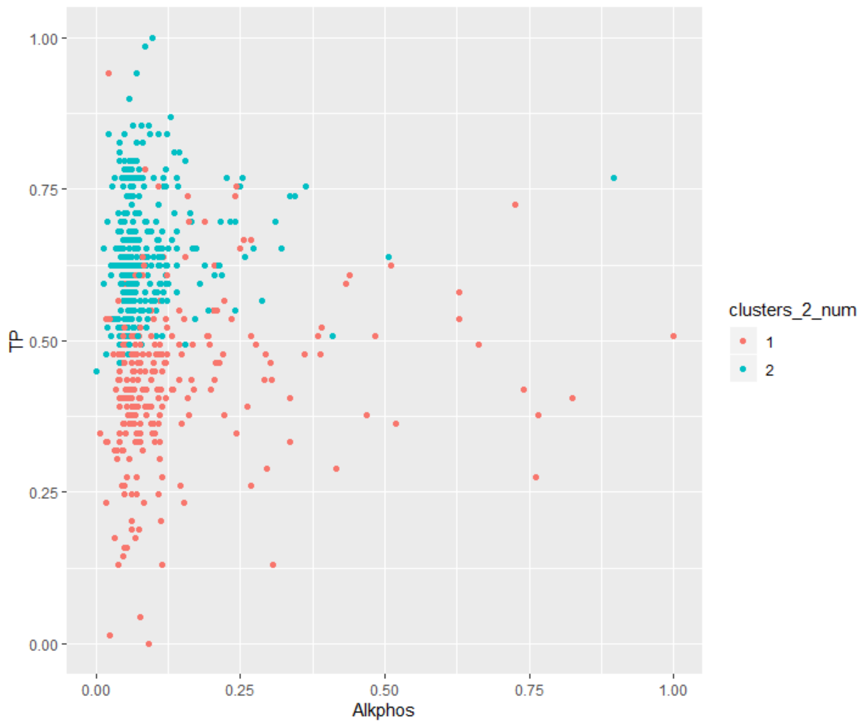
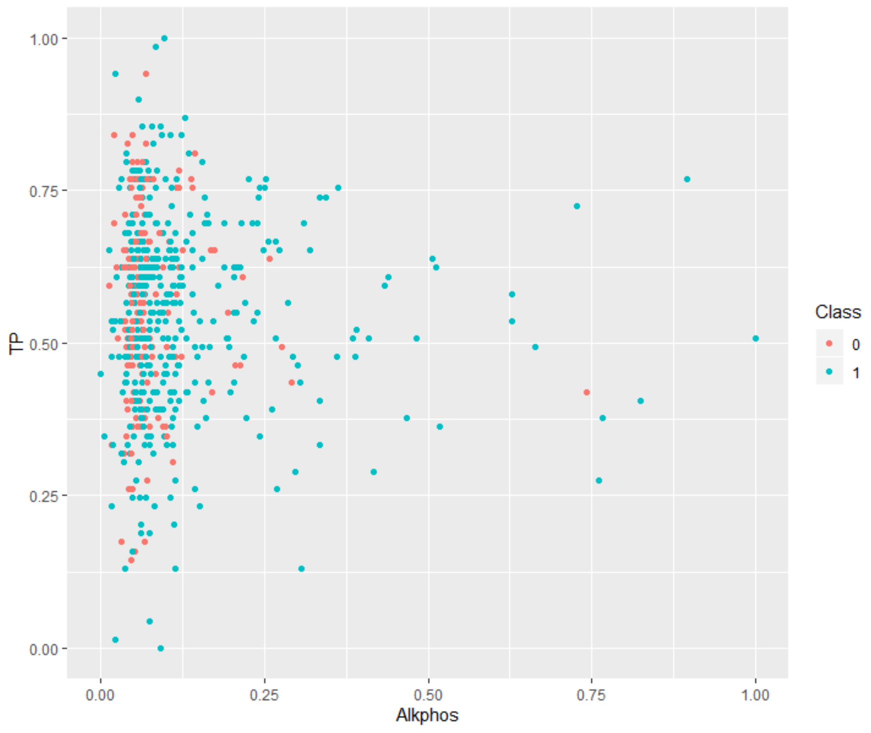
Task 2 – clustering

2.3 Cluster the data into 2 clusters (i.e. k=2) using K-Means clustering using the default parameters for the kmeans function. Plot the results of the clusters as a 2D plot where the x-axis is Alkphos and the y-axis is TP.



2.4 Plot another 2D plot with the same dimensions above, but color the points according to the Class column.

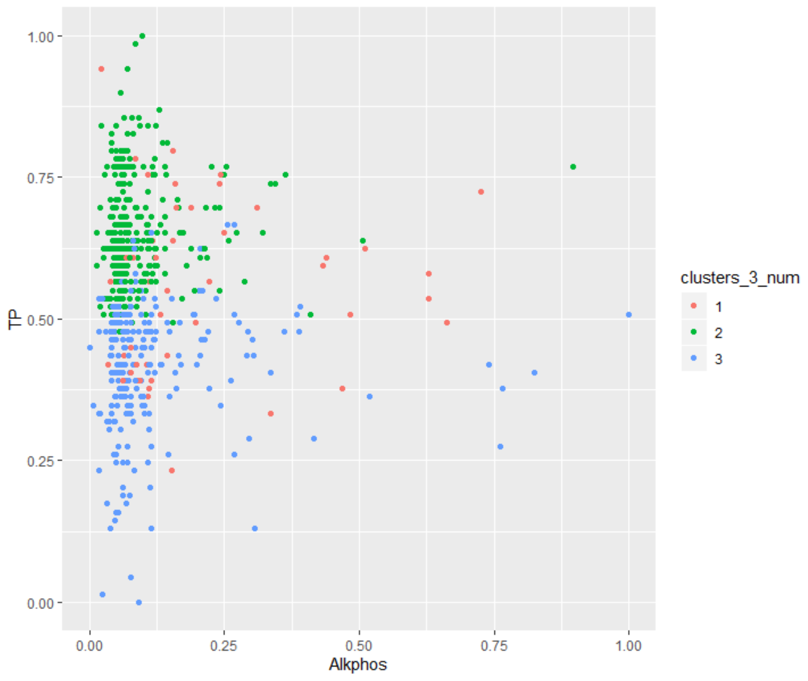


2.5 Compare the 2 plots obtained in tasks 2.3 and 2.4 – do the clusters visually represent the patient vs non-patient classes?

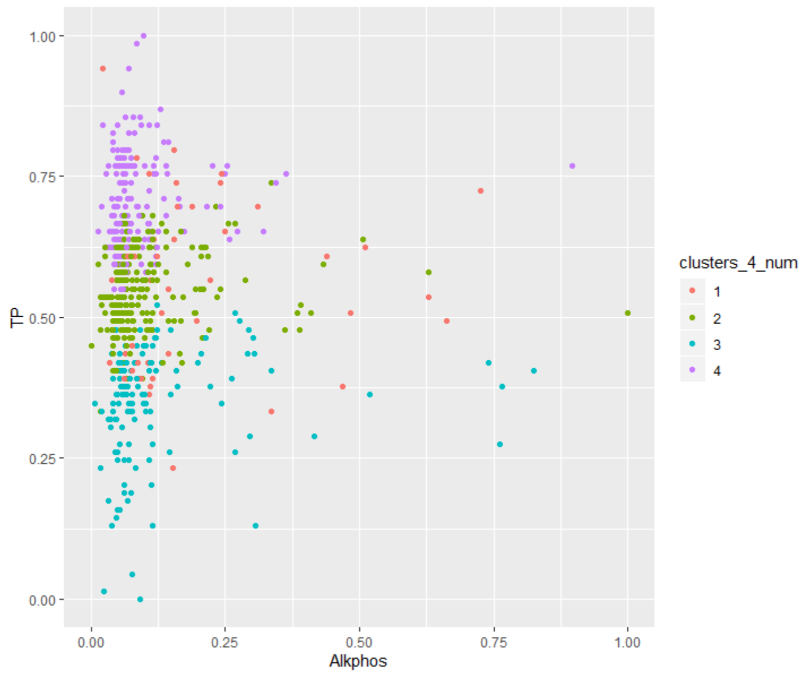
As two graphs show above, the clusters do not visually represent the patient and non-patient classes with TP and Alkphos attributes. From the graph with the points plot according to the Class column, patient and non-patient classes are overlapping. However, K-means is hard clustering approach with non-overlapping clusters. Each cluster have a centroid and each instance is assigned to the cluster with the closest centroid. Therefore, the clusters could not visually represent the patient and non-patient classes.

2.6 Cluster the data into more than 2 clusters (i.e. k=3,4,5) using K-Means clustering and plot all the clustering results.

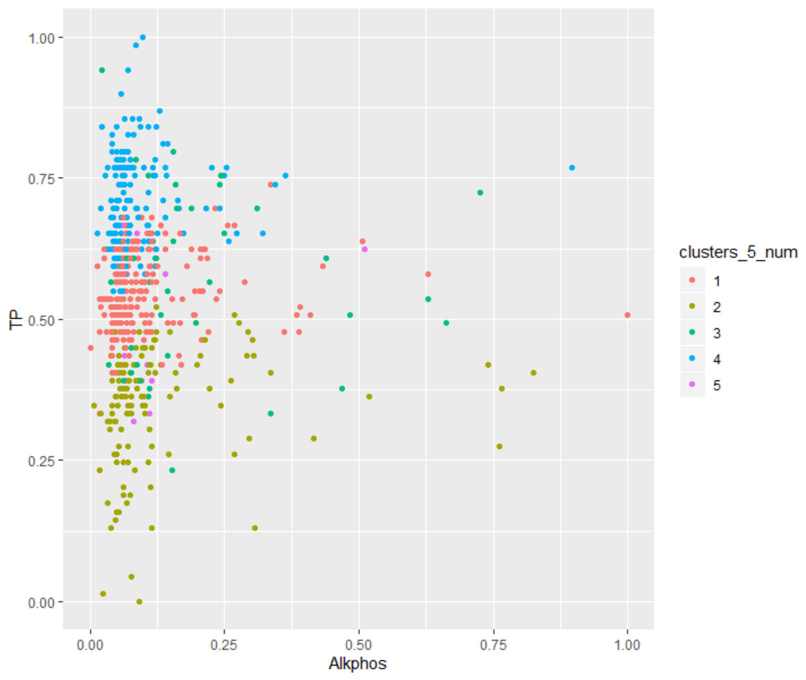
k=3:



k=4:



k=5:



2.7 Compare the plots and sum of squared error (SSE) obtained in previous task and provide your comments on the quality of clustering

Clustering	SSE
K = 2	43.4730
K = 3	31.9672
K = 4	27.2022
K = 5	23.2871

SSE (sum of squared error) is the most common measure for k-means clusters. For each point, the error is the distance to the nearest cluster/centroid, then we can get SSE by square these errors and sum them. The formula below is used to calculate SSE.

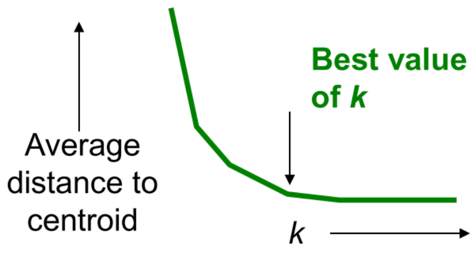
$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{x \in C_i} \mathrm{dist}^2(c_i, x)$$

K – the number of clusters

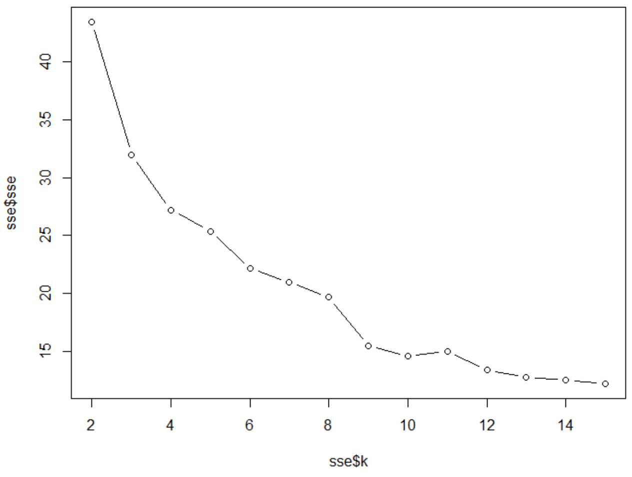
x – a data point in cluster C_i

c_i – the centroid point for cluster C_i

The smaller the SSE, the better the clustering. One way to reduce SSE is to increase k, which is the number of clusters. A good clustering should have a smaller k that can have a lower SSE. The k value for k-means method should choose the k that when average distance to centroid falls rapidly until right k, then changes little (shown in the picture below).



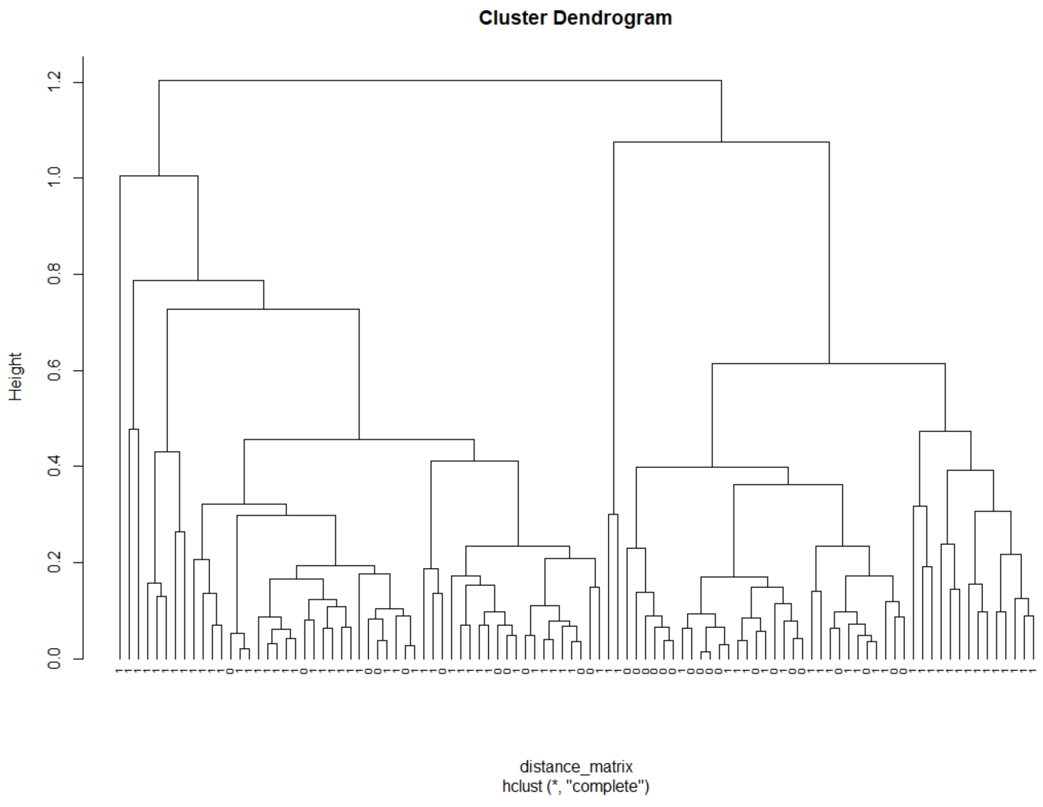
The graph below plots the SSE with k value from 2 to 15.



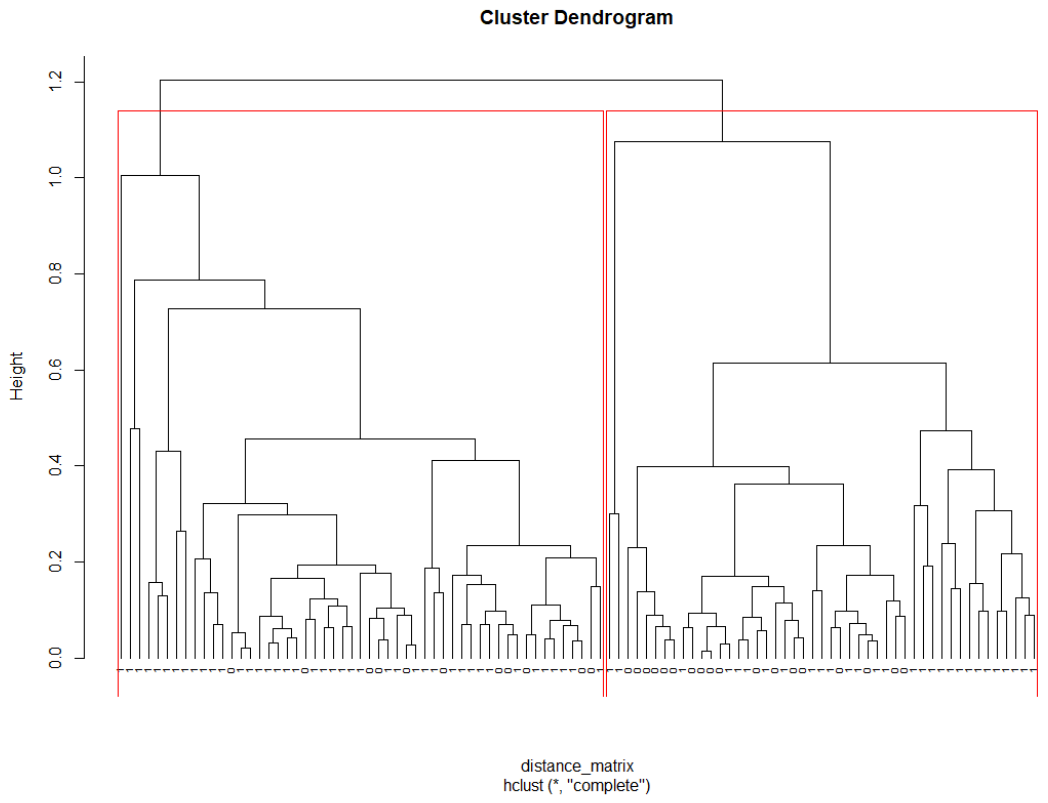
It shows that SSE is dropping rapidly from 2 to 5. The table in last page shows the SSEs for clusterings with k value from 2 to 5. The clustering with k =5 has the smallest SSE, therefore it is the best clustering.

2.8 Apply hierarchical clustering to the data using the hclust function with default parameters and plot the corresponding dendrogram. Particularly, cluster the dendrogram into 2, 3, 4 and 5 clusters and plot all of them.

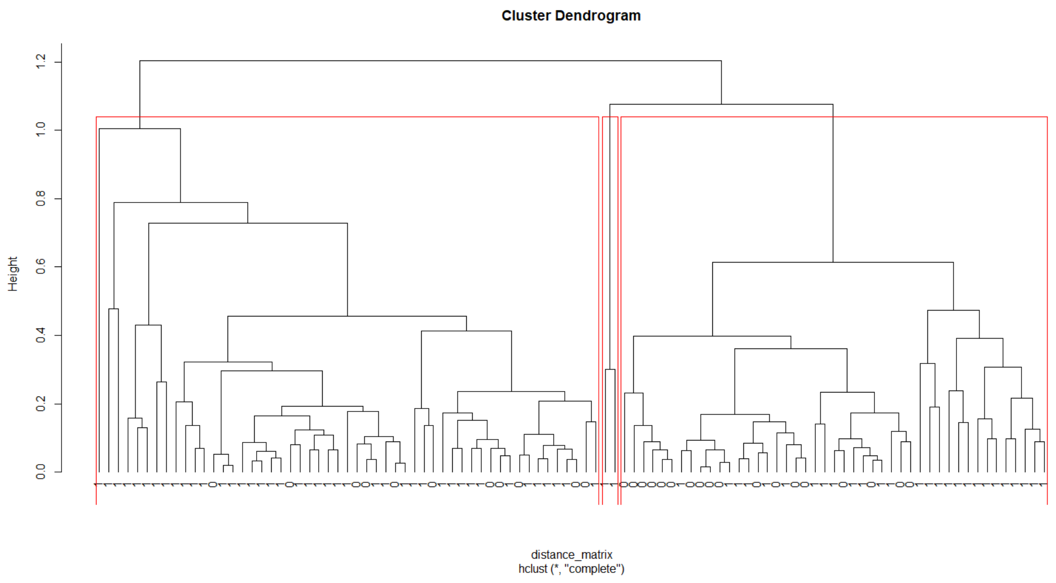
I random uniform sample with 100 instances and draw the dendrograms based on the sample data.



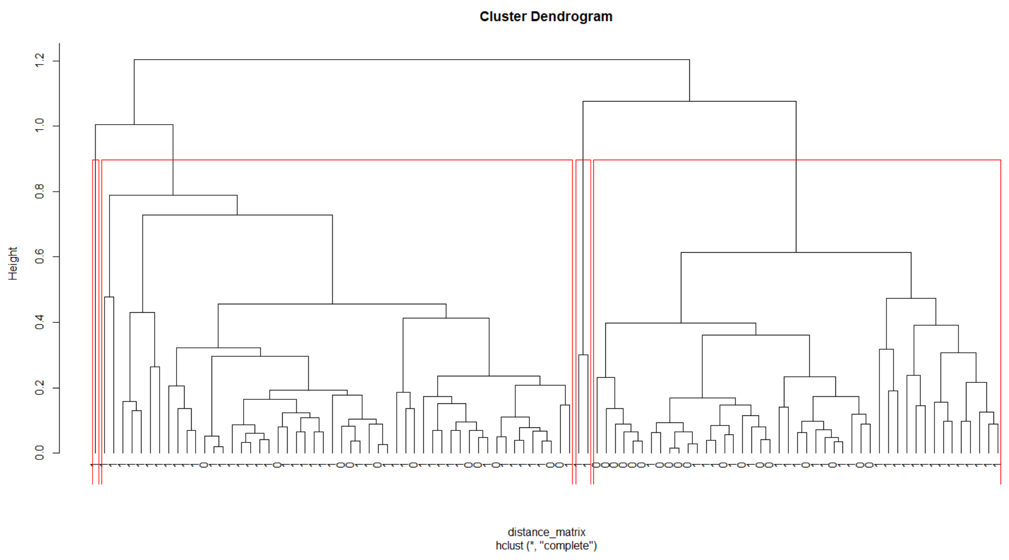
Cluster the dendrogram into 2 clusters:



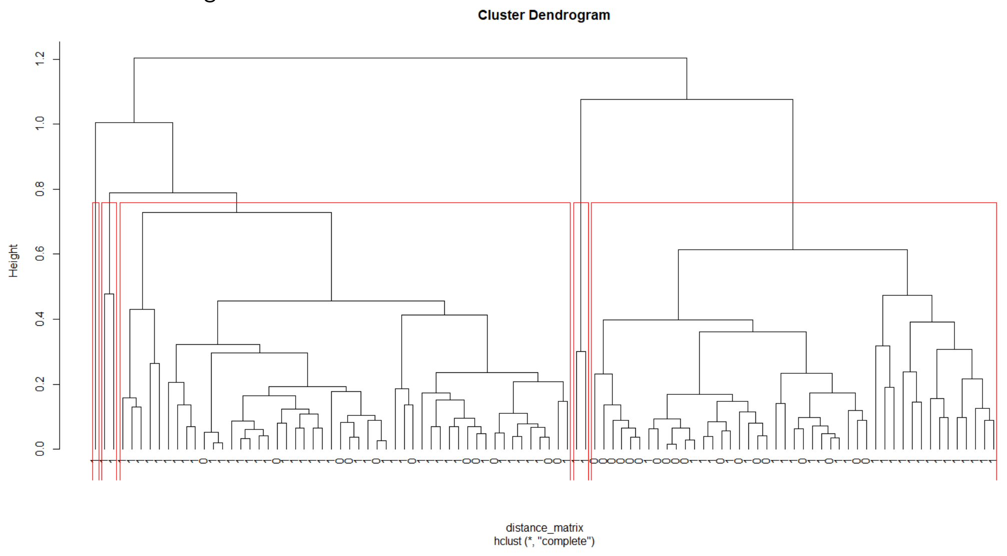
Cluster the dendrogram into 3 clusters:



Cluster the dendrogram into 4 clusters:



Cluster the dendrogram into 5 clusters:



2.9 Compare the plots obtained in the tasks 2.3, 2.4, 2.6 and 2.8 and provide your observations on the achieved clusters – should we have a new subtype of diseases?

The tables below show the relationship between the clustering and actual class for hierarchical clusterings with 2, 3, 4 and 5 clusters respectively.

Hierarchical clusterings with 2 clusters:

Clustering / Actual	1	2
1	29	42
0	18	11

Hierarchical clusterings with 3 clusters:

Clustering / Actual	1	2	3
1	27	42	2
0	18	11	0

Hierarchical clusterings with 4 clusters:

Clustering / Actual	1	2	3	4
1	27	41	2	1
0	18	11	0	0

Hierarchical clusterings with 5 clusters:

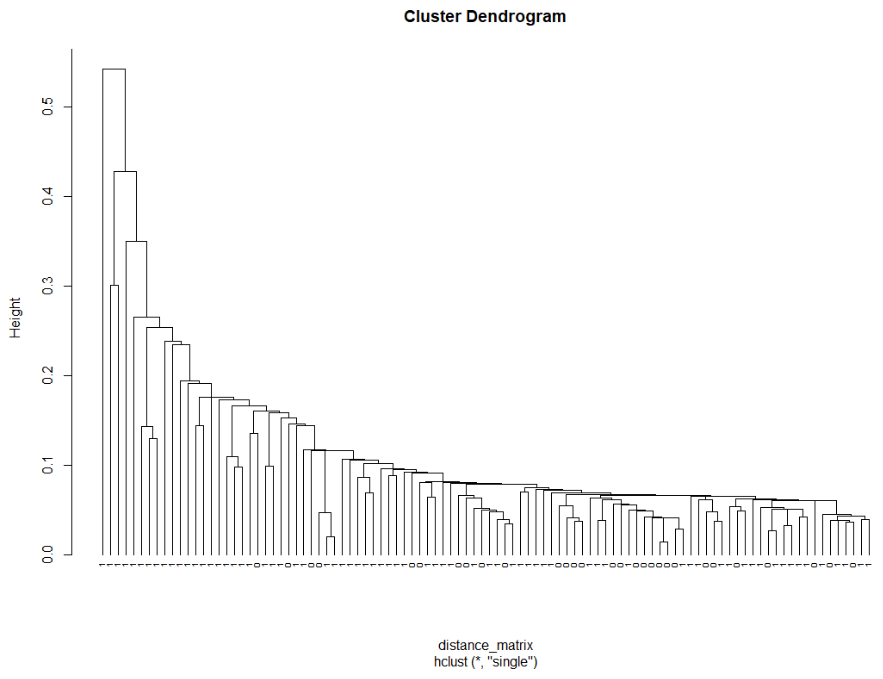
Actual \ Clustering	1	2	3	4	5
1	27	39	2	2	1
0	18	11	0	0	0

For hierarchical clustering method, the graphs in question 2.8 show that splitting the hierarchical clustering into 2 clusters have almost evenly number of instances in each cluster. However, when splitting into more than 2 clusters, there would be only 2 clusters contain the main part of the data, and the rest of the clusters only have a few numbers of data. In addition, for hierarchical clustering with 2 clusters, both clusters have more instances that belong to actual class label 1 (patient class). Therefore, hierarchical clustering approach may not be good for this case.

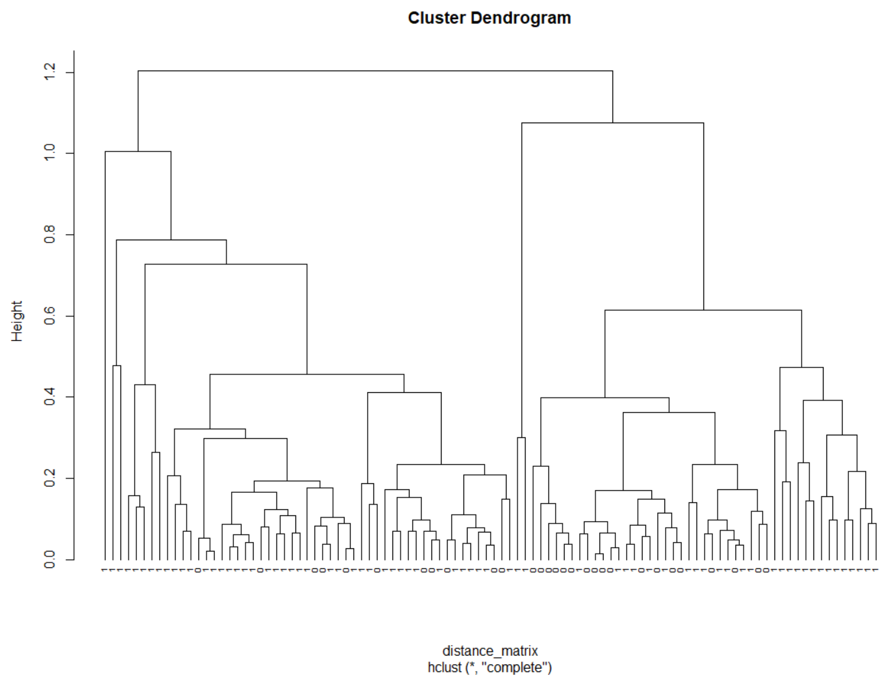
For the K-Means method, as I have explained in question 2.5, the K-Means with k = 2 cannot visually represent the non-patient and patient classes well. However, when k value increases, SSE for the decrease (shown in question 2.7), which means that the more the number of clusters, the better the performance of the clustering. This may imply that we should have a new subtype of diseases as a class label.

2.10 Try different agglomeration methods in hierarchical clustering (i.e. “MIN”, “MAX” and “AVERAGE”). Plot the resulting dendrograms and provide your comments on the quality of clustering – is the data sensitive to the used agglomeration method? Based on your results, what do you think is the default agglomeration method used in task 2.8

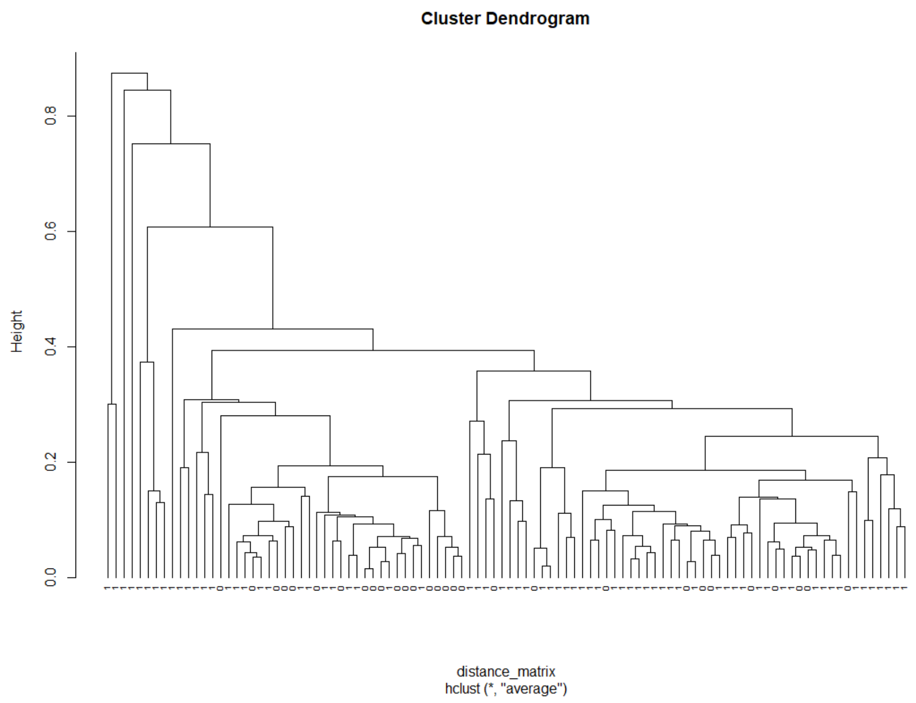
“MIN” agglomeration methods:



“MAX” agglomeration methods:



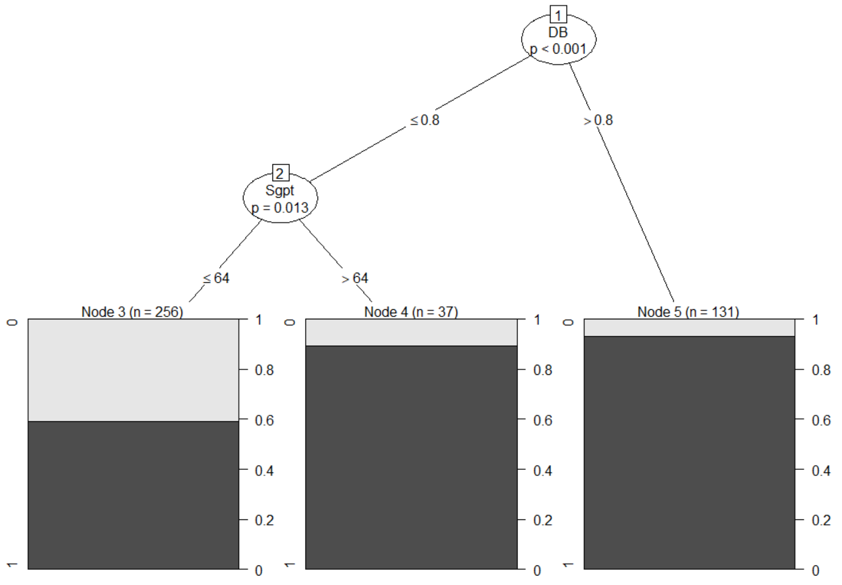
“AVERAGE” agglomeration methods:



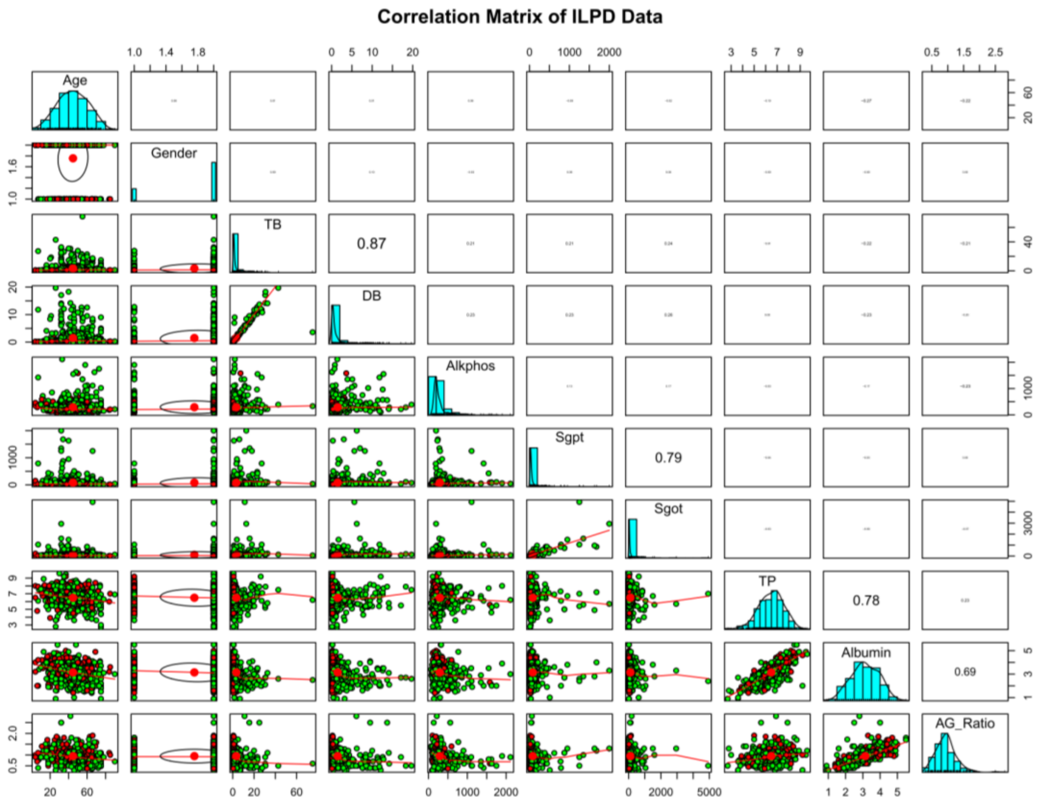
The data is sensitive to different agglomeration methods. Different approaches define the distance between clusters in different ways. “MIN” is using the two closest points in the different clusters to calculate the distance between clusters, while “MAX” calculate the proximity of two clusters is based on the two most distant points in the different clusters. “AVERAGE” is using the average of pair-wise proximity between points in the two clusters. As the graphs show above, different agglomeration methods would generate different hierarchical clusters. The default agglomeration method used in task 2.8 is complete (max).

Task 3 – classification

3.2 Plot that classification tree and provide your comments on its structure (e.g. what are the important/unimportant variables? Is there any knowledge we can infer from the tree representation that helps in differentiating between the classes ?). Using the learned tree, predict the class labels of the test data. Calculate the accuracy, precision, and recall.



The graph above shows that the decision tree splits in binary and firstly splits with DB attribute. If DB is greater than 0.8, it would assign to class label 1 (patient). If the DB is less than or equal to 0.8, it would split again based on Sgpt value. The Sgpt value greater than 64 would assign to class label 1 (patient), and class label 0 (non-patient) would be assigned to the instance with Sgpt less than or equal to 64.



From the correlation matrix in last page, TB and DB have the highest correlation (0.87), Sgpt and Sgot, TP and Albumin have the second and third highest correlation (0.79 and 0.78 respectively). Albumin and AG_Ratio have the fourth highest correlation (0.69). Therefore TB, DB, Sgpt, Sgot, TP, Albumin and AG_Ratio are relatively important variables, others (includes Age, Gender, Alkphos) are unimportant variables.

Predict / Actual	1	0
1	110	49
0	0	0

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{110 + 0}{110 + 0 + 49 + 0} = 0.6918239$$

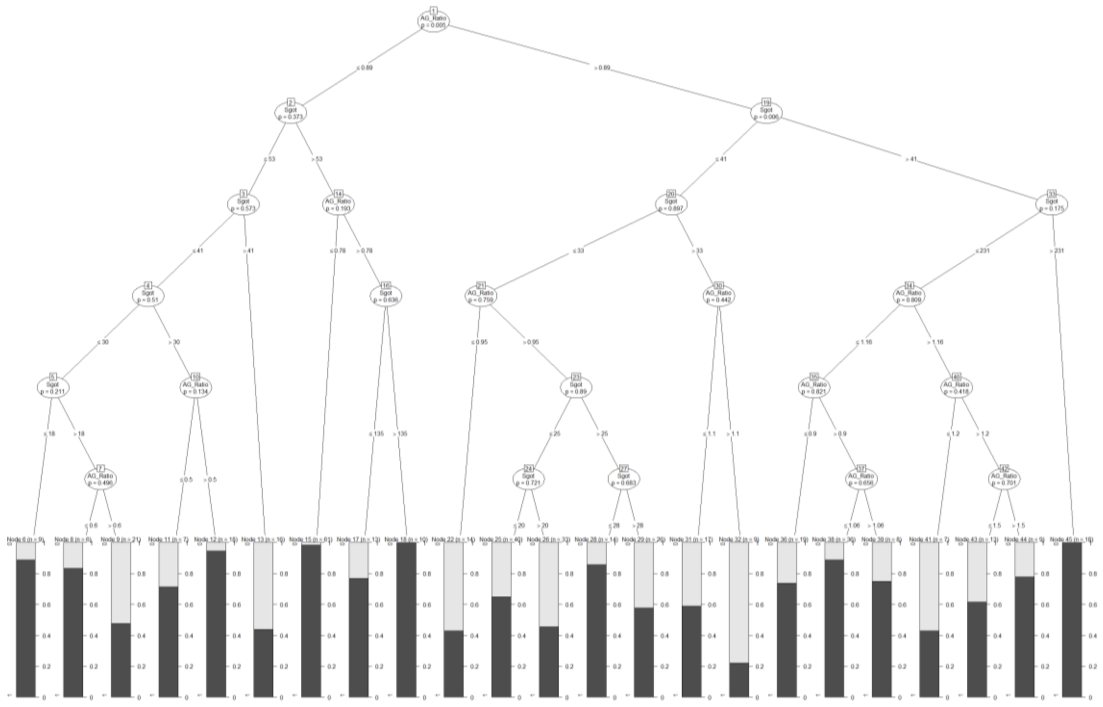
$$Precision = \frac{TP}{TP + FP} = \frac{110}{110 + 0} = 1$$

$$Recall = \frac{TP}{TP + FN} = \frac{110}{110 + 49} = 0.6918239$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{1 \times 0.6918239}{1 + 0.6918239} = 0.8178439$$

Using the learned tree to predict the class labels of the test data which having accuracy 69.18%, precision 100%, recall 69.18% and f1 81.78%.

3.3 Can you achieve better accuracy or more meaningful representation by tuning some parameters?



$$\begin{aligned} ct <- ctree(&Class \sim Sgot + AG\_Ratio, data = train\_data, controls \\ &= ctree\_control(maxdepth = 6, minbucket = 5, mincriterion \\ &= 0.083)) \end{aligned}$$

I use the formula $Class \sim Sgot + AG\_Ratio$ instead of $Class \sim .$ (formula used for question

3.2). As Sgot and AG_Ratio are two of the important variables. Minbucket is the minimum number of observations in any terminal node. Mincriterion is the value of the test statistic $(1 - p\ value)$ that must be exceeded in order to implement a split. With a smaller mincriterion, it would have a less restrict to implement a split. Therefore, in this trained decision tree would have more split. With many splits contain in the trained decision tree, it always would improve the accuracy. Maxdepth which is the maximum depth of the tree, which set as 6 in order to restrict the depth of the tree.

Predict / Actual	1	0
1	94	28
0	16	21

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{94 + 21}{94 + 21 + 16 + 28} = 0.7232704$$

$$Precision = \frac{TP}{TP + FP} = \frac{94}{94 + 16} = 0.8545455$$

$$Recall = \frac{TP}{TP + FN} = \frac{94}{94 + 28} = 0.7704918$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8545455 \times 0.7704918}{0.8545455 + 0.7704918} = 0.8103448$$

Using the above learned tree to predict the class labels of the test data which having accuracy 72.33%, precision 85.45%, recall 77.05% and f1 81.03%. It improves the accuracy but have a lower f1.

3.4 Apply K-NN classification to predict the labels in the test subset and calculate the accuracy, precision and recall. Particularly, try different values of K (e.g. K = 1,2,3,4,5) and report your observations on the achieved classification.

K = 1:

Predict / Actual	1	0
1	86	35
0	24	14

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{86 + 14}{86 + 14 + 24 + 35} = 0.6289308$$

$$Precision = \frac{TP}{TP + FP} = \frac{86}{86 + 24} = 0.7818182$$

$$Recall = \frac{TP}{TP + FN} = \frac{86}{86 + 35} = 0.7107438$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.7818182 \times 0.7107438}{0.7818182 + 0.7107438} = 0.7445887$$

K = 2:

Predict / Actual	1	0
1	88	33
0	22	16

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{88 + 16}{88 + 16 + 22 + 33} = 0.6540881$$

$$Precision = \frac{TP}{TP + FP} = \frac{88}{88 + 22} = 0.8$$

$$Recall = \frac{TP}{TP + FN} = \frac{88}{88 + 33} = 0.7272727$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8 \times 0.7272727}{0.8 + 0.7272727} = 0.7619047$$

K = 3:

Predict / Actual	1	0
1	93	37
0	17	12

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{93 + 12}{93 + 12 + 17 + 37} = 0.6603774$$

$$Precision = \frac{TP}{TP + FP} = \frac{93}{93 + 17} = 0.8454545$$

$$Recall = \frac{TP}{TP + FN} = \frac{93}{93 + 37} = 0.7153846$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8454545 \times 0.7153846}{0.8454545 + 0.7153846} = 0.775$$

K = 4:

Predict / Actual	1	0
1	93	38
0	17	11

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{93 + 11}{93 + 11 + 17 + 38} = 0.6540881$$

$$Precision = \frac{TP}{TP + FP} = \frac{93}{93 + 17} = 0.8454545$$

$$Recall = \frac{TP}{TP + FN} = \frac{93}{93 + 38} = 0.7099237$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8454545 \times 0.7099237}{0.8454545 + 0.7099237} = 0.7717842$$

K = 5:

Actual \ Predict	1	0
1	94	38
0	16	11

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{94 + 11}{94 + 11 + 16 + 38} = 0.6603774$$

$$Precision = \frac{TP}{TP + FP} = \frac{94}{94 + 16} = 0.8545455$$

$$Recall = \frac{TP}{TP + FN} = \frac{94}{94 + 38} = 0.7121212$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8545455 \times 0.7121212}{0.8545455 + 0.7121212} = 0.7768595$$

K	Accuracy	Precision	Recall	F1
1	0.6289308	0.7818182	0.7107438	0.7445887
2	0.6540881	0.8	0.7272727	0.7619047
3	**0.6603774**	0.8454545	0.7153846	0.775
4	0.6540881	0.8454545	0.7099237	0.7717842
5	**0.6603774**	0.8545455	0.7121212	**0.7768595**

For the classifiers with k = 1,2,3,4,5 respectively, classifiers with k = 3 and k = 5 have the highest accuracy. Classifier with k = 5 have higher F1 score compare to classifier with k = 3. Therefore, K-NN classifier with k = 5 is the best.