Lab 5a: K-Means and Hierarchical Clustering

First Name: Siyuan **Last Name**: Yin **NetID**: sy652

Lab 5 will be split into two parts, 5a for K-means and hierarchical clustering and 5b for PCA. Lab 5a is due April 21 by 4:30pm along with Lab 5b. For Lab 5a, submit your code for 'NCI60 data example' as well as your answers to the problems. You may attach additional pages for your answers.

In this lab, we will learn how to use `R` to perform K-means and hierachical clustering. We will first illustrate both techniques on an simple example, which allows you to learn the basic commands of these 2 methods. After that you will be able to apply these method youself on the data set `NCI60`, the cancer cell line microarray data, which consists of 6,830 gene expression measurements on 64 cancer cell lines. You can import the `NCI60` data set by importing package `ISLR` (using command `library(ISLR)`) .

K-Means Clustering

The function `kmeans()` performs K-means clustering in `R`.

For demonstration purpose, let's first generate some fake data in $\mathbb{R}^2$ in which there truly are 4 clusters in the data. We do this by shifting the means of the points around. K-means work in any dimension, but is most fun in 2 because we can plot pictures.

```
> set.seed(101)
> x=matrix(rnorm(100*2),100,2)
> xmean=matrix(rnorm(8,sd=4),4,2)
> which=sample(1:4,100,replace=TRUE)
> x=x+xmean[which,]
> plot(x,col=which,pch=19)
```

Then we apply function `kmeans()` to the data with $K = 4$ and `nstart` $= 15$.

```
> set.seed(4)
> km.out=kmeans(x,4,nstart=15)
> km.out
> plot(x, col=km.out$cluster, pch=1, cex=2,lwd=2)
> points(x, col=c(3,1,4,2)[which], pch=19)
```

We see that the K-means clustering perfectly separated the observations into 4 clusters with only 2 miss match.

To run the `kmeans()` function in R with multiple initial cluster assignments, we use the `nstart` argument. If a value of `nstart` greater than one is used, then K-means clustering will be performed using multiple random assignments in Step 1 of Algorithm 10.1, and the `kmeans()` function will report only the best results.

Compare the 'toal within-cluster variability' using `nstart`=1 to `nstart`=20 and report your answer (use random seed(1)).

The within-cluster variability is much larger with nstart value of 20 than nstart value of 1 except for the fourth cluster, where nstart value of 1 has much higher within-cluster variability.

Note that `km.out$tot.withinss` is the total within-cluster sum of squares, which we seek to minimize by performing K-means clustering. The individual within-cluster sum-of-squares are contained in the vector `km.out$withinss`.

We strongly recommend always running K-means clustering with a large value of `nstart`, such as 20 or 50, since otherwise an undesirable local optimum may be obtained.

When performing K-means clustering, in addition to using multiple initial cluster assignments, it is also important to set a random seed. This way, the initial cluster assignments can be replicated, and the K-means output will be fully reproducible.

Hierachical Clustering

The `hclust()` function implements hierarchical clustering in R.

For demonstration we will use same data generated in the first part and apply hierachical clustering using complete, single, and average linkage, with Euclidean distance as the dissimilarity measure. The `dist()` function is used to compute the 100×100 inter-observation Euclidean distance matrix.

```
> hc.complete=hclust(dist(x), method="complete")
> hc.average=hclust(dist(x), method="average")
> hc.single=hclust(dist(x), method="single")
```

Then we can now plot the dendrograms obtained using each linkage. The numbers at the bottom of the plot identify each observation.

```
> plot(hc.complete,main="Complete Linkage", cex=.9)
> plot(hc.average, main="Average Linkage", cex=.9)
> plot(hc.single, main="Single Linkage", cex=.9)
```

Now let's compare this with the actual clusters in the data. To determine the cluster labels for each observation associated with a given cut of the dendrogram, we can use the `cutree()` function. Here we cut the tree at level 4.

```
> hc.cut=cutree(hc.complete,4)
```

We can use `table()` to see how well they match the actual clusters. Report your table of matching result using complete linkage clustering (like we did in logistic regression).

```
     which
hc.cut  1  2  3  4
     1  0  0 30  0
     2  1 31  0  2
     3 17  0  0  0
     4  0  0  0 19,    97/100 = 97%, the accuracy is 97%.
```

Visualize your result using average linkage clustering like we did in K-means. That is do a plot with both your clusters using average linkage and the actural clusters on the same graph.

Let's now examine hierachical clustering using correlation-based distance.

Correlation-based distance can be computed using the `as.dist()` function, which converts an arbitrary square symmetric matrix into a form that the `hclust()` function recognizes as a distance matrix. However, this only makes sense for data with at least three features since the absolute value of correlation between any two observations with measurements on two features is always 1. Why?

With two observations and two features, we can fit a linear model to them and obtain a correlation of either 1 or -1, and therefore the absolute value of correlation between any two of them is always 1.

Hence we will generate a 3-dim data set as an example.

```
> x3=matrix(rnorm(30*3), ncol=3)
> dd=as.dist(1-cor(t(x3)))
> plot(hclust(dd, method="complete"))
```

NC160 Data Example

Now your are familiar with the commands of K-means and hierachical clustering, let's apply them on a real data set `NCI60`.

```
> library(ISLR)
> nci.labs=NCI60$labs
> nci.data=NCI60$data
```

Each cell line is labeled with a cancer type. We do not make use of the cancer types in performing clustering, as these are unsupervised techniques. But after performing clustering, we will check to see the extent to which these cancer types agree with the results of these

unsupervised techniques.

To begin, we standardize the variables to have mean zero and standard deviation one. this step is optional and should be performed only if we want each gene to be on the same scale.

```
> sd.data=scale(nci.data)
```

Now perform hierarchical clustering of the observations using complete, single, and average linkage, where Euclidean distance is used as the dissimilarity measure. Plot your dendrograms of each method. Given the result, which linkage method you think is preferred in this case and why? (You may attach additional page)

Using complete linkage would be preferred in this case because 1) the cluster is more balanced than the other two method 2) the distances among clusters are larger than the other two and 3) the range of height is smaller than the other two. This three observations indicate a better clustering result using complete linkage.

For following analysis we will use complete linkage hierarchical clustering.
Try cutting the dendrogram at the height that will yield 5 number of clusters, create a table of your clustering result vs. `nci.labs`. Do you find any clear patterns in your result? For example, which types of cell lines tends to lie in the same cluster? and which types tends to spread over different clusters?

```
             hc.cut1
nci.labs      1 2 3 4 5
  BREAST      0 3 0 2 2
  CNS         3 2 0 0 0
  COLON       2 0 0 5 0
  K562A-repro 0 0 1 0 0
  K562B-repro 0 0 1 0 0
  LEUKEMIA    0 0 6 0 0
  MCF7A-repro 0 0 0 1 0
  MCF7D-repro 0 0 0 1 0
  MELANOMA    2 0 0 0 6
  NSCLC       7 1 0 0 1
  OVARIAN     6 0 0 0 0
  PROSTATE    2 0 0 0 0
  RENAL       8 1 0 0 0
  UNKNOWN     1 0 0 0 0
```

There are some subtle pattern in the result. For example, MELANOMA, NSCLC, OVARIAN, PROSTATE, AND RENAL tend to lie in the same cluster. The rest of the cells tend to spread over different clusters.

We claimed earlier that K-means clustering and hierarchical clustering with the dendrogram cut to obtain the same number of clusters can yield very different results. How do these `NCI60` hierarchical clustering results compare to what we get if we perform K-means clustering with $K = 5$?
Perform K-means clustering on the data set with random seed(3), `nstart`=20 and create a table of the clustering result using hierachical vs. using K-means. Find clusters that emerge consistently and describe how you find them.

```
hc.cut1
  1  2  3  4  5
1 13  7  0  0  0
2  0  0  0  4  0
3  1  0  0  0  8
4  0  0  8  0  0
5 17  0  0  5  1
```

Some data points such as 13 and 8 in the fourth row are are consistently clustered in to the same cluster by different methods. This is found by checking data points across methods.

Finally, use correlation-based distance instead of Euclidean distance and perform hierachical clustering again (complete linkage), try finding the cluster that emerge consistently with the previous two methods (i.e., complete linkage hierarchical clustering with Euclidean distance, and K-means).

We can see some cluster that emerges consistently with the previous two methods.

```
          hc.cut1
hc.cut_cor  1  2  3  4  5
       1  7  6  0  0  0
       2  3  1  0  0  8
       3 11  0  0  0  0
       4  5  0  8  0  1
       5  5  0  0  9  0

hc.cut_cor  1  2  3  4  5
       1 11  0  0  0  2
       2  3  0  9  0  0
       3  6  0  0  0  5
       4  0  0  0  8  6
       5  0  4  0  0 10
```