Stat 437 Project 1

Nam Jun Lee (11606459)

Contents

1 Introduction

1.1 Using Methods

This project is a discriminant analysis for (1) K-means clustering, (2) hierarchical clustering, and (3) nearest-neighbor classifier and (4) discriminant analysis classification, and the results are explained using the visualization techniques learned.

(1) K-mean clustering: It is one of the clustering techniques that groups into K clusters using the average of each cluster.

(2) Hierarchical clustering: It is one of the clustering techniques that reduces the number of clusters by selecting two clusters with the highest similarity or closest distance among several clusters and combining them into one.

(3) Nearest Neighbor Classifier: When new data is input, it determines the type of new data based on the nearest center point.

(4) Discriminant analysis: Using samples from two or more populations, we find criteria to determine which population these samples were extracted from.

1.2 Data Information

The data files are "labels.csv" that contains the cancer type for each sample, and "data.csv" that contains the "gene expression profile" (i.e., expression measurements of a set of genes) for each sample. Here each sample is for a subject and is stored in a row of "data.csv". In fact, the data set contains the gene expression profiles for 801 subjects, each with a cancer type, where each gene expression profile contains the gene expressions for the same set of 20531 genes. The cancer types are: "BRCA", "KIRC", "COAD", "LUAD" and "PRAD". In both files "labels.csv" and "data.csv", each row name records which sample a label or observation is for.

1.3 Seven Tasks

In the case of **task A1**, data preprocessing is performed before applying **k-means clustering** and **hierarchical clustering**.
In **task A2**, the gene samples are randomly extracted, compared with the number of clusters from Gap statistics and the number of clusters from total with in-cluster sum of squares, and the results are analyzed. In addition, the two clustering using the number of clusters from total with in-cluster sum of squares is compared and the results are interpreted after randomly extracting the gene sample once more. Finally, we compare how the number of genetic samples affects the **k-means clustering** results.
In **task A3**, genetic samples are randomly extracted and **hierarchical clustering** with average, single, and complete linkages are applied and visualized. In addition, the dendrogram obtained from the average linkage is cut, the height is calculated, and compared with the label.
In the case of **task B1**, data preprocessing is performed before applying the quadratic discriminant analysis. In the case of **task B2**, a **quadratic discriminant analysis** is performed to see if each

observation follows a Gaussian distribution given a class or group membership of the observation. A quadratic discriminant analysis model is created using the training set, and the obtained model is classified into two labels. And the results are interpreted through the classification error table.
In the case of **task B3**, after selecting a random gene sample, a **quadratic discriminant analysis** model is created using a training set, the obtained model is classified into two labels, and a classification error table is generated. Thereafter, the difference found compared to the classification error table obtained in task B2 will be described.
In the case of **task B4**, after selecting a random gene sample, apply the **KNN** method to the test data using a training set, and then perform the task of classifying each observation into one of the cancer types. After that, a classification error table is generated, compared with the classification error table obtained in task B3, and the difference is described.

2 Methods & Results

2.1 Task A. Clustering

For this task, you need to apply k-means and hierarchical clustering to cluster observations into their associated cancer types, and report your findings scientifically and professionally. Your laptop may not have sufficient computational power to implement k-means and hierarchical clustering on the whole data set, and genes whose expressions are zero for most of the subjects may not be so informative of a cancer type.

Please use `set.seed(123)` for random sampling via the command `sample`, random initialization of `kmeans`, implementing the gap statistic, and any other process where artificial randomization is needed.

2.1.1 Task A1

- Filter out genes (from "data.csv") whose expressions are zero for at least 300 subjects, and save the filtered data as R object "gexp2".
- Use the command `sample` to randomly select 1000 genes and their expressions from "gexp2", and save the resulting data as R object "gexp3".
- Use the command `sample` to randomly select 30 samples and their labels from the file "labels.csv", and save them as R object "labels1". For these samples, select the corresponding samples from "gexp3" and save them as R object "gexpProj1".
- Use the command `scale` to standard the gene expressions for each gene in "gexpProj1", so that they have sample standard deviation 1. Save the standardized data as R object "stdgexpProj1".

2.1.2 Task A2

2.1.2.1 Part 1 Randomly pick 50 genes and their expressions from "stdgexpProj1", and do the following to these expressions: apply the "gap statistic" to estimate the number of clusters, apply K-means clustering with the estimated number of clusters given by the gap statistic, visualize

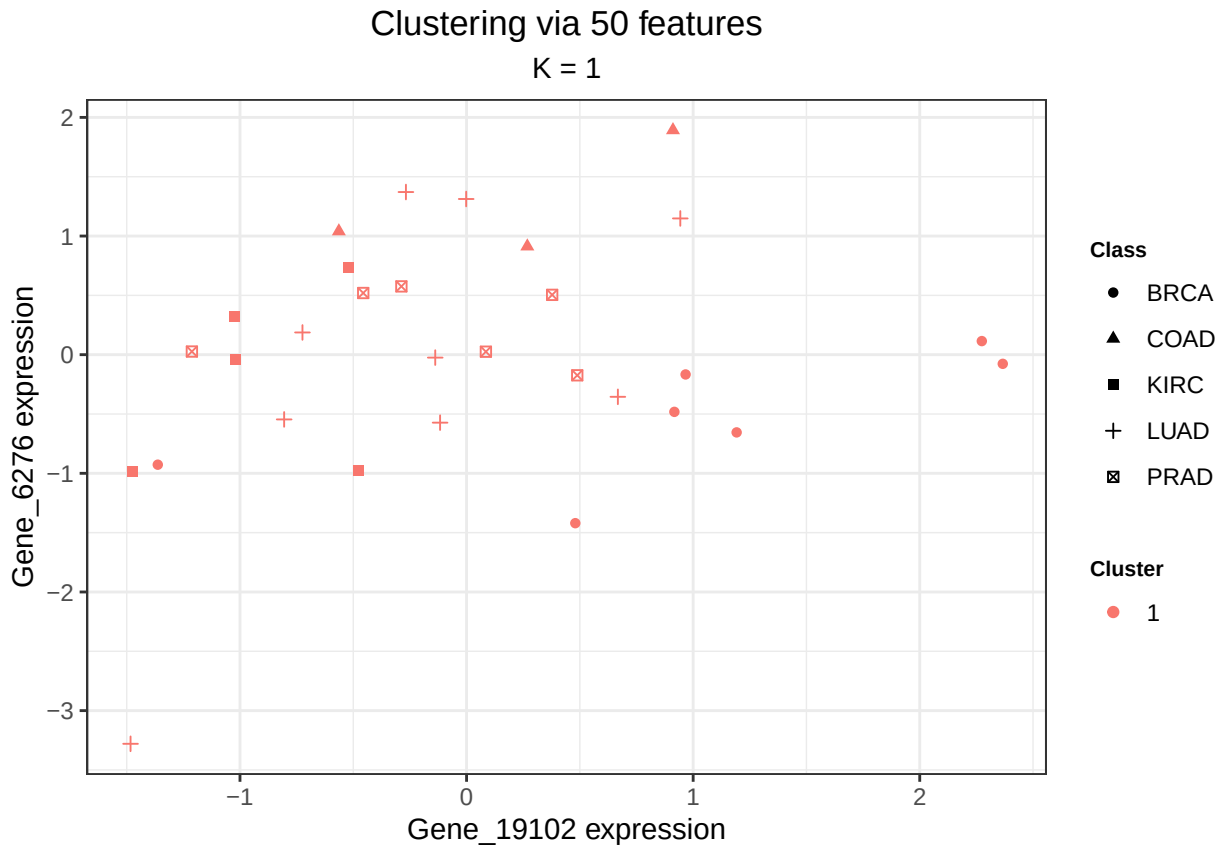
the classification results using techniques given by "LectureNotes3_notes.pdf.pdf", and provide a summary on classification errors. You may use the command `table` and "labels1" to obtain classification errors. Note that the cluster numbering given by `kmeans` will usually be coded as follows:

```
#   Class  label
#     PRAD  5
#     LUAD  4
#     BRCA  1
#     KIRC  3
#     COAD  2
```

When you apply `clusGap`, please use arguments `K.max=10, B=200,iter.max=100`, and when you use `kmeans`, please use arguments `iter.max = 100, nstart=25, algorithm = c("Hartigan-Wong")`.

When visualizing, I will set the genes in the first column and the second coloumn to x and y.

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




```
##             Class
## Sample       BRCA COAD KIRC LUAD PRAD
```

```
##   sample_117    0    0    1    0    0
##   sample_13     1    0    0    0    0
##   sample_178    1    0    0    0    0
##   sample_194    0    0    0    1    0
##   sample_210    0    0    0    1    0
##   sample_228    1    0    0    0    0
##   sample_243    0    0    0    0    1
##   sample_25     0    0    0    1    0
##   sample_298    0    0    0    1    0
##   sample_347    1    0    0    0    0
##   sample_354    0    1    0    0    0
##   sample_372    0    0    0    0    1
##   sample_373    0    0    0    1    0
##   sample_414    0    1    0    0    0
##   sample_425    0    0    0    0    1
##   sample_462    0    0    0    1    0
##   sample_518    0    0    0    1    0
##   sample_525    0    0    0    0    1
##   sample_554    0    0    1    0    0
##   sample_589    0    0    0    1    0
##   sample_592    0    0    1    0    0
##   sample_601    0    0    0    0    1
##   sample_602    0    0    1    0    0
##   sample_648    0    0    0    1    0
##   sample_664    1    0    0    0    0
##   sample_708    1    0    0    0    0
##   sample_765    0    0    1    0    0
##   sample_767    0    1    0    0    0
##   sample_782    1    0    0    0    0
##   sample_90     0    0    0    0    1
```

```
##
##        1
##   BRCA 7
##   COAD 3
##   KIRC 5
##   LUAD 9
##   PRAD 6
```

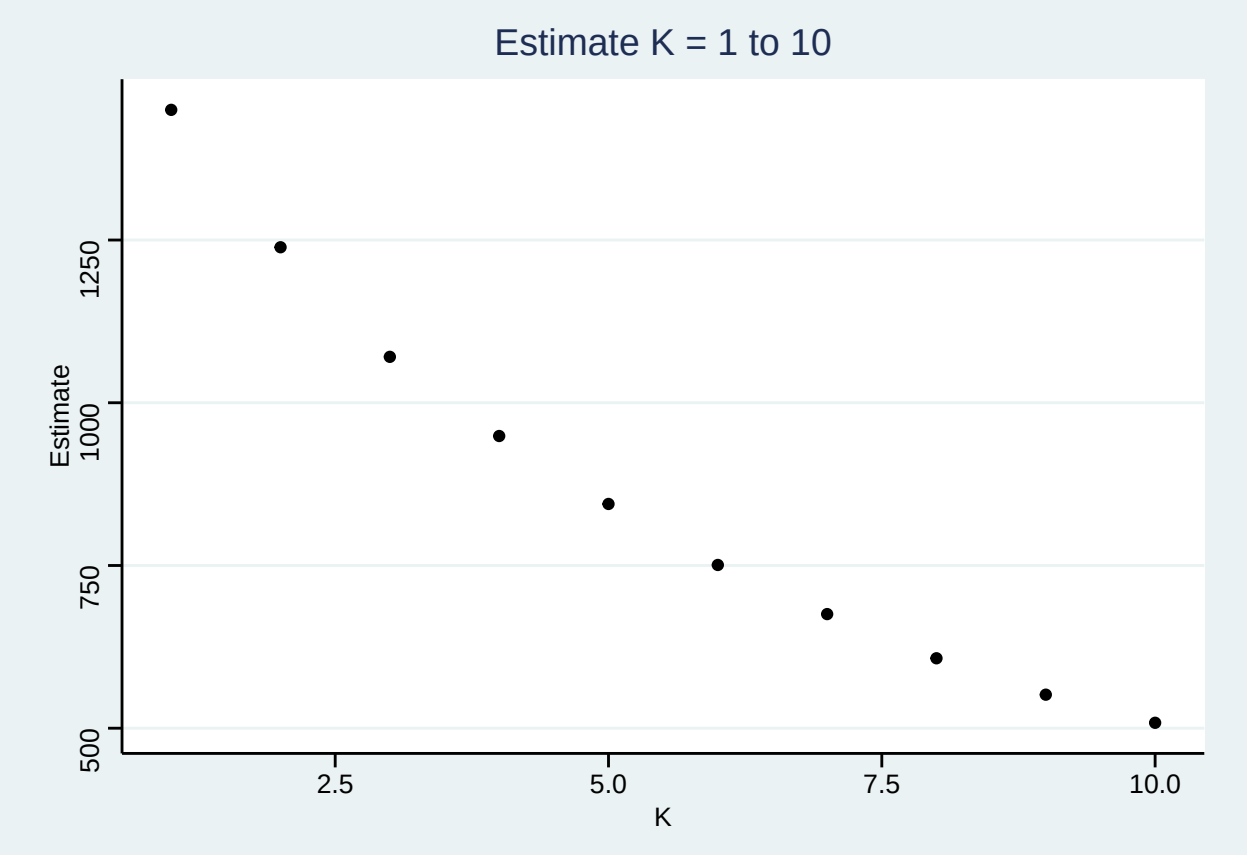
The number of clusters obtained through gap statistics is one, and as a result of visualizing the k means clustering of k=1, it can be confirmed that five classes belong to one cluster.
In addition, as a result of executing a table command on the labels1, it can be seen that each sample belongs to one class.
Also, as a result of checking the table for classification errors, it can be seen that five classes belong to one cluster.

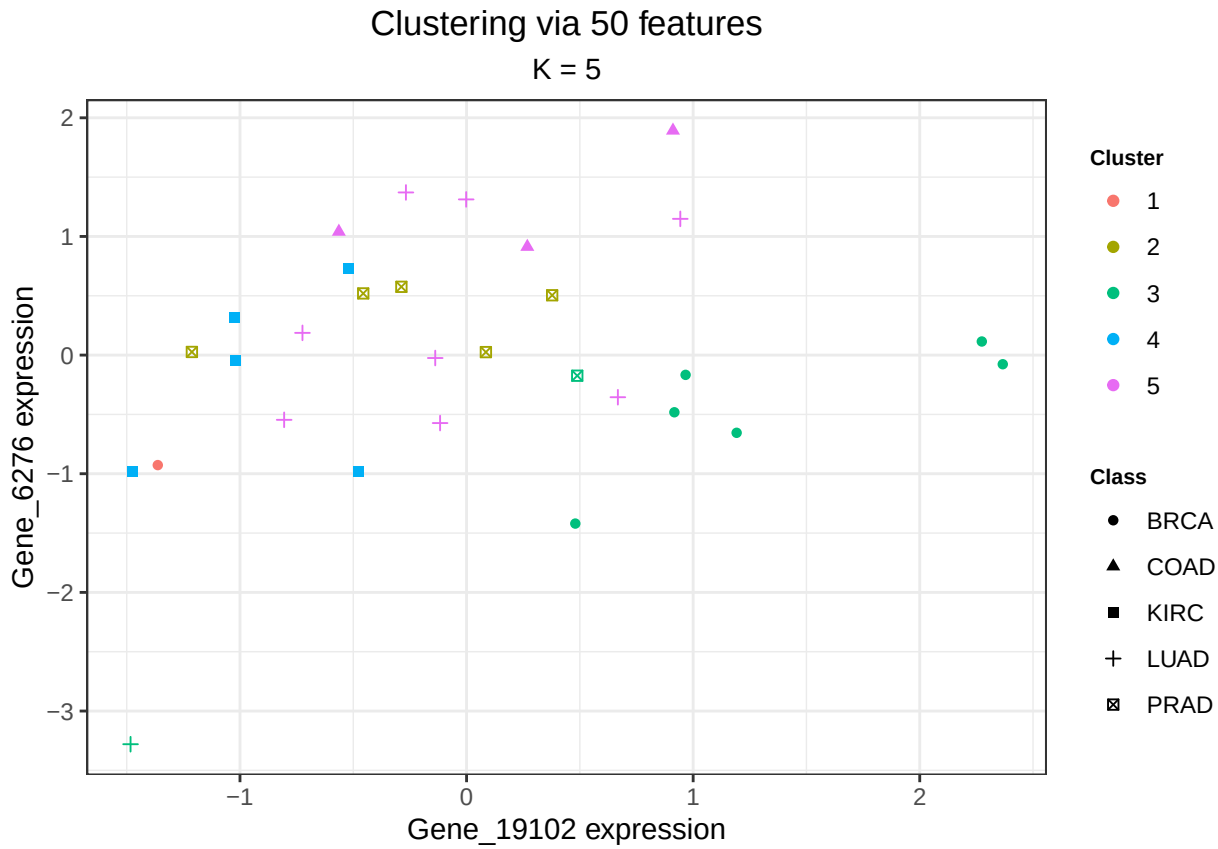
2.1.2.2 Part 2 Upon implementing `kmeans` with k as the number of clusters, we will obtain the "total within-cluster sum of squares" $W(k)$ from the output `tot.withinss` of `kmeans`. If we try a sequence of $k = 1, 2, 3, ..., 10$, then we get $W(k)$ for each k between 1 and 10. Let us look at the difference $\Delta_k = W(k) - W(k+1)$ for k ranging from 1 to 9. The K^* for which

$$\{\Delta_k : k < K^*\} \gg \{\Delta_k : k \geq K^*\}$$

is an estimate of the true number K of clusters in the data, where $\gg$ means "much larger". Apply this method to obtain an estimate of K for the data you created in **Part 1**, and provide a plot of $W(k)$ against k for each k between 1 and 10. Compare this estimate with the estimate obtained in **Part 1** given by the gap statistic, comment on the accuracy of the two estimates, and explain why they are different.

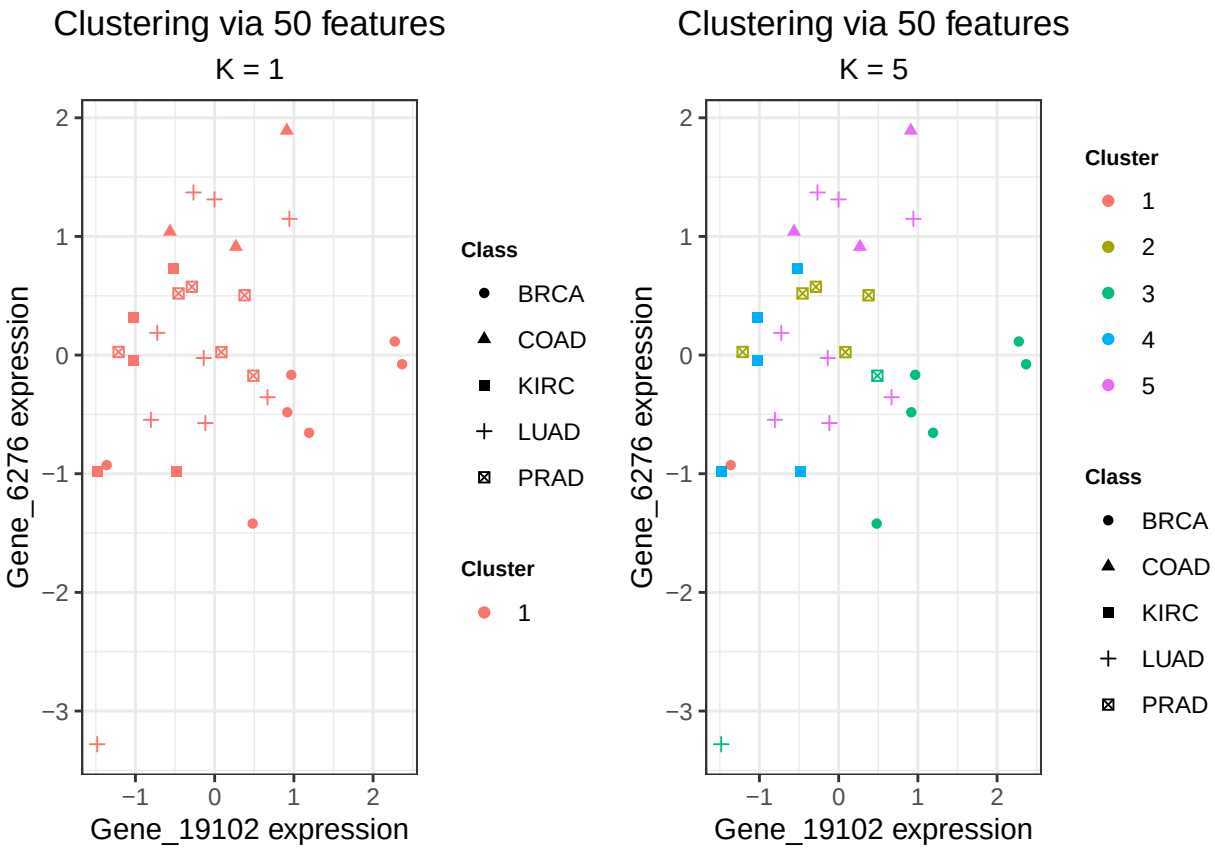
```
##      K  Estimate
## 1    1 1450.0000
## 2    2 1238.9016
## 3    3 1070.5300
## 4    4  948.9287
## 5    5  844.5063
## 6    6  750.7843
## 7    7  675.3198
## 8    8  607.4560
## 9    9  551.5170
## 10 10  508.3566
```






```
##
##         1 2 3 4 5
##    BRCA 1 0 6 0 0
##    COAD 0 0 0 0 3
##    KIRC 0 0 0 5 0
##    LUAD 0 0 1 0 8
##    PRAD 0 5 1 0 0
```

As a result of checking the number of clusters with a data frame and graph calculating the estimated distance from 1 to 10, it is judged that it is better to set the number of clusters to 5. As a result of setting the number of clusters to 5 and forming a k cluster analysis graph, it can be confirmed that the classes are divided into five clusters. However, if you check the classification error table, it is better to think of cluster 1 as an outlier because group 1 is grouped with only one variable.




```
## [1] 1450.0000  844.5063
```

```
## [1] -1.364242e-12  6.054937e+02
```

As a result of comparing the graph obtained in part 1 with the current graph, it can be seen that the current graph is better clustered.
As a result of comparing the two total withinss,
k = 1: **1450.00**
k = 5: **844.5063**
the model with k set to 5 is smaller, so it can be said to be a better model.
In addition, as a result of comparing betweenss,
k = 1: **-1.364242e-12**
k = 5: **6.054937e+02**
this is also a better model because the model with k set to 5 is larger.

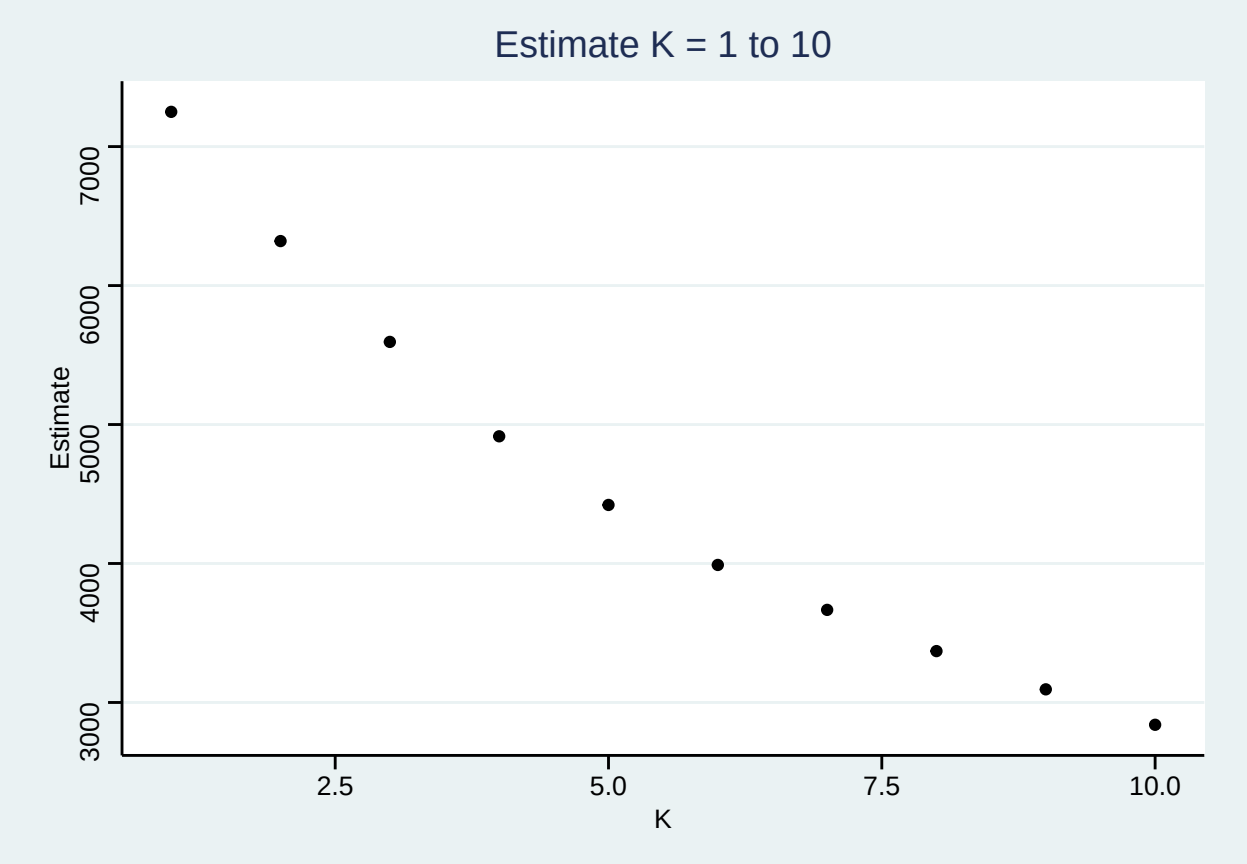
2.1.2.3 Part 3 Randomly pick 250 genes and their expressions from "stdgexpProj1", and for these expressions, do the analysis in **Part 1** and **Part 2**. Report your findings, compare your findings with those from **Part 1** and **Part 2**; if there are differences between these findings, explain why. Regard using more genes as using more features, does using more features necessarily give more accurate clutering or classification results?

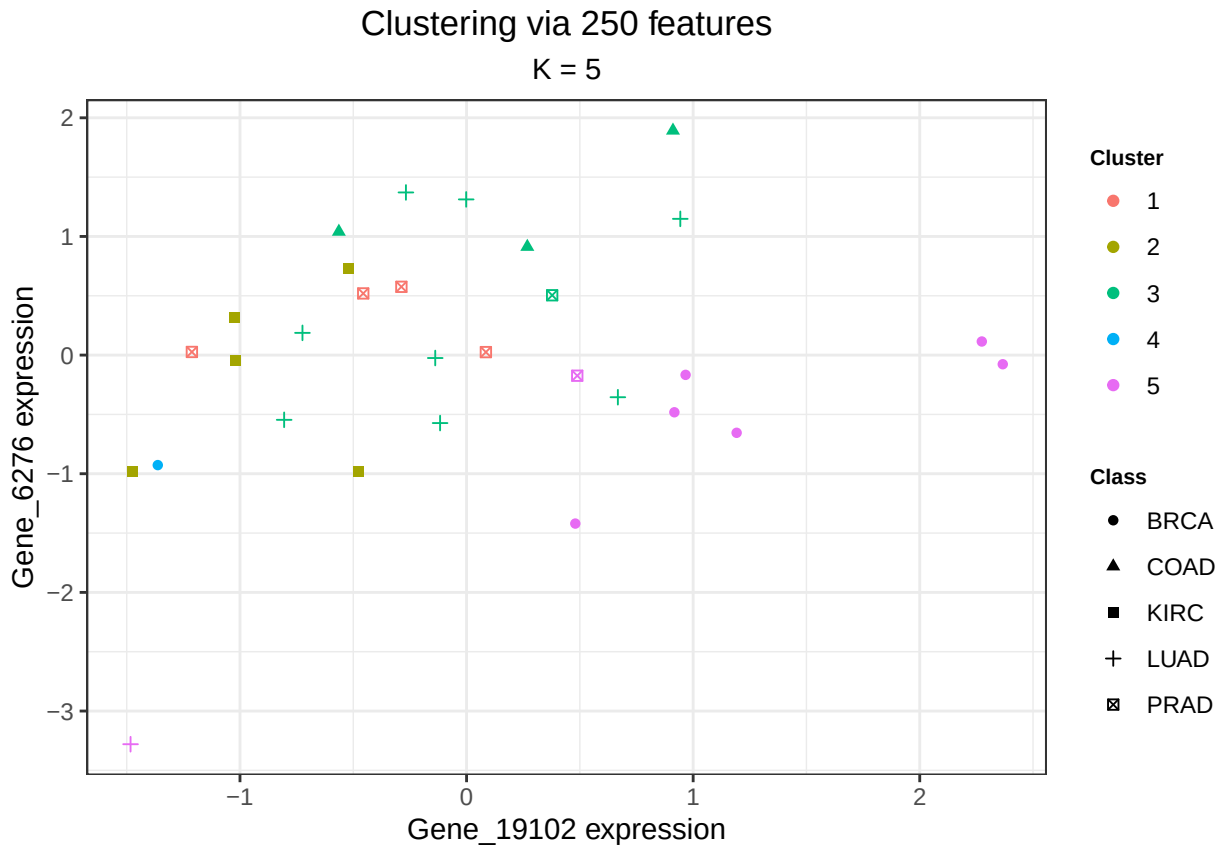
```
## [1] 1
```


```
##
##        1
##   BRCA 7
##   COAD 3
##   KIRC 5
##   LUAD 9
##   PRAD 6
```

As a result of randomly selecting 250 genes and their expressions and performing gap statistics, the result was 1. Through this, as a result of performing the k cluster analysis, it can be seen that five classes are grouped into one cluster.

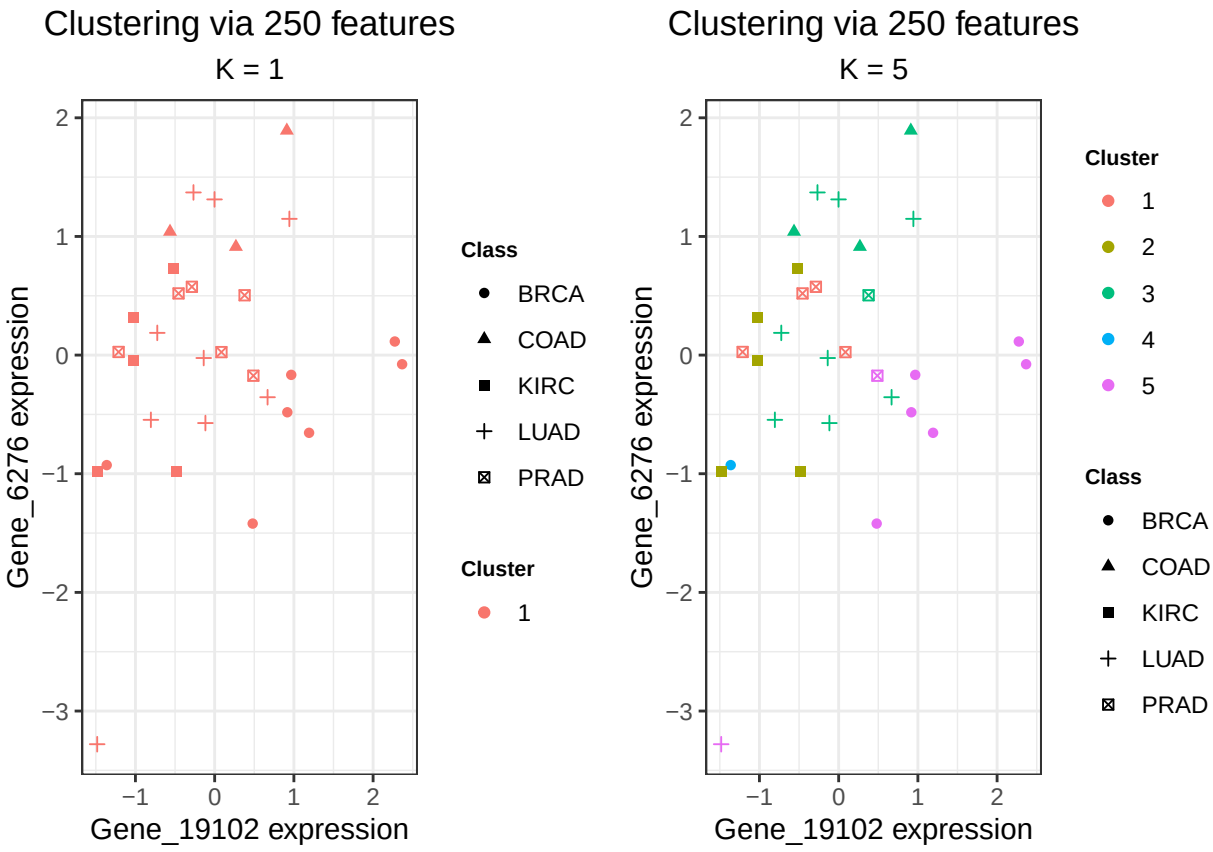
```
##     K Estimate
## 1   1 7250.000
## 2   2 6320.140
## 3   3 5594.501
## 4   4 4915.341
## 5   5 4421.287
## 6   6 3989.524
## 7   7 3666.299
## 8   8 3369.543
## 9   9 3094.439
## 10 10 2839.522
```

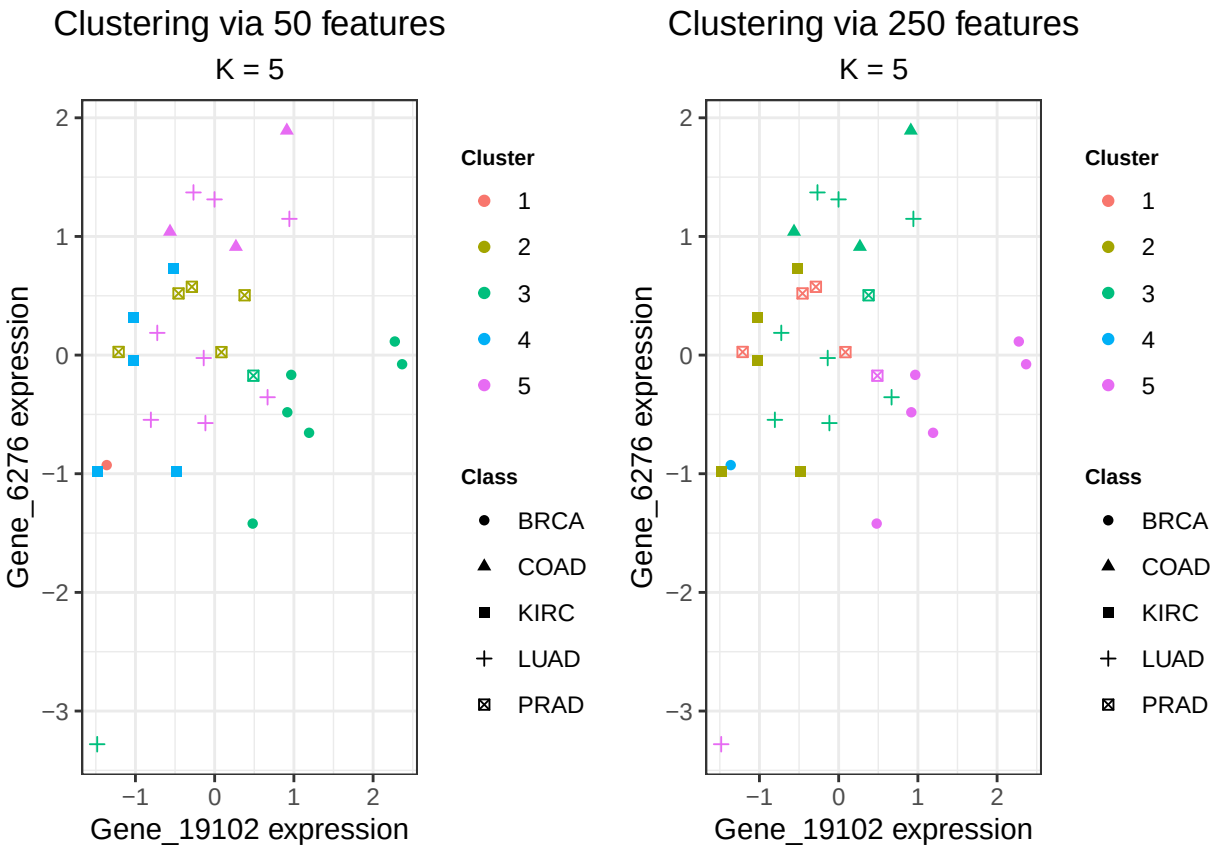

Estimate K = 1 to 10
Estimate
3000
4000
5000
6000
7000
2.5
5.0
7.5
10.0
K




```
##
##         1 2 3 4 5
##   BRCA 0 0 0 1 6
##   COAD 0 0 3 0 0
##   KIRC 0 5 0 0 0
##   LUAD 0 0 8 0 1
##   PRAD 4 0 1 0 1
```

As a result of checking the number of clusters with data frames and graphs, we calculated the estimated distance between 1 and 10 and determined that it was better to set the number of clusters to 5. As a result of setting the number of clusters to 5 and configuring the k cluster analysis graph, it can be seen that 5 classes are divided into 5 clusters. However, if you check the classification error table, it is better to think of cluster 4 as an outlier because group 4 is grouped with only one variable.


Clustering via 250 features
K = 1
Gene_6276 expression
Gene_19102 expression
Class
BRCA
COAD
KIRC
LUAD
PRAD
Cluster
1
Clustering via 250 features
K = 5
Gene_6276 expression
Gene_19102 expression
Cluster
1
2
3
4
5
Class
BRCA
COAD
KIRC
LUAD
PRAD



The first two graphs show k cluster analysis graphs using 250 genes, each k being formed by 1 and 5.
Looking at the two graphs, it can be seen that the k cluster analysis graph, which forms k as 5, classifies clusters better.

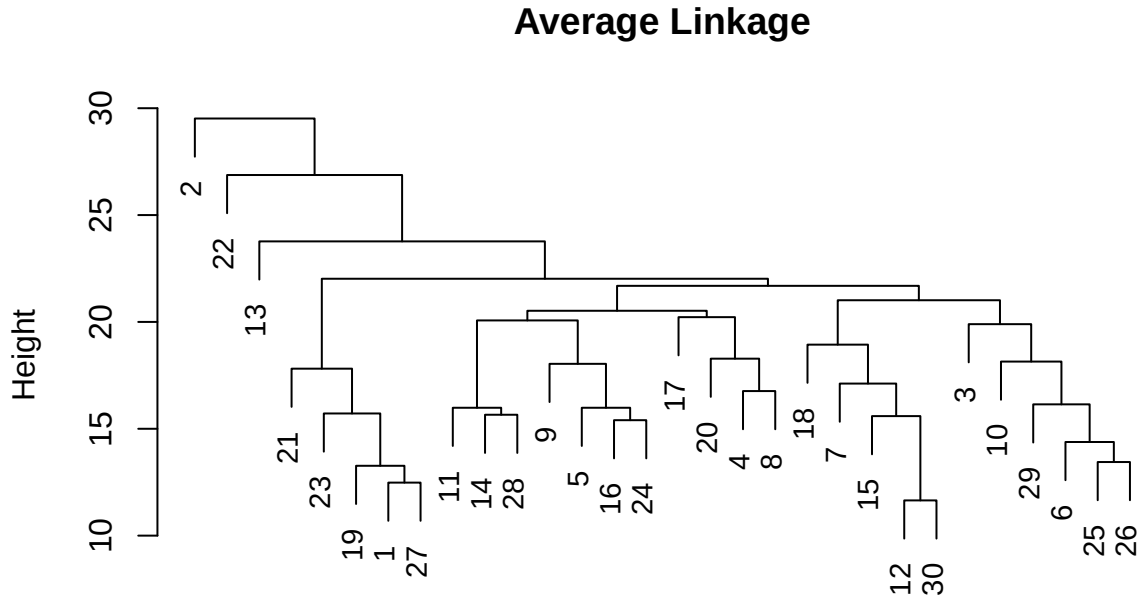
The second two graphs show a k cluster analysis graph in which k is formed into 5 using 50 genes and 250 genes.
When comparing the two graphs, **it can be seen that using more genes and more features does not mean that clustering is more accurate**.

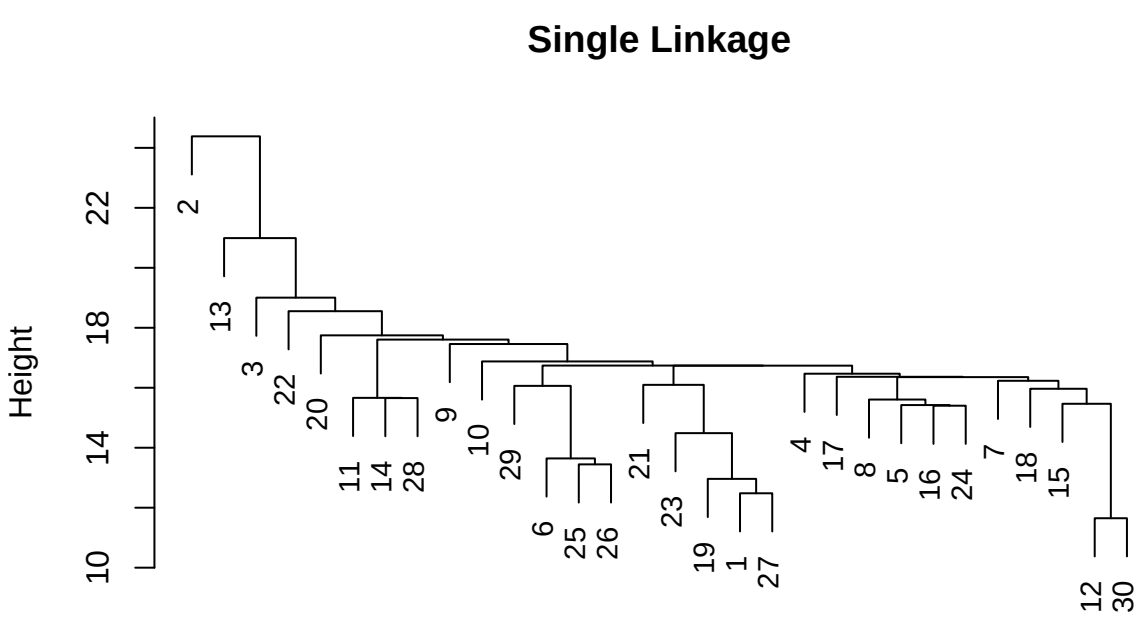
```
## $K
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $Estimate
##  [1] 1450.0000 1238.9016 1070.5300  948.9287  844.5063  750.7843  675.3198
##  [8]  607.4560  551.5170  508.3566
##
## $K
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $Estimate
##  [1] 7250.000 6320.140 5594.501 4915.341 4421.287 3989.524 3666.299 3369.543
##  [9] 3094.439 2839.522
```

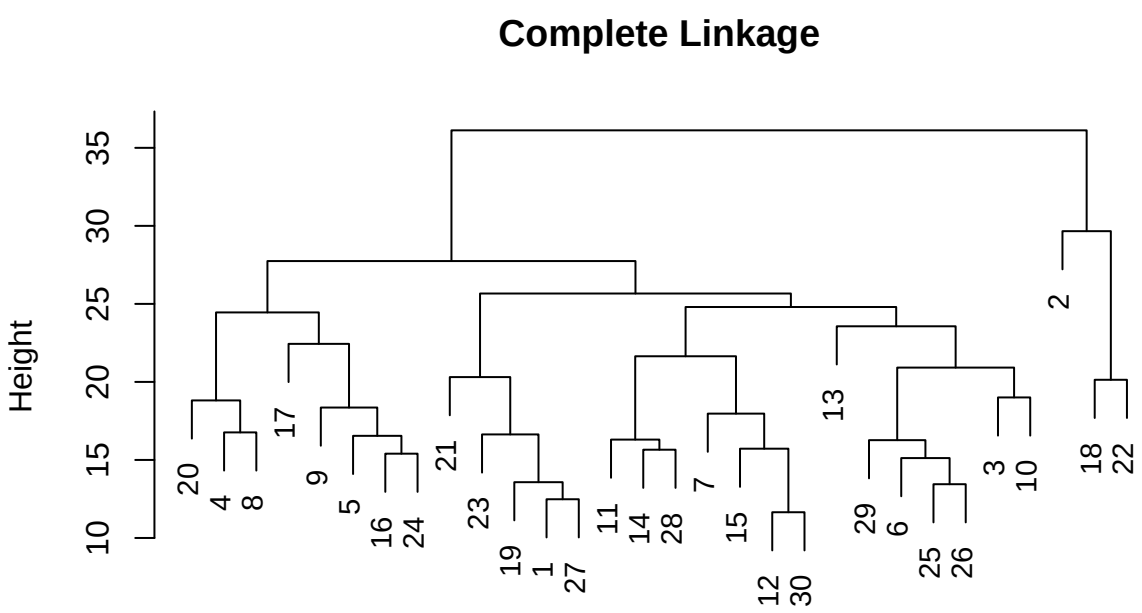
As a result of calculating the estimate distance of the number of clusters from 1 to 10 using 50 genes and 250 genes, the distances of the two models are different. Because the number of randomly designated genes is different at first, the estimated distance of the number of clusters is inevitably higher in models using 250 genes.

2.1.3 Task A3

Randomly pick 250 genes and their expressions from "stdgexpProj1", and for these expressions, do the following: respectively apply hierarchical clustering with average linkage, single linkage, and complete linkage to cluster subjects into groups, and create a dendrogram. For the dendrogram obtained from average linkage, find the height at which cutting the dendrogram gives the same number of groups in "labels1", and comment on the clustering results obtained at this height by comparing them to the truth contained in "labels1".




Single Linkage
Height
10
14
18
22
2
13
3
22
20
11
14
28
9
10
29
6
25
26
21
23
19
1
27
4
17
8
5
16
24
7
18
15
12
30

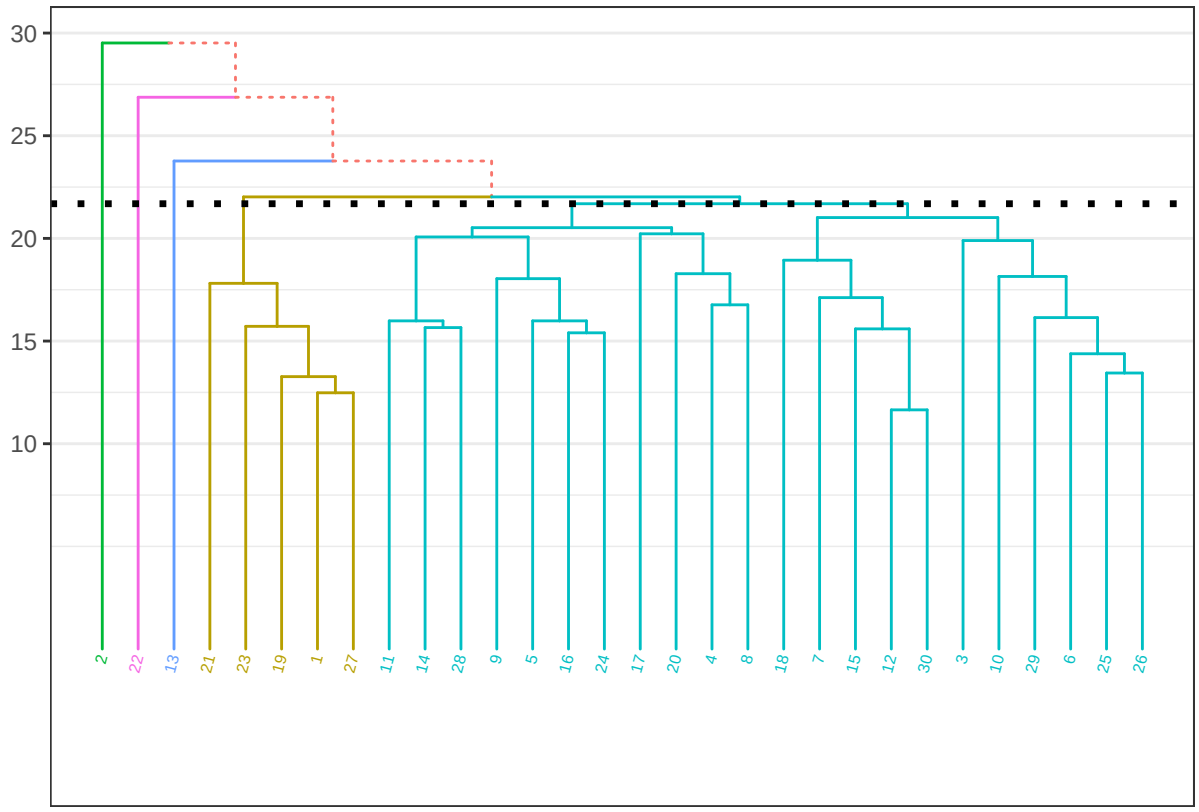


It can be seen that hierarchical clustering dendrograms with average linkage, single linkage, and complete connection were created, respectively.

```
## [1] TRUE
```

```
## [1] 21.68848
```

```
##
##      BRCA COAD KIRC LUAD PRAD
##   1    0    1    0    3    1
##   2    0    0    0    1    0
##   3    7    2    5    4    4
##   4    0    0    0    0    1
##   5    0    0    0    1    0
```



As a result of checking the height of the dendrogram obtained from the average linkage after cutting, it is **21.68848**.
Comparing the clustering results obtained at this height with the truths included in "labels1", it can be confirmed that there are five clusters well distributed.

2.2 Task B. Classification

For this task, we will use the same data set you would have downloaded. Please use `set.seed(123)` for random sampling via the command `sample` and any other process where artificial randomization is needed.

2.2.1 Task B1

After you obtain "labels.csv" and "data.csv", do the following:

- Filter out genes (from "data.csv") whose expressions are zero for at least 300 subjects, and save the filtered data as R object "gexp2".
- Use the command `sample` to randomly select 1000 genes and their expressions from "gexp2", and save the resulting data as R object "gexp3".
- Pick the samples from "labels.csv" that are for cancer type "LUAD" or "BRCA", and save them as object "labels2". For these samples, pick the corresponding gene expressions from "gexp3" and save them as object "stdgexp2"

2.2.2 Taks B2:

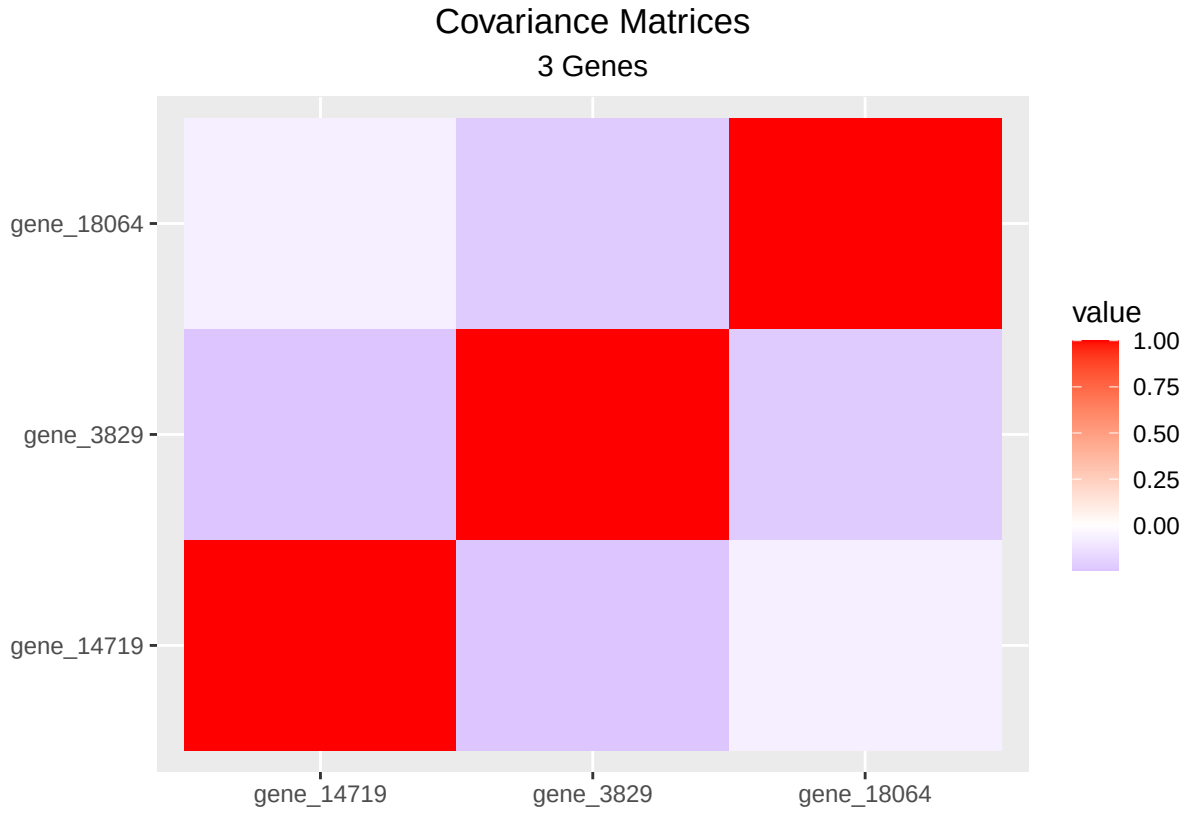
The assumptions of linear or quadratic discriminant analysis requires that each observation follows a Gaussian distribution given the class or group membership of the observation, and that each observation follows a Gaussian mixture model. In our settings here, each observation (as a row) within a group would follow a Gaussian with dimensionality equal to the number of genes (i.e., number of entries of the row). So, the more genes whose expressions we use for classification, the higher the dimension of these Gaussian distributions. Nonetheless, you need to check if the Gaussian mixture assumption is satisfied. Note that we only consider two classes "LUAD" and "BRCA", for which the corresponding Gaussian mixture has 2 components and hence has 2 bumps when its density is plotted.

Do the following and report your findings on classification:

- Randomly pick 3 genes and their expressions from "stdgexp2", and save them as object "stdgexp2a".
- Randomly pick 60% of samples from "stdgexp2a", use them as the training set, and use the rest as the test set. You can round down the number of samples in the training set by the command `floor` if it is not an integer.
- Build a quadratic discriminant analysis model using the training set, and apply the obtained model to the test set to classify each of its observations. You should code "BRCA" as 0 and "LUAD" as 1. If for an observation the posterior probability of being "BRCA" is predicted by the model to be greater than 0.5, the observation is classified as "BRCA". Report via a 2-by-2 table on the classification errors. Note that the predicted posterior probability given by `qda` is for an observation to belong to class "BRCA".

- Before building a quadratic discriminant analysis model, you need to check for highly correlated gene expressions, i.e., you need to check the sample correlations between each pair of columns of the training set. If there are highly correlated gene expressions, the estimated covariance matrix can be close to to being singular, leading to unstable inference. You can remove a column from two columns when their contained expressions have sample correlation greater than 0.9 in absolute value.

After setting "BRCA" to 0 and "LUAD" to 1, a new column called Code was created in the data.




```
## integer(0)
```

Before constructing the quadratic discriminant analysis model, a sample correlation was checked between each pair of columns of the training set to confirm high correlation gene expression, and it was confirmed that there were no gene expressions with a higher correlation than 0.9.

```
##                       TrueClassLabel
## QDAEstimtedClassLabel   0   1
##                     0 112  27
##                     1   5  33
```

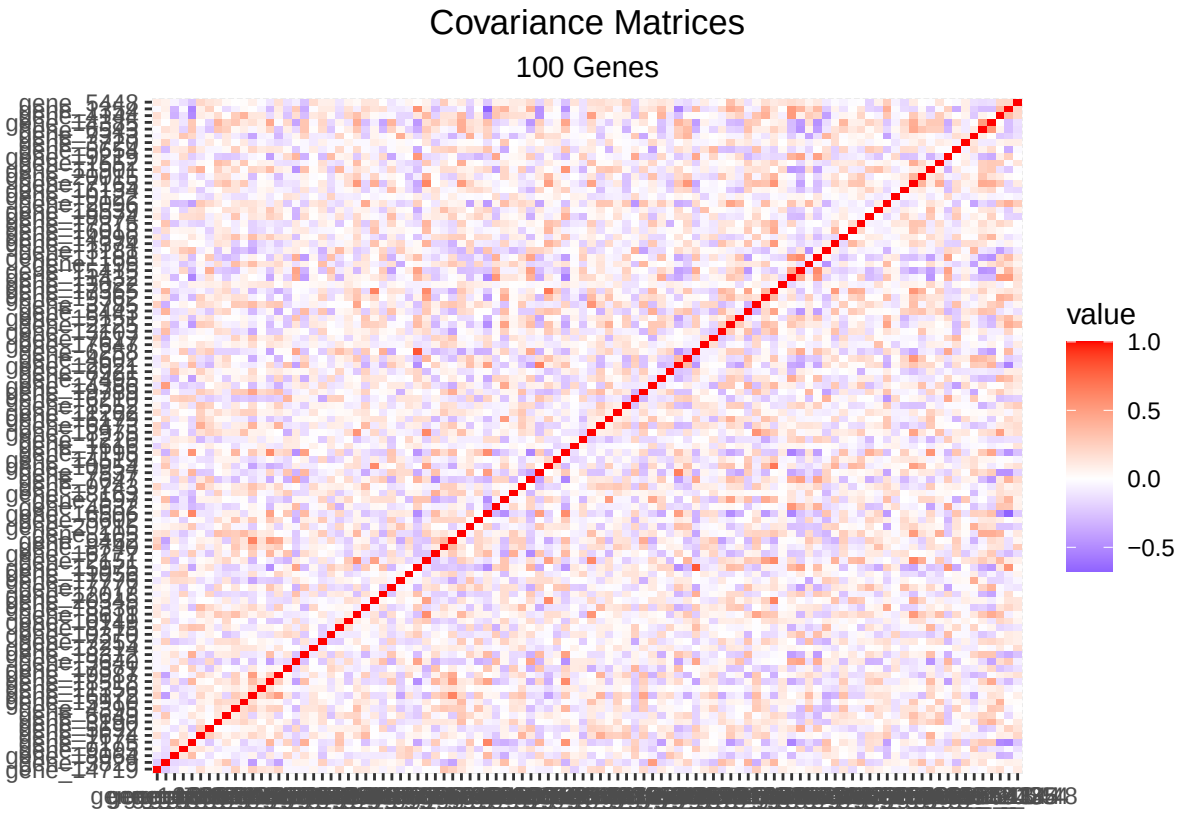
```
## [1] 0.819209
```

```
## [1] 0.180791
```

As a result of checking the 2x2 table for classification errors and accuracy, it can be seen that the accuracy is **81.92 %** and the classification error is **18.08 %**.

2.2.3 Taks B3

- Randomly pick 100 genes and their expressions from "stdgexp2", and save them as object "stdgexp2b".
- Randomly pick 75% of samples from "stdgexp2b", use them as the training set, and use the rest as the test set. You can round down the number of samples in the training set by the command `floor` if it is not an integer.
- Apply quadratic discriminant analysis by following the requirements given in **Taks B2**. Compare classification results you find here with those found in **Taks B2**, and explain on any difference you find between the classification results.




```
## integer(0)
```

Before constructing the quadratic discriminant analysis model, a sample correlation was checked between each pair of columns of the training set to confirm high correlation gene expression, and it was confirmed that there were no gene expressions with a higher correlation than 0.9.

```
##                         TrueClassLabel1
## QDAEstimtedClassLabel1  0  1
##                       0 71 40
##                       1  0  0
```

```
## [1] 0.6396396
```

```
## [1] 0.3603604
```

As a result of checking the 2x2 table for classification errors and accuracy, it can be seen that the accuracy is **63.96 %** and the classification error is **36.04 %**.

As a result of checking the QDA analysis model obtained from Task B2 and the QDA analysis model obtained from Task B3, it can be seen that the **Task B2 model is a better model** because Task B2 more higher accuracy and lower error rate than Task B3.

2.2.4 Taks B4

- Randomly pick 100 genes and their expressions from "stdgexp2", and save them as object "stdgexp2b".
- Randomly pick 75% of samples from "stdgexp2b", use them as the training set, and use the rest as the test set. You can round down the number of samples in the training set by the command `floor` if it is not an integer.
- Then apply k-nearest-neighbor (k-NN) method with neighborhood size k=3 to the test data to classify each observation in the test set into one of the cancer types. Here, for an observation, if the average of being cancer type "BRCA" is predicted by k-NN to be greater than 0.5, then the observation is classified as being "BRCA". Report via a 2-by-2 table on the classification errors. Compare and comment on the classification results obtained here to those obtain in **Task B3**. If there is any difference between the classification results, explain why.

```
##                       KNNTrueClassLabel
## KNNEstimtedClassLabel  0  1
##                     0  0 38
##                     1 71  2
```

```
## [1] 0.01801802
```

```
## [1] 0.981982
```

```
##                       TrueClassLabel
## QDAEstimtedClassLabel   0   1
##                     0 112  27
##                     1   5  33
```

```
## [1] 0.819209
```

```
## [1] 0.180791
```

```
##                         TrueClassLabel1
## QDAEstimtedClassLabel1  0  1
##                        0 71 40
##                        1  0  0
```

```
## [1] 0.6396396
```

```
## [1] 0.3603604
```

As a result of checking the 2x2 table for classification errors and accuracy, it can be seen that the accuracy is **1.80 %** and the classification error is **98.20 %**.

Also as a result of compare the 2x2 table of the QDA analysis model obtained from Task B2, Task B3 and the KNN size k = 3 model obtained from Task B4, it can be seen that the **Task B2 QDA analysis model is a better model**. Because, Task B2 QDA model more higher accuracy and lower error rate than both TaskB3 and TaskB4. I think that KNN is a completely nonparametric approach and is not suitable for this dataset.

3 Discussion

When analyzing cancer with genetic data, I think it would be good to add linear discriminant analysis to analyze the results through accuracy and AUC model. In addition, for observations when performing k-NN, I think it would be better to exclude the classification of observations as "BRCA" if the mean of cancer type "BRCA" was predicted to be greater than 0.5 by k-NN.

4 Appendix

4.1 Task A

4.1.1 Task A1

```
# import csv file
dt <- read.csv("data.csv", stringsAsFactors = F, header = T)
# filter out expressions are zero for at least 300 subjects
gexp2 <- dt[, (colSums(dt == 0, na.rm = TRUE) <= 300), drop=TRUE]
# select randomly 1000 genes and expressions from gexp2
set.seed(123)
cSel <- sample(1:dim(gexp2)[2], 1000, replace=FALSE)
gexp3 = gexp2[,cSel]
# set X is column of all sample list and change the Sample column location.
gexp3$Sample <- gexp2$X
gexp3 <- gexp3[, c(1001, 1:1000)]
# import the labels data and change the X column to Sample
set.seed(123)
lb <- read.csv("labels.csv", stringsAsFactors = F, header=T)
names(lb)[names(lb)=="X"] <- "Sample"
# select randomly 30 labels from labels data
lb1 <- sample(nrow(lb), size=30)
labels1 <- lb[lb1,]
# corresponding samples from gexp3
gexpProj1 <- merge(gexp3, labels1, by="Sample")
gexpProj1 <- gexpProj1[, c(1,1002, 2:1001)]
# standard the gene expressions for each gene.
# set c(2:1001): not included sample and class
stdgexpProj1 <- scale(gexpProj1[,c(3:1002)], center = T, scale = T)
stdgexpProj1 <- data.frame(stdgexpProj1)
# change class type factor
stdgexpProj1$Class = factor(gexpProj1$Class)
```

4.1.2 Task A2

```
set.seed(123)
# set randomly 50 genes
g <- 50
sp <- sample(1:dim(stdgexpProj1)[2], size = g, replace = FALSE)
stdgexpProj1_a = stdgexpProj1[,sp]
stdgexpProj1_b = stdgexpProj1[,sp]
# set clusGap
gap <- clusGap(stdgexpProj1_a, kmeans, K.max=10, B=200, iter.max=100)
k <- maxSE(gap$Tab[ ,"gap"], gap$Tab[, "SE.sim"],
```

```
           method="Tibs2001SEmax")
# estimated number of clusters
k
# kmeans data object as a matrix. so change data frame to matrix
stdgexpProj1_mt = as.matrix(stdgexpProj1_a)
# cluster using estimated number of clusters given by the gap stat.
km.out = kmeans(stdgexpProj1_mt, center = k,
                iter.max =  100,
                nstart = 25,
                algorithm = c("Hartigan-Wong"))
# create class and cluster in data frame (K = 1)
stdgexpProj1_a$Class = stdgexpProj1$Class
stdgexpProj1_a$Cluster = factor(km.out$cluster)
# show the plot (k = 1) given by the gap stat.
p1 <- ggplot(stdgexpProj1_a, aes(x= stdgexpProj1_a[,1],
                                 y=stdgexpProj1_a[,2])) +
   xlab("Gene_19102 expression") +
   ylab("Gene_6276 expression") +
   theme_bw() +
   geom_point(aes(shape=Class, color = Cluster), na.rm =T) +
   theme(legend.position = "right") +
   ggtitle("Clustering via 50 features", subtitle = "K = 1") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5),
         legend.title = element_text(size = 8, face = "bold"))
p1
# show labels1 table
table(labels1)
# show classfication error
table(stdgexpProj1_a$Class, stdgexpProj1_a$Cluster)
set.seed(123)
# set kmeans 1 to 10 function
k1to10 <- function(dat,kk) {
   x <- kmeans(dat,center = kk, iter.max = 100,
               nstart=25,
               algorithm = c("Hartigan-Wong"))
}
# set k between 1 to 10 data table
estTable <- data.frame(matrix(ncol=2, nrow=10))
colnames(estTable) = c("K","Estimate")
estTable$K = c(1:10)
# calculate k for each k between 1 and 10
for (i in 1:10) {
  est = k1to10(stdgexpProj1_mt, i)
  estTable[i, 2] = est$tot.withinss
}
# show k between 1 to 10 data table
```

```
estTable
# show estimates plot
estPlot <- ggplot(estTable, aes(x=K,y=Estimate)) +
   geom_point() +
   theme_stata() +
   ggtitle("Estimate K = 1 to 10")
estPlot
# set k = 5 clustering
set.seed(123)
km.out1 = kmeans(stdgexpProj1_mt, center = 5,
                 iter.max =  100,
                 nstart = 25,
                 algorithm = c("Hartigan-Wong"))
# create class and cluster in data frame (K=5)
stdgexpProj1_b$Class = stdgexpProj1$Class
stdgexpProj1_b$Cluster = factor(km.out1$cluster)
# show the plot (k = 5) given by the gap stat.
p2 <- ggplot(stdgexpProj1_b, aes(x= stdgexpProj1_b[,1],
                                 y=stdgexpProj1_b[,2])) +
  xlab("Gene_19102 expression") +
   ylab("Gene_6276 expression") +
   theme_bw() +
   geom_point(aes(shape=Class, color = Cluster), na.rm =T) +
   theme(legend.position = "right",
         legend.title = element_text(size = 8, face = "bold")) +
   ggtitle("Clustering via 50 features", subtitle = "K = 5") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5))
p2
# show classfication error
table(stdgexpProj1_b$Class, stdgexpProj1_b$Cluster)
# show two plot in same page
grid.arrange(p1,p2, ncol=2)
# compare part 1 and current total withinss
c(km.out$tot.withinss, km.out1$tot.withinss)
# compare part 1 and current betweenss
c(km.out$betweenss, km.out1$betweenss)
set.seed(123)
# set randomly 250 genes
g <- 250
sp <- sample(1:dim(stdgexpProj1)[2], size = g, replace = FALSE)
stdgexpProj1_c = stdgexpProj1[,sp]
stdgexpProj1_d = stdgexpProj1[,sp]

# set clusGap
gap <- clusGap(stdgexpProj1_c, kmeans, K.max=10, B=200, iter.max=100)
k <- maxSE(gap$Tab[ ,"gap"], gap$Tab[, "SE.sim"],
```

```
          method="Tibs2001SEmax")
# estimated number of clusters
k
# kmeans data object as a matrix. so change data frame to matrix
stdgexpProj1_mtt = as.matrix(stdgexpProj1_c)
# cluster using estimated number of clusters given by the gap stat.
km.out2 = kmeans(stdgexpProj1_mtt, center = k,
                 iter.max =  100, nstart = 25,
                 algorithm = c("Hartigan-Wong"))
# create class and cluster in data frame (K = 1)
stdgexpProj1_c$Class = stdgexpProj1$Class
stdgexpProj1_c$Cluster = factor(km.out2$cluster)
# show the plot (k = 1) given by the gap stat.
p3 <- ggplot(stdgexpProj1_c, aes(x= stdgexpProj1_c[,1],
                                 y=stdgexpProj1_c[,2])) +
   xlab("Gene_19102 expression") +
   ylab("Gene_6276 expression") +
   theme_bw() +
   geom_point(aes(shape=Class, color = Cluster), na.rm =T) +
   theme(legend.position = "right",
         legend.title = element_text(size = 8, face = "bold")) +
   ggtitle("Clustering via 250 features", subtitle = "K = 1") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5))
p3
# show classfication error
table(stdgexpProj1_c$Class, stdgexpProj1_c$Cluster)
# set k between 1 to 10 data table
estTable1 <- data.frame(matrix(ncol=2, nrow=10))
colnames(estTable1) = c("K","Estimate")
estTable1$K = c(1:10)
# calculate k for each k between 1 and 10
for (i in 1:10) {
  est = k1to10(stdgexpProj1_mtt, i)
  estTable1[i, 2] = est$tot.withinss
}
estTable1
# show estimate plot
estPlot1 <- ggplot(estTable1, aes(x=K,y=Estimate)) +
   geom_point() +
   theme_stata() +
   ggtitle("Estimate K = 1 to 10")
estPlot1
# set k = 5 clustering
set.seed(123)
km.out3 = kmeans(stdgexpProj1_mtt, center = 5,
                 iter.max =  100, nstart = 25,
```

```
                  algorithm = c("Hartigan-Wong"))
# create class and cluster in data frame (K=5)
stdgexpProj1_d$Class = stdgexpProj1$Class
stdgexpProj1_d$Cluster = factor(km.out3$cluster)
# show the plot (k = 5) given by the gap stat.
p4 <- ggplot(stdgexpProj1_d, aes(x= stdgexpProj1_d[,1],
                                 y=stdgexpProj1_d[,2])) +
   xlab("Gene_19102 expression") +
   ylab("Gene_6276 expression") +
   theme_bw() +
   theme(legend.title = element_text(size = 8, face = "bold")) +
   geom_point(aes(shape=Class, color = Cluster), na.rm =T) +
   theme(legend.position = "right") +
   ggtitle("Clustering via 250 features", subtitle = "K = 5") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5))
p4
# show classfication error
table(stdgexpProj1_d$Class, stdgexpProj1_d$Cluster)
# show compare two plots (Part2 and Part3)
grid.arrange(p3,p4, ncol=2)
# show compare two plots (Part1 and Part3)
grid.arrange(p2, p4, ncol=2)
# compare part 2 and part3 estimate table
c(estTable, estTable1)
```

4.1.3 Task A3

```
set.seed(123)
# set randomly 250 genes
gene <- 250
sp <- sample(1:dim(stdgexpProj1)[2], size = gene, replace = FALSE)
stdgexpProj1_hc = stdgexpProj1[,sp]
# set average linkage
hcAvg = hclust(dist(stdgexpProj1_hc), method="average")
# show average linkage plot
plot(hcAvg, main="Average Linkage", xlab="", sub="", cex=0.9)
# set single linkage
hcSingle = hclust(dist(stdgexpProj1_hc), method="single")
# show single linkage plot
plot(hcSingle, main="Single Linkage", xlab="", sub="", cex=0.9)
# set complete linkage
hcComplete = hclust(dist(stdgexpProj1_hc), method="complete")
# show complete linkage plot
plot(hcComplete, main="Complete Linkage", xlab="", sub="", cex=0.9)
```

```r
# import source code (using dendrogram)
source("Plotggdendro.r", encoding = "UTF-8")
# cut the obtain 5 clusters
ct5 <- cutree(hcAvg, k=5)
nh = length(hcAvg$height)
cutheightAVG <- hcAvg$height[nh - 4]
# check boolean cut tree to obtain 5 clusters
all.equal(cutree(hcAvg, h= cutheightAVG), ct5)
# show cut height (numeric)
cutheightAVG
# show table
table(cutree(hcAvg, h=cutheightAVG) ,labels1$Class)
# show average dendrogram (cut height)
dendro_AVG <- plot_ggdendro(dendro_data_k(hcAvg, 5), direction = "tb",
                            heightReferece = cutheightAVG,
                            expand.y = 0.2,
                            label.size = 2,
                            branch.size = 0.5)
dendro_AVG
```

4.2 Task B

4.2.1 Task B1

```r
# filter out expressions are zero for at least 300 subjects
gexp2 <- dt[, (colSums(dt == 0, na.rm = TRUE) <= 300), drop=TRUE]
# select randomly 1000 genes and expressions from gexp2
set.seed(123)
cSel <- sample(1:dim(gexp2)[2], 1000, replace=FALSE)
# set X is column of all sample list and rotate location
gexp3 = gexp2[,cSel]
gexp3$Sample <- gexp2$X
gexp3 <- gexp3[, c(1001, 1:1000)]
# filter cancer type LUAD or BRCA in labels
labels2 <- lb %>%
   filter(Class %in% c("LUAD", "BRCA"))
# corresponding samples from gexp3 and rotate the location
stdgexp2 <- merge(gexp3, labels2, by="Sample")
stdgexp2 <- stdgexp2[, c(1,1002, 2:1001)]
```

4.2.2 Task B2

```r
set.seed(123)
# pick randomly 3 genes expression from stdgexp2
```

```r
n = 3
sp1 <- sample(1:dim(stdgexp2)[2], size = n, replace = F)
stdgexp2a <- stdgexp2[,sp1]
set.seed(123)
# pick randomly 60% of samples in training set
train2a <- base::sample(1:nrow(stdgexp2a), floor(0.6*nrow(stdgexp2a)))
trainSet2a <- stdgexp2a[train2a,]
# reamin 40% of samples in test set
test2a <- (1:nrow(stdgexp2a))[-train2a]
testSet2a <- stdgexp2a[test2a,]
# set code BRCA as 0 and LUAD as 1
stdgexp2$Code[stdgexp2$Class == "BRCA"] <- 0
stdgexp2$Code[stdgexp2$Class == "LUAD"] <- 1
# set code type to numeric
stdgexp2$Code = as.numeric(stdgexp2$Code)
# set train and test labels
trainLabels2a <- stdgexp2$Code[train2a]
testLabels2a <- stdgexp2$Code[test2a]
# extra estimated covariance matrices and plotting it
melted <- melt(cor(trainSet2a))
ggplot(data=melted, aes(x=Var1, y=Var2, fill=value)) +
   geom_tile() +
   scale_fill_gradient2(low="blue", high = "red") +
   xlab("") +
   ylab("") +
   ggtitle("Covariance Matrices", subtitle = "3 Genes") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5))
# find expressions have sample correlation greater than 0.9 in absolute value
which(abs(melted$value) > 0.9 & abs(melted$value) != 1, arr.ind=TRUE)
# train and test sets to data frame
ta1 <- data.frame(trainSet2a)
te1 <- data.frame(testSet2a)
# fit qda
qda.fit <- qda(trainLabels2a~., data=ta1)
# qdda predicting
qda.pred <- predict(qda.fit, te1)
# set true label
TrueClassLabel = testLabels2a
# set estimed class label if for an observation the
# posterior of being BRCA is predicted by the model to be greater than 0.5,
# the observation is classified as BRCA (0).
QDAEstimtedClassLabel = qda.pred$class
yesIdx = which(qda.pred$posterior[,1] > 0.5)
QDAEstimtedClassLabel[yesIdx] = 0
# show table
table(QDAEstimtedClassLabel, TrueClassLabel)
```

```
# accuarcy of QDA
mean(QDAEstimtedClassLabel == TrueClassLabel)
# classfication error
mean(QDAEstimtedClassLabel != TrueClassLabel)
```

4.2.3 Task B3

```
set.seed(123)
# set randomly 100 genes expressions from stdgexp2
n = 100
sp2 <- sample(1:dim(stdgexp2)[2], size = n, replace = F)
stdgexp2b <- stdgexp2[,sp2]
set.seed(123)
# set randomly 75 % of samples as the train set
train3a <- base::sample(1:nrow(stdgexp2b), floor(0.75*nrow(stdgexp2b)))
trainSet3a <- stdgexp2b[train3a,]
# set remaining of samples as the test set
test3a <- (1:nrow(stdgexp2b))[-train3a]
testSet3a <- stdgexp2b[test3a,]
# set train and test label
trainLabels3a <- stdgexp2$Code[train3a]
testLabels3a <- stdgexp2$Code[test3a]
# extra estimated covariance matrices and plotting it
melted1 <- melt(cor(trainSet3a))
ggplot(data=melted1, aes(x=Var1, y=Var2, fill=value)) +
   geom_tile() +
   scale_fill_gradient2(low="blue", high = "red") +
   xlab("") +
   ylab("") +
   ggtitle("Covariance Matrices", subtitle = "100 Genes") +
   theme(plot.title = element_text(hjust = 0.5),
         plot.subtitle = element_text(hjust = 0.5))
# find expressions have sample correlation greater than 0.9 in absolute value
which(abs(melted1$value) > 0.9 & abs(melted1$value) != 1, arr.ind=TRUE)
# train and test sets to data frame
ta2 <- data.frame(trainSet3a)
te2 <- data.frame(testSet3a)
# set qda
qda.fit1 <- qda(trainLabels3a~., data=ta2)
# qda predicting
qda.pred1 <- predict(qda.fit1, te2)
# set true class
TrueClassLabel1 = testLabels3a
# set estimed class label if for an observation the posterior
# of being BRCA is predicted by the model to be greater than 0.5,
# the observation is classified as BRCA (0).
```

```
QDAEstimtedClassLabel1 = qda.pred1$class
yesIdx1 = which(qda.pred1$posterior[,1] > 0.5)
QDAEstimtedClassLabel1[yesIdx1] = 0
# show table
table(QDAEstimtedClassLabel1, TrueClassLabel1)
# accuarcy of QDA
mean(QDAEstimtedClassLabel1 == TrueClassLabel1)
# classfication error
mean(QDAEstimtedClassLabel1 != TrueClassLabel1)
```

4.2.4 Task B4

```
set.seed(123)
# set randomly 100 genes expressions from stdgexp2
n = 100
sp2 <- sample(1:dim(stdgexp2)[2], size = n, replace = F)
stdgexp2b <- stdgexp2[,sp2]
set.seed(123)
# set randomly 75 % sample in training set and set remaining sample in test set
train4a <- base::sample(1:nrow(stdgexp2b), floor(0.75*nrow(stdgexp2b)))
test4a <- (1:nrow(stdgexp2b))[-train4a]
trainSet4a <- stdgexp2b[train4a,]
testSet4a <- stdgexp2b[test4a,]
# set train and test labels
trainLabels4a <- stdgexp2$Code[train4a]
testLabels4a <- stdgexp2$Code[test4a]
# train and test sets to data frame
ta4 <- data.frame(trainSet4a)
te4 <- data.frame(testSet4a)
# set k-nearest-neighbor method k = 3.
knnT4 = knn(ta4, te4, cl = trainLabels4a, k = 3)
knnT4 <- as.matrix(knnT4)
# if the average of being cancer type "BRCA" is predicted by
# k-NN to be greater than 0.5,
# then the observation is classified as being "BRCA".
KNNEstimtedClassLabel = rep(1, nrow(knnT4))
BRCAIdx = which(knnT4 > 0.5)
KNNEstimtedClassLabel[BRCAIdx] = 0
KNNTrueClassLabel = testLabels4a
# show table
table(KNNEstimtedClassLabel, KNNTrueClassLabel)
# accuarcy of KNN
mean(KNNEstimtedClassLabel == KNNTrueClassLabel)
# classfication error
mean(KNNEstimtedClassLabel != KNNTrueClassLabel)
```

```
# recall task B2 and task B3 result (using compare)
# show table QDA Task B2
table(QDAEstimtedClassLabel, TrueClassLabel)
# accuarcy of QDA Task B2
mean(QDAEstimtedClassLabel == TrueClassLabel)
# classfication error Task B2
mean(QDAEstimtedClassLabel != TrueClassLabel)
# show table QDA Task B3
table(QDAEstimtedClassLabel1, TrueClassLabel1)
# accuarcy of QDA Task B3
mean(QDAEstimtedClassLabel1 == TrueClassLabel1)
# classfication error Task B3
mean(QDAEstimtedClassLabel1 != TrueClassLabel1)
```