Lab: Clustering with K-means and Model-based Techniques.

Jay Singhvi

04/05/2024

In this lab, you will perform two types of clustering: k-means and model-based, on a medical data set. You will evaluate the clustering results for each technique, and compare the results to see if both techniques agree or not on the number of clusters. The data comes from the medical domain. The variables are:

test diagnosis- a factor with three levels: "Normal", "Prediabetic", "Diabetic".
glucose a measure of blood glucose level.
insulin a measure of blood insulin level.
sspg steady-state plasma glucose- derived from a measure of insulin level.

The categorical variable "test" is not involved in the clustering, but will be used to examine its distribution across each cluster. We would like to see the levels of test, "Normal", "Prediabetic", and "Diabetic", have a high correlation to cluster membership.

QUESTION 1 (2 parts, 4pts)- 46 total Lab points MAP TO GRADESCOPE

1a(2pts): Write a series of R statements which do the following:

1- Read in the data from the file "glucose_data.csv". 2- Ensure that the test variable is a factor. 3- Visualize the distribution of the levels of the variable test by creating a table on this vector. 4- Use the kable function to generate a table showing the levels and frequency counts. The col.names should be 'Condition' and 'Frequency', and the caption should be 'Test Levels in the Dataset'

```
g.data = read.csv("glucose_data.csv")
g.data$test = factor(g.data$test)
g.tbl = table(g.data$test)
kable(g.tbl, col.names = c("Condition", "Frequency"), caption = "Test levels in the Dataset")
```

Table 1: Test levels in the Dataset

Condition	Frequency
Diabetic	33
Normal	76
Prediabetic	36

1b(2pts): Use the scale function to create an object that contains the scaled values of all columns except the "test" column.

```
g.data_scaled = scale(g.data[-1])
#here [-1] means exclude the first row and we can also do this like [, 2:4] which also excludes row 1 a
```

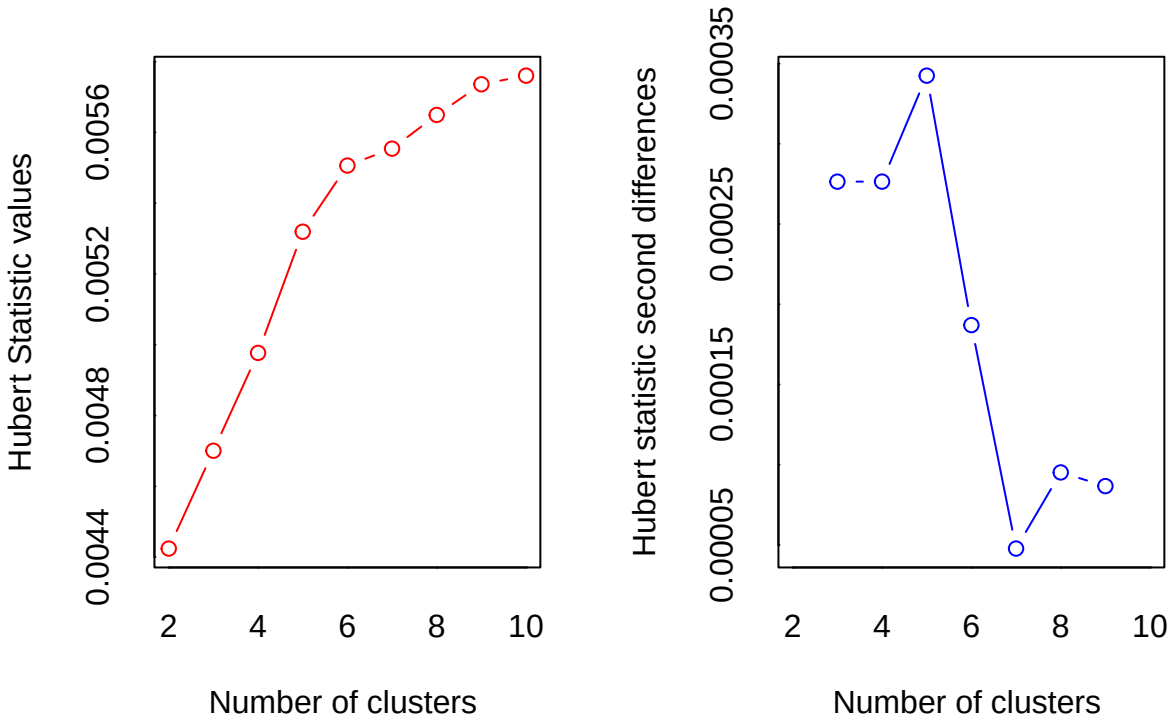
QUESTION 2 (4 parts, 8pts) MAP TO GRADESCOPE

Part 1: Clustering with K-means. 2a(2pts): Use the NbClust function to determine the optimal number of clusters using k-means. Use the scaled data from above, and set the minimum number of clusters to 2, the max to 10, The method parameter should be "kmeans".

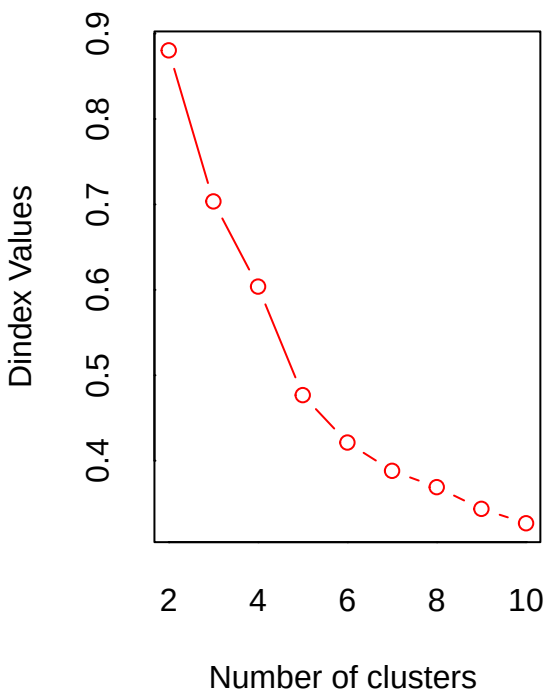
Assign the return of the call to NbClust to a variable as you will need it later.

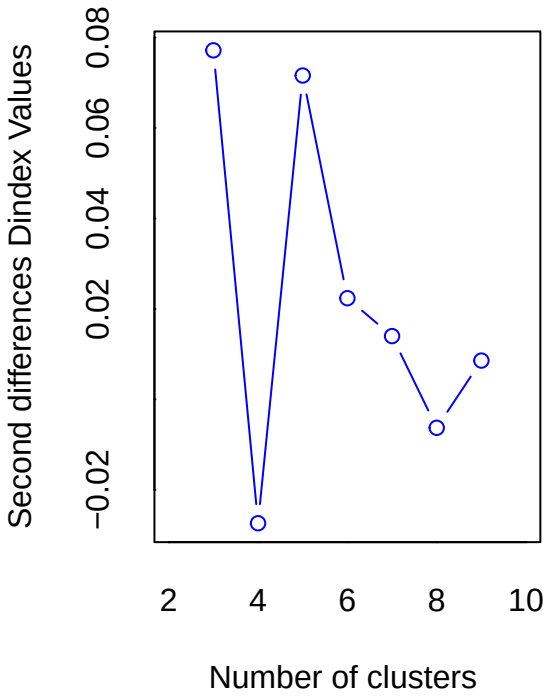
Note: the nbClust function can take a few seconds to complete- wait until the stop sign disappears and you see the cursor, >, re-appear in the console.

Display the table from the result of NbClust that shows the number of clusters on the top row and the bottom row shows the number of indices that reported that clustering as "best".




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```






```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 6 proposed 2 as the best number of clusters
## * 7 proposed 3 as the best number of clusters
## * 6 proposed 5 as the best number of clusters
## * 2 proposed 7 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 1 proposed 9 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************

##
##  0  2  3  5  7  8  9 10
##  2  6  7  6  2  1  1  1
```

2b(2pts): How many indices in total were used by NbClust to determine the optimal number of clusters?

26 indices

2c(2pts): According to NbClust, what is the optimal number of clusters?

3

2d(2pts): How many indices proposed the optimal number of clusters?

7

QUESTION 3 (5 parts, 10pts) MAP TO GRADESCOPE

Examine Cluster Assignments and "test" Variable. Since NbClust has executed k-means on the best k (which is 3), we can obtain the cluster assignments from the NbClust object. We can then make a contingency table on cluster assignments and the "test" vector from the dataframe.

3a(2pts):
Create a variable "cluster.assignments" and assign to it the vector of cluster assignments made by the NbClust function. This is the attribute "Best.partition" in the object returned from NbClust. Note: the "Best.partition" vector is an integer type, so make this a factor. Include a call to the summary, passing in the cluster assignments to print the summary of your "cluster.assignments" vector. You should have: 1 2 3 17 22 106

```
cluster.assignments<-factor(nc$Best.partition)
summary(cluster.assignments)
```

```
##   1   2   3
##  17  22 106
```

3b(2pts): Check the correlation between the "test" values and the cluster assignments. Create and print a contingency table on the cluster assignments and the "test" column from the data frame. You don't need to use the kable function here.

```
cont.tbl = table(cluster.assignments, g.data$test)
```

3c(2pts): Which cluster has the most "Normal" observations?

3

3d(2pts): The "purity" of a cluster can be calculated by this formula:

purity = max frequency/total number of data points in the cluster.

Calculate the purity of each cluster and report it below as a percent (to the nearest whole number):

cluster 1: 76 %, cluster 2: 100 %, cluster 3: 70 %,

3e(2pts): Run a chi-square test on the table.

```
chisq.test(table(cluster.assignments, g.data$test))
```

```
## Warning in chisq.test(table(cluster.assignments, g.data$test)): Chi-squared
## approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(cluster.assignments, g.data$test)
## X-squared = 115.21, df = 4, p-value < 2.2e-16
```

You should see a warning message that the Chi-squared approximation may be incorrect. This is because some of the frequencies in the contingency table are very small or 0.

QUESTION 4 (2 parts, 4pts) MAP TO GRADESCOPE

Fisher's exact test. To deal with low frequencies, run Fisher's exact test on the table.

Null Hypothesis: There is no a significant relationship between the two categorical variables. Alternate Hypothesis: There is a significant relationship between the two categorical variables.

If the null hypothesis is rejected, you can conclude that knowing the value of one variable helps to predict the value of the other variable. In terms of clustering, this means that there is a relationship between the variable and cluster membership.

4a(2pts): How to run the test in R? Do a search for "Fisher's exact test in R" for the syntax. Write and execute the statement that will perform the test in the chunk below.

```
test = fisher.test(cont.tbl)
```

4b(2pts): Answer the following questions based on the results of Fisher's exact test: 1- Can you accept or reject the null hypothesis? 2- Is there some correlation between the clustering and the "test" levels?

> 1- Reject null hypothesis 2- Yes, there is a significant relationship between the clustering and test levels

QUESTION 5 (2 parts, 4pts) MAP TO GRADESCOPE

Part 2: Model-based Clustering with the mclust Library Now use the Mclust function to take a model-based approach to these data. It is always a good idea to use several techniques in your analysis.

5a(2pts): Run the Mclust function on the scaled data and assign it to a variable called "model.clus". Call the "summary" function to display the cluster info.

```
model.clus = Mclust(g.data_scaled)
summary(model.clus)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 3
## components:
##
##  log-likelihood   n df       BIC       ICL
##       -169.0908 145 29 -482.5069 -501.4662
##
## Clustering table:
##  1  2  3
## 81 36 28
```

5b(2pts): What is the number of clusters produced by the Mclust function?

3

QUESTION 6 (5 parts, 10pts) MAP TO GRADESCOPE

Examine Cluster Assignments and "test" Variable. 6a(2pts): Check the correlation between the "test" values and the cluster assignments from Mclust (it's the "classification" attribute- remember to use the $ to access it in the object). Create and print a contingency table on the cluster assignments and the "test" column from the data frame. You don't need to use the kable function here.

```
mol.tbl = table(model.clus$classification, g.data$test)
```

6b(2pts): Which cluster has the most "Normal" observations? (Note that the cluster numbers in this model have no relationship to the cluster numbers from k-means).

1

6c(2pts): Calculate the purity of each cluster and report it below as a percent (to the nearest whole number):

cluster 1: 89%, cluster 2: 72 %, cluster 3:96 %,

6d(2pts): Run a chi-square test on the table.

```
chisq.test(mol.tbl)
```

```
##
##  Pearson's Chi-squared test
##
## data:  mol.tbl
## X-squared = 176.55, df = 4, p-value < 2.2e-16
```

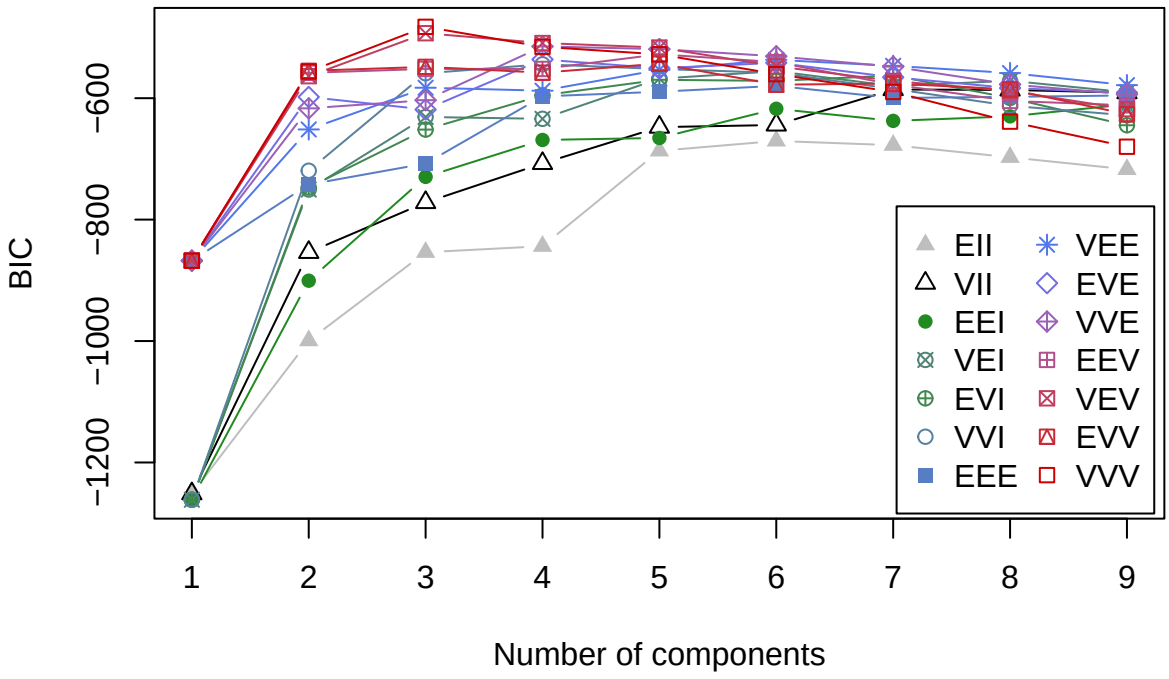
6e(2pts): Answer the following questions based on the chi-square test: Based on the results of the test, is there some correlation between the clustering and the "test" levels?

Yes there is a significant relation between clustering and test levels

QUESTION 7 (2 parts, 4pts) MAP TO GRADESCOPE

7a(2pts): Generate a plot of the clustering results for all 14 models used by Mclust. Call the plot function passing in the McClust object, the scaled data, and the what=BIC parameters. This will generate a line plot of each of the 14 models and its BIC score over the number of components (clusters) tested.

```
plot(model.clus, data = g.data_scaled, what = "BIC")
```



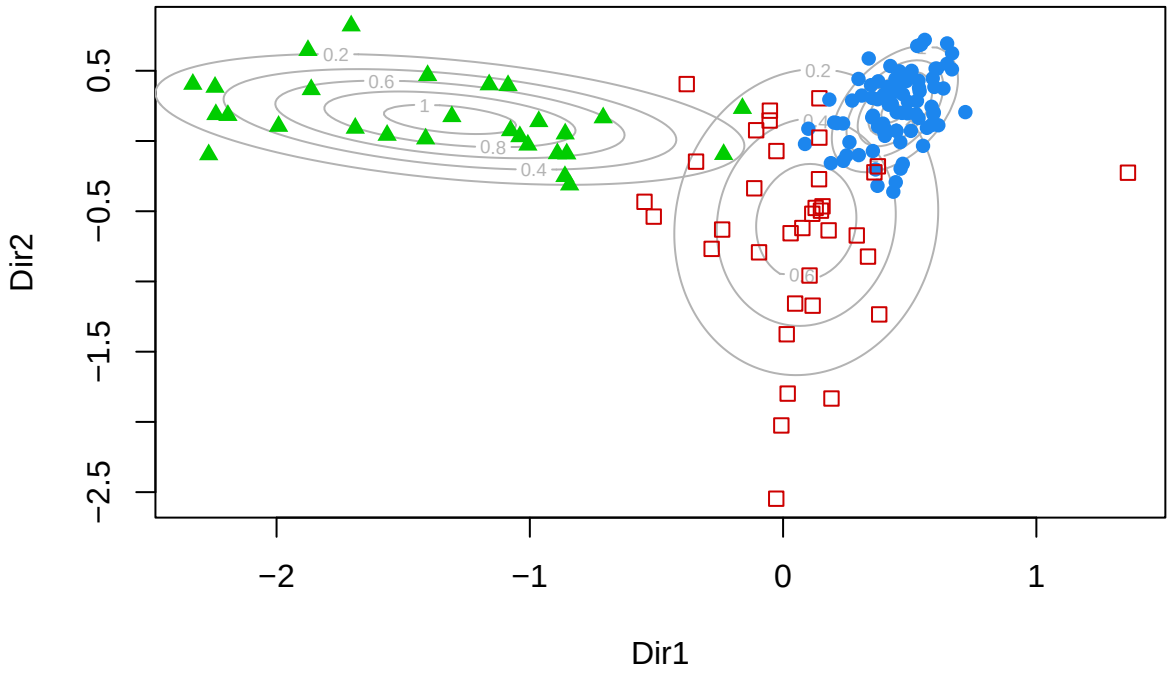
7b(2pts): Which model is the best for this clustering task at 3 clusters?

k means

QUESTION 8 (1 part, 2pts) MAP TO GRADESCOPE

This code uses dimension reduction for visualization and generates a plot of the model's clusters:

```
model.dr <- MclustDR(model.clus)
plot(model.dr, what = "contour")
```



Summary. Based on your analysis of the k-means and model-based clusterings, summarize by answering the following:

8a(2pts): 1- Did the k-means and model-based clustering produce similar results? If so, what were the similarities in the two clustering results? 2- How did the levels of the "test" variable correlate with the k-means and model-based clusters? 3- Which clustering had the best purity of grouping the "test" levels?

1- Both provided 3 clusters but the most pure clusters in both were different 2- Both provided that there was an significant relation between cluster and test levels 3- Model based clustering