Stat 432 Homework 3

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

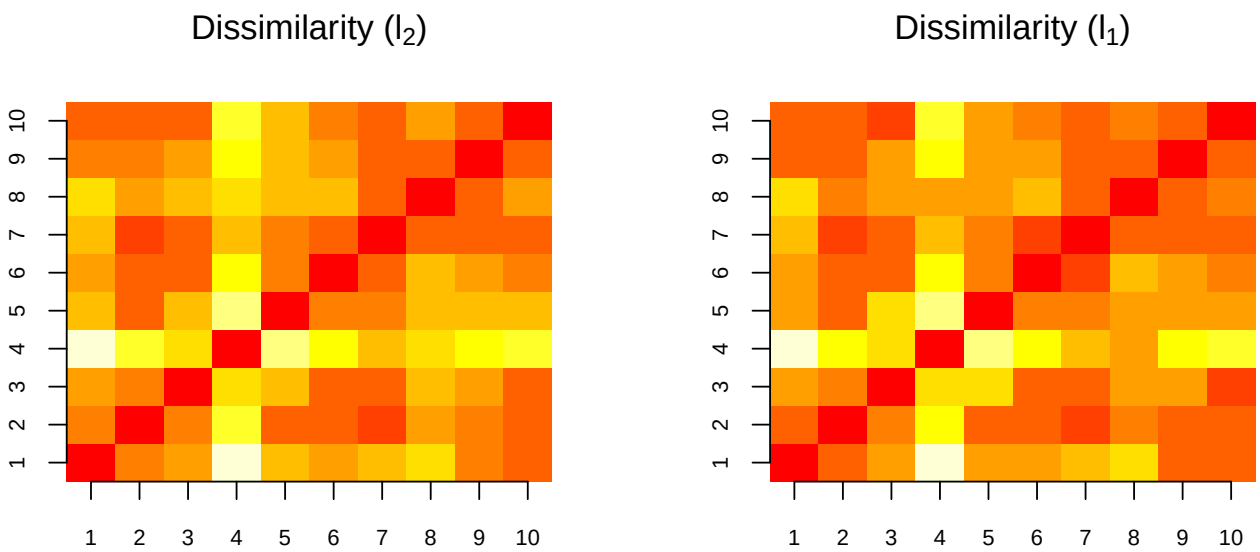
Question 1 (dissimilarity matrix)

[3 points] Consider $n = 10$ samples with $p = 5$ features:

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

We can calculate the dissimilarity matrices using `dist` function and plot them using `heatmap` function. 1 point: .5 for each figure.

```
# calculate the dissimilarity matrix
# Euclidean distance
distX2=as.matrix(dist(x,diag=T,upper=T))
distX1=as.matrix(dist(x,method='manhattan',diag=T,upper=T))
# par(mfrow=c(1,2),oma=c(2,0,0,0),mar=c(2,1,1,1)) # not easy to plot them side by side
# heatmap(distX2,Rowv=NA,Colv=NA)
# title(main=expression(paste('Dissimilarity (l'['2'],')')))
# heatmap(distX1,Rowv=NA,Colv=NA)
# title(main=expression(paste('Dissimilarity (l'['1'],')')))
# alternative plots
library(gplots)
layout(matrix(c(1,2),1,2),respect = T)
image(distX2,axes=F)
axis(1, at = seq(0,1,length.out=10), labels=1:10,cex.axis=.75)
axis(2, at = seq(0,1,length.out=10), labels=1:10,cex.axis=.75)
title(main=expression(paste('Dissimilarity (l'['2'],')')))
image(distX1,axes=F)
axis(1, at = seq(0,1,length.out=10), labels=1:10,cex.axis=.75)
axis(2, at = seq(0,1,length.out=10), labels=1:10,cex.axis=.75)
title(main=expression(paste('Dissimilarity (l'['1'],')')))
```



Now we calculate the linkages based on dissimilarity matrices. Based on the definition of single linkage, we should look for the minimal entry in `distX2[1:5,6:10]`. On the other hand, the avage linkage should be

calculated as the average of entries in `distX1[1:5,6:10]`. 2 points: 1 for each linkage result.

```
# single linkage based on the dissimilarity matrix with Euclidean distance.
(single_lnkg=min(distX2[1:5,6:10]))
```

```
## [1] 1.26637
```

```
# average linkage based on the dissimilarity matrix with the $\ell_1$ norm distance.
(avg_lnkg=mean(distX1[1:5,6:10]))
```

```
## [1] 4.709157
```

Therefore, with the specified partition, the single linkage based on Euclidean distance between the two clusters is 1.26637, and the average linkage based on the ℓ_1 norm distance between the two clusters is 4.7091573.

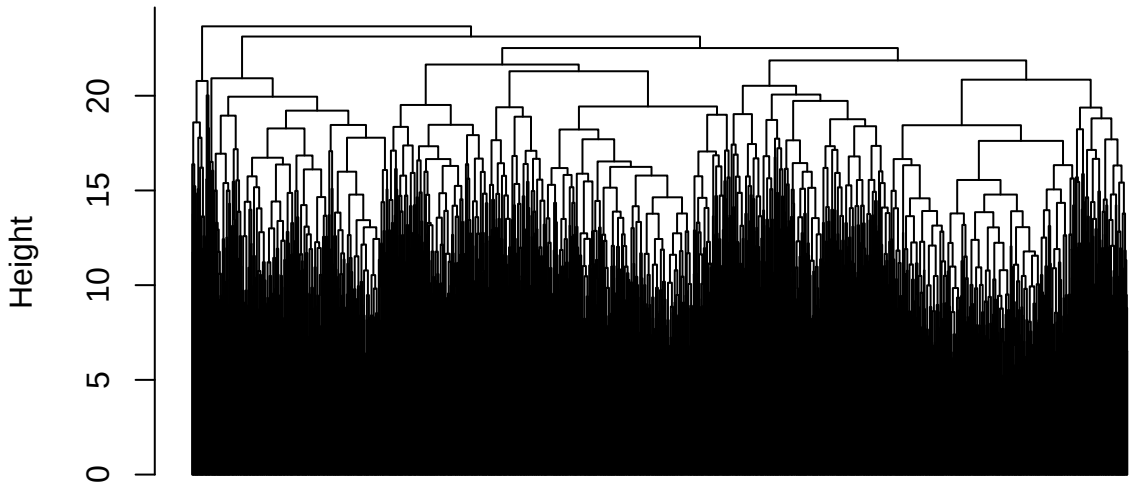
Question 2 (hierarchical clustering)

[3.5 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.
0.5 points for the dendrogram; 1 point for each confusion matrix.

```
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])
```

Now we use `hclust` to do a hierachical clustering based on Euclidean distance and plot the dendrogram. The figure can differ a little from the solution. Then we cut the hierachical tree at level 4 and compare the result with the true labels in the confusion matrix.

```
cl = hclust(dist(even.train.x))
# plot the dendrogram
plot(cl, hang = -1, labels=FALSE)
```




```
# plot the confusion matrix
table(cutree(cl, k = 4), even.train.y)
```

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

This solution is acceptable. Note, `hclust` is a {**clustering**} **algorithm, not a {classification} algorithm. Therefore, the clustering assignment** ($k = 4$) **may not necessarily coincide with the true labels. However, we can relabel the clusters to minimize the number of the mis-labeled, i.e.** ($1 \to 6, 2 \to 4, 3 \to 2, 4 \to 8$)**:**

```
# relabel the cluster labels
cut4=as.factor(cutree(cl, k = 4))
levels(cut4)<-c(6,4,2,8)
# plot the confusion matrix after relabeling
table(cut4, even.train.y)[c(3,2,1,4),]
```

```
##     even.train.y
## cut4   2   4   6   8
##    2 371 177  24 376
##    4  21 467   4   4
##    6 334   8 635 120
##    8   5   0   1  42
```

This is an OK result, but there are still many instances that mismatch the true labels.

Now we consider the `single` linkage and `average` linkage.

```
# single linkage
cl_sing = hclust(dist(even.train.x),method='single')
# plot(cl_sing, hang = -1)
table(cutree(cl_sing, k = 4), even.train.y)
```

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

```
# average linkage
cl_avg = hclust(dist(even.train.x),method='average')
# plot(cl_avg, hang = -1)
table(cutree(cl_avg, k = 4), even.train.y)
```

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

These two linkages return results much worse than the default linkage (`complete`), even after we relabel the cluster assignments.

```
# plot the confusion matrix after relabeling the cluster labels for single linkage
cut4_sing=as.factor(cutree(cl_sing, k = 4))
levels(cut4_sing)<-c(2,4,6,8)
table(cut4_sing, even.train.y)
```

```
##          even.train.y
## cut4_sing   2   4   6   8
##         2 729 650 664 542
##         4   0   1   0   0
##         6   2   0   0   0
##         8   0   1   0   0
```

```
# plot the confusion matrix after relabeling the cluster labels for average linkage
cut4_avg=as.factor(cutree(cl_avg, k = 4))
levels(cut4_avg)<-c(2,8,6,4)
table(cut4_avg, even.train.y)[c(1,4,3,2),]
```

```
##         even.train.y
## cut4_avg   2   4   6   8
##        2 728 652 663 523
##        4   2   0   0   0
##        6   1   0   0   0
##        8   0   0   1  19
```

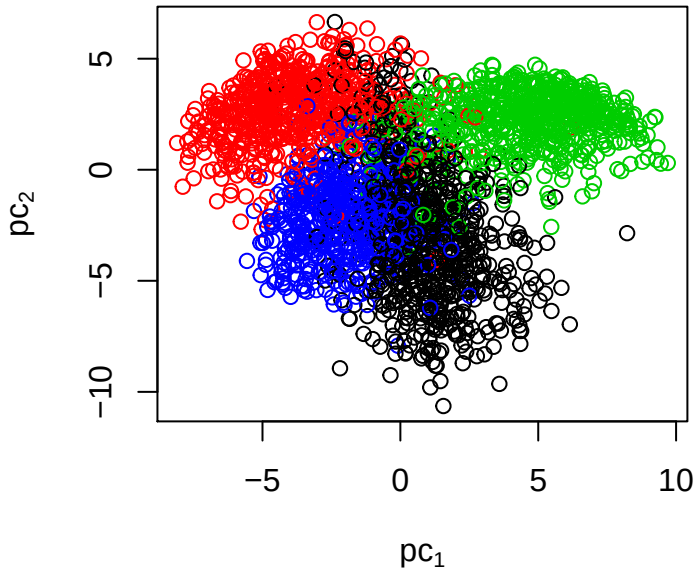
Results without relabeling are acceptable. Full credit should be given as long as you comment that these two linkages return worse resutls compared with the compelete linkage.

Question 3 (PCA and clustering)

[3.5 points] Now we perform PCA on the pixels and plot all observations on the first two principal components and color the observations based on their true digits. 0.5 for ploting the

principal components; 1 point for each confusion matrix.

```
pc = prcomp(even.train.x)
plot(pc$x[, 1], pc$x[,2], col = even.train.y, xlab=expression('pc'[1]), ylab=expression('pc'[2]))
```




```
# plot(pc$x[, 3], pc$x[,4], col = even.train.y) # less informative
```

Now we repeat Question 2 with the first 3 pricinpal components. For each linkage, we show the results after relabeling the cluster.

```
# complete linkage
cl = hclust(dist(pc$x[, 1:3]))
# plot(cl, hang = -1)
cut4=as.factor(cutree(cl, k = 4))
# table(cut4, even.train.y) # also acceptable
levels(cut4) <- c(6,4,8,2)
table(cut4, even.train.y)[c(4,2,1,3),]
```

```
##     even.train.y
## cut4   2   4   6   8
##    2 393  12  29  24
##    4  88 537  45  14
##    6  80   2 546   1
##    8 170 101  44 503
```

```
# single linkage
cl_sing = hclust(dist(pc$x[, 1:3]), method = "single")
cut4_sing=as.factor(cutree(cl_sing, k = 4))
levels(cut4_sing) <- c(2,6,4,8)
table(cut4_sing, even.train.y)[c(1,3,2,4),]
```

```
##          even.train.y
## cut4_sing   2   4   6   8
##         2 729 651 664 542
##         4   0   1   0   0
##         6   1   0   0   0
##         8   1   0   0   0
```

```r
# average linkage
cl_avg = hclust(dist(pc$x[, 1:3]), method = "average")
cut4_avg=as.factor(cutree(cl_avg, k = 4))
levels(cut4_avg) <- c(6,4,8,2)
table(cut4_avg, even.train.y)[c(4,2,1,3),]
```

```
##          even.train.y
## cut4_avg   2   4   6   8
##        2 544  19 117  40
##        4  87 611  21  40
##        6  14   6 522   5
##        8  86  16   4 457
```

Note that, the preprocessing step of PCA improves the cluster labeling results, especially for the `complete` and `average` linkages. Among the three linkages, the `average` linkage attains the optimal result.

Results without relabeling are acceptable. Full credit should be given as long as you compare the results with those based on the full data and you conclude the 'average' linkage attains the best result. Take 1 point off for each missing of the two comments.