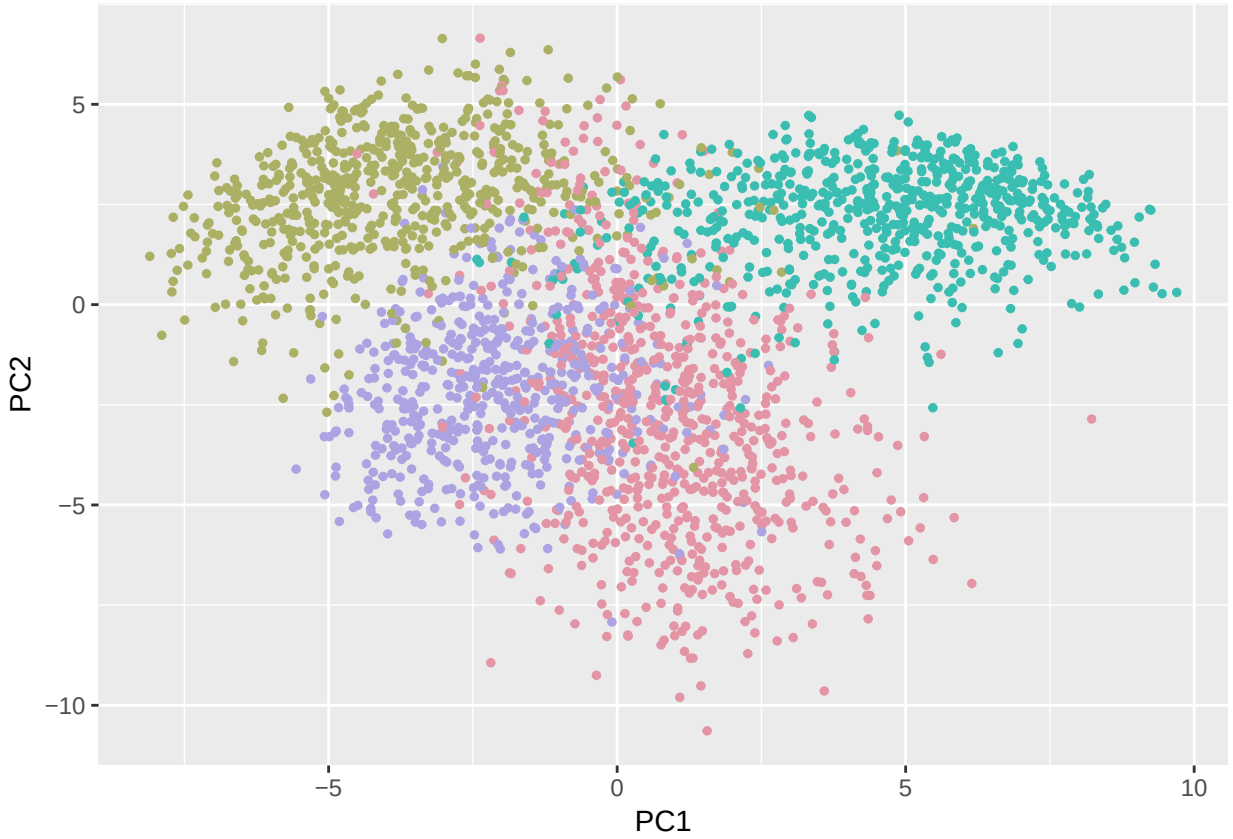
This also couldn't classify digits at all. Complete methods seemed to be best to classify these four digits.

Question 3 (PCA and clustering)

[3.5 points] Use the same dataset defined in Question 2), perform PCA on the pixels. Plot all observations on the first two principal components and color the observations based on their true digits.

Take the first 3 principal components from the PCA and treat them as 3 new covariates. Hence, you have a new dataset with 3 variables, and the same number of observations as the original data. Now, perform hierarchical clustering again on this new dataset using all three linkage methods. Which one seems to match the true labels the best? You should again demonstrate some necessary results to support your argument. Is this an improvement from the original hierarchical clustering method performed on the 256 pixels? Comment on your findings.

```
#Plot all observations on the first two principal components and color the observations based on their
zip_pc = prcomp(even.train.x)
library(ggplot2)
library(colorspace)
ggplot(data = data.frame(zip_pc$x), aes(x=PC1, y=PC2)) +
  geom_point(color = rainbow_hcl(4)[even.train.y], size = 1)
```




```
#complete linkage
pc_hclust=hclust(dist(zip_pc$x[, 1:3]))
```

```r
hclust.cut <- cutree(pc_hclust, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1  80   2 546   1
##          2  88 537  45  14
##          3 170 101  44 503
##          4 393  12  29  24
```

```r
#average linkage
pc_hclust=hclust(dist(zip_pc$x[, 1:3]),method="average")
hclust.cut <- cutree(pc_hclust, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1  14   6 522   5
##          2  87 611  21  40
##          3  86  16   4 457
##          4 544  19 117  40
```

```r
#single linkage
pc_hclust=hclust(dist(zip_pc$x[, 1:3]),method="single")
hclust.cut <- cutree(pc_hclust, 4)
table(hclust.cut, even.train.y)
```

```
##           even.train.y
## hclust.cut   2   4   6   8
##          1 729 651 664 542
##          2   1   0   0   0
##          3   0   1   0   0
##          4   1   0   0   0
```

2 and 4 were classified better when we used average linkage but 6 and 8 were classified better when we used complete linkage. Single linkage performed terrebile. There was an improvement from the original hierarchical clustering method performed on the 256 pixels.