**CENG 464**

**INTRODUCTION TO DATA MINING**

**Project Report**

**Eren Berk SALTAŞ**

**201711056**

**İlayda GÖLEN**

**201811032**

**07.01.2021**

Contents

[1. Introduction and Aim 1](#_Toc397291483)

[2. Explanation of Code and Methods 1](#_Toc397291484)

[3. Bonus Step 2](#_Toc397291493)

[4. Results 3](#_Toc397291485)

[5. Conclusion 1](#_Toc397291493)2

[Appendix 1](#_Toc397291494)7

1. Introduction and Aim

Since our id is even, we used data2. We are expected to use R language in our project. There are five steps we are asked to do for the project. The first is to preprocess the data. Second step, not mandatory, is to find top 5 using Countries\_Unique\_Count and Countries\_First\_Author attributes. The third step is to select features that are important for analysis. The next step is to apply clustering algorithms for the output variable J\_or\_C, compare their results, and apply classification models. The last and the fifth step is to apply the operations in the fourth step for the output variable Quartile.

The most crucial point in the project is the preprocessing of the data. With the new data we have preprocessed, we aim to complete the clustering and classification processes in the most successful way. At the same time, we aim to consolidate our knowledge by seeing the methods we know in theory, in practice.

1. Explanation of Code and Methods

We wrote our code as a whole, not step by step, for the 5 steps requested from us. First of all, we added the necessary libraries to our code. When the names(my\_data) [nearZeroVar (my\_data)] function is written to the console, it prints columns with variance close to 0 to us. Using this, we assign NULL values to some of the attributes according to the results we get. In order to preprocess the data, we converted the J\_or\_C attributes to numeric. We copied the data for each of the J\_or\_C and Quartile attributes before deleting them. While we named these copied ones as my\_data\_w, we created a new copy as my\_data\_with. my\_data\_with, the data of the version with these attributes and we created it for comparison. Next, we deleted the Quartile attribute from the data my\_data\_w and made the J\_or\_C attribute numeric for both my\_data\_w and my\_data\_with. Likewise, we created two copies of the J\_or\_C attribute, my\_data\_w\_jc and my\_data\_with\_jc. Then, we wrote the function cor (my\_data, my\_data $ Quartile) and updated the my\_data\_w \_q that we prepared for the Quartile attribute according to the results. We have added the correlation results to appendix. We did the same for my\_data\_with\_jc. This function returns columns that might be associated with that attribute. Then, we chose those columns with select function and we set those chosen columns sync to the same vector variable. In this way, the columns in the data are processed. Next, we wrote four for loops for the four data copies we created.

If there is any missing value in the data, we did this to fill the empty space with the mean of that column so that the missing data would not fail while preprocessing. After scaling the new data we created, we started the clustering and classification process. In order to capture the maximum value without any loss from the processed data, the tree classification process was performed immediately after preprocessing.

We used K-means clustering and Hierarchical clustering (Ward’s method), Tree classification and Unsupervised classification. The same process is applied for different attributes for the fourth and fifth steps. Before kmeans clustering, tree classification was performed for both attribute J\_or\_C and Quartile. To give an example for the J\_or\_C attribute, we calculated using the kmeans function for my\_data\_with\_jc and my\_data\_w\_jc. In this way, k-means clustering and unsupervised classification are done. Two different clusters are created, using and not using the J\_or\_C column. Later, we visualized using autoplot. We used the fviz\_nbclust function we found on the Internet to test the accuracy of our own work. The fviz\_nbclust function calculates the optimal number of clusters. Then we did hierarchical clustering and unsupervised classification. Again, one is the attribute is included and the other is not. We plotted it using ggdendrogram.

1. Bonus Step

After vectorizing the column with the header, when we make it a table using the table function, we see that it arranges the data in the column according to the frequency of repetition. Then we achieve the desired goal by using tale and sort functions together. Our results are as follows.

Top5 countries:



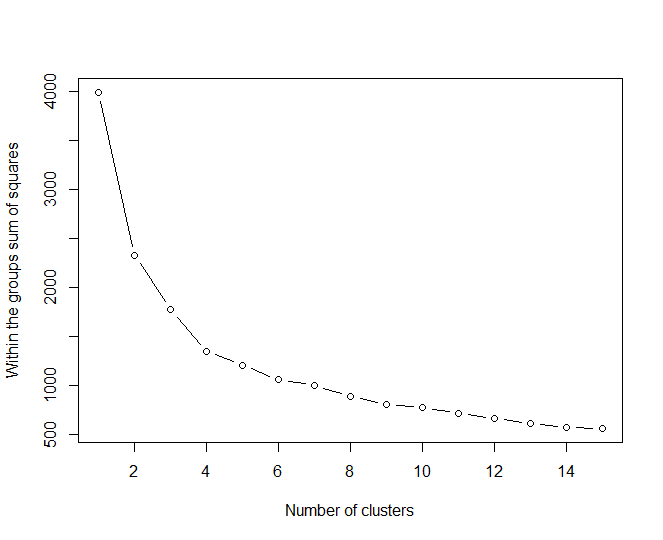
Top5 first author’s countries:



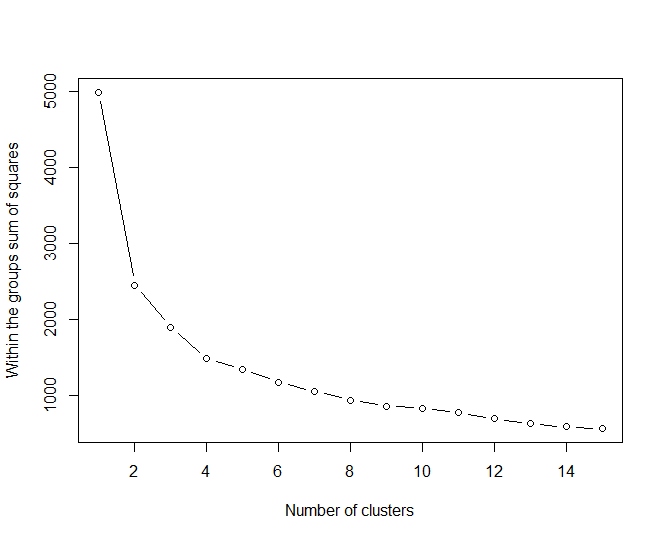
1. Results

Before all else, these results are wssplot, k-means cluster, k-means unsupervised classification, hierarchical clustering and fviz\_nbclust for J\_or\_C and Quartile attributes. To begin with, we would like to show the results of the J\_or\_C attribute.

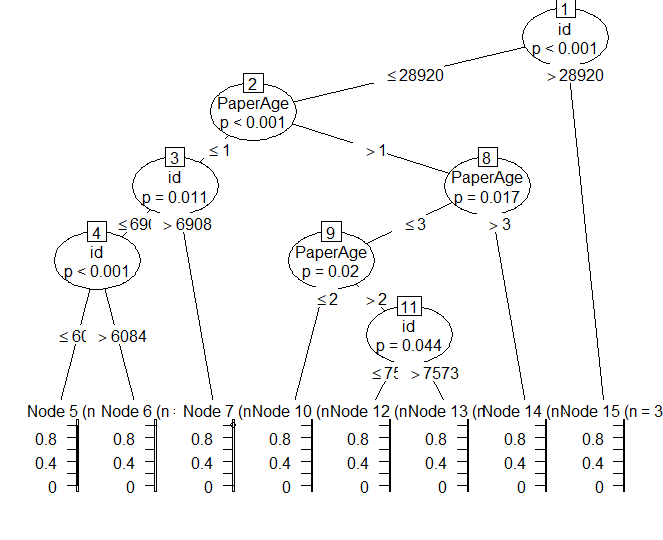
wssplot result for without J\_or\_C attribute:



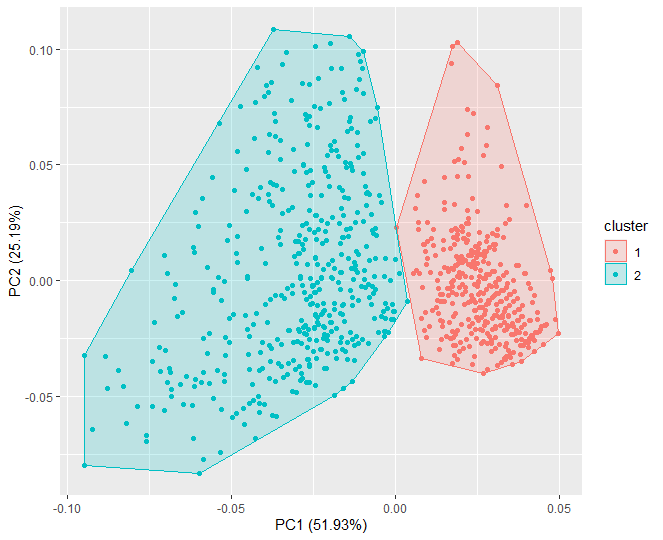
wssplot result for with J\_or\_C attribute:



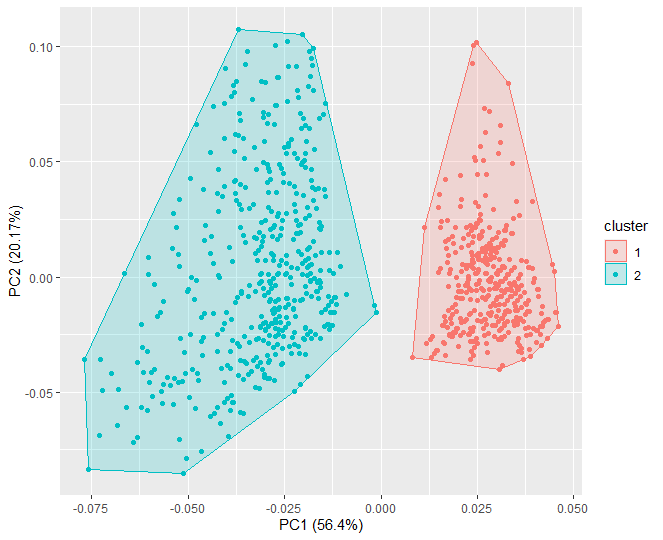
Tree classification result for with J\_or\_C attribute:



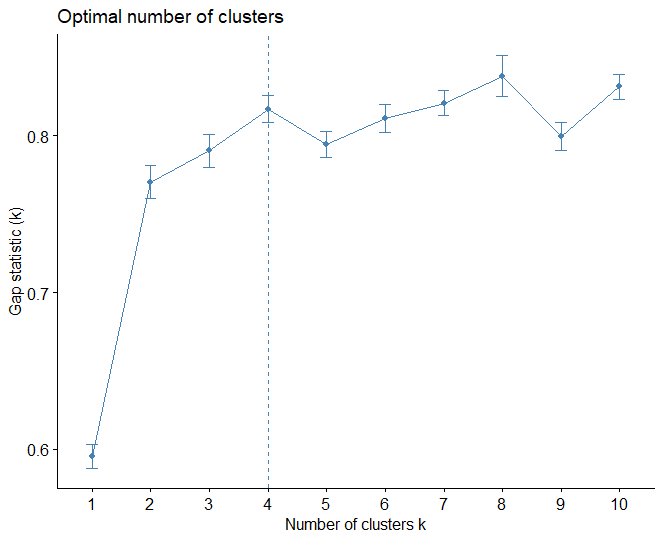
k-means result for without J\_or\_C attribute:



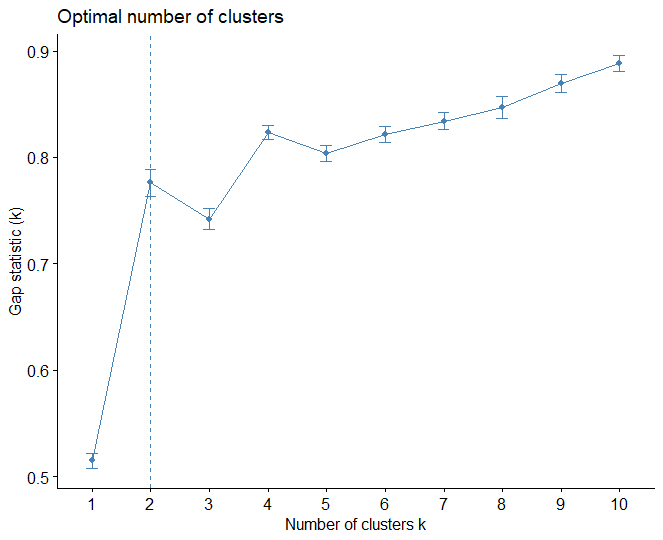
k-means result for with J\_or\_C attribute:



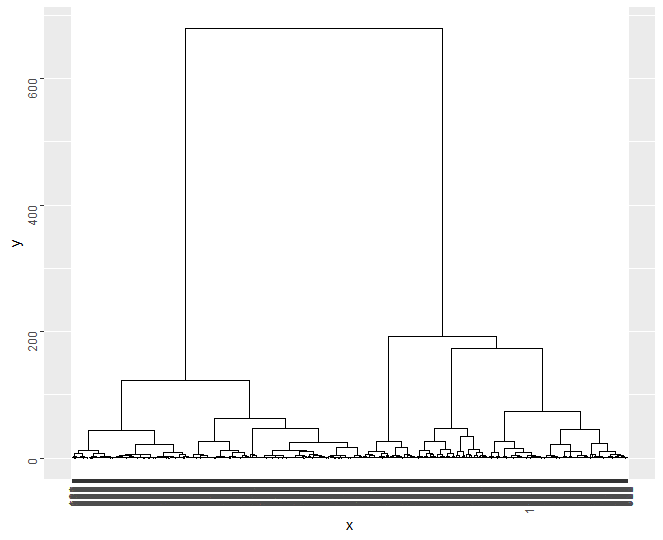
fviz\_nbclust result for without J\_or\_C attribute:



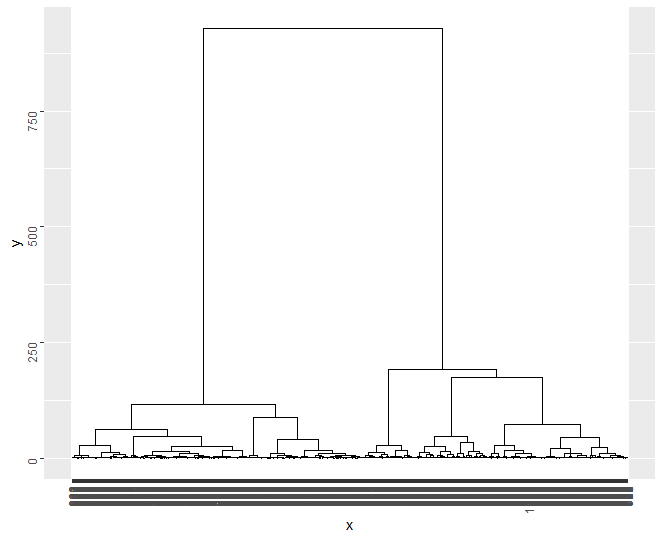
fviz\_nbclust result for with J\_or\_C attribute:



Hclust result for without J\_or\_C attribute:

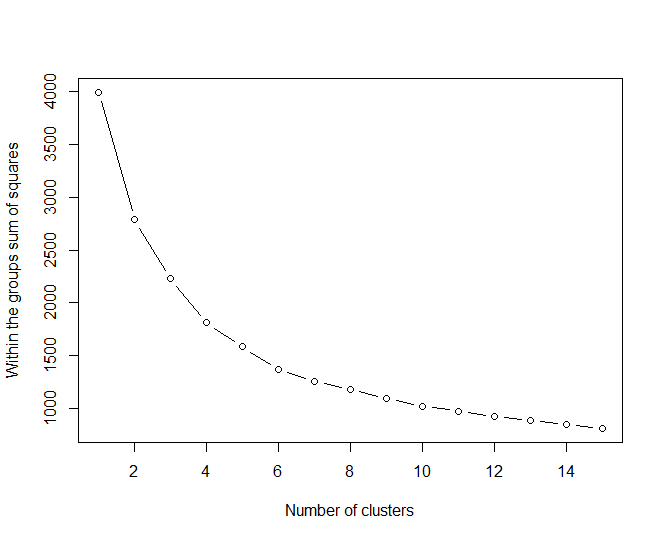


Hclust result for with J\_or\_C attribute:

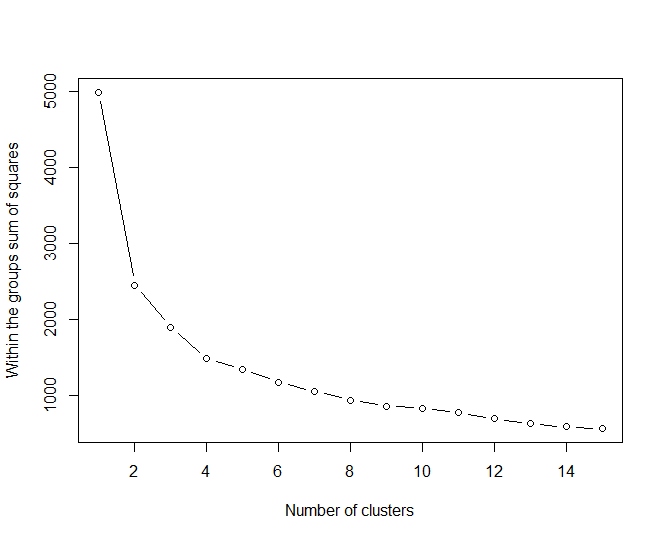


In addition to these, the results of the Quartile attribute are below.

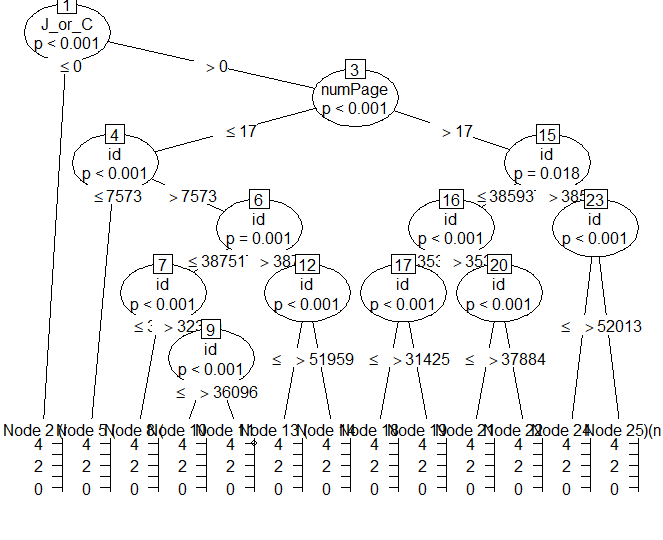
wssplot result for without Quartile attribute:



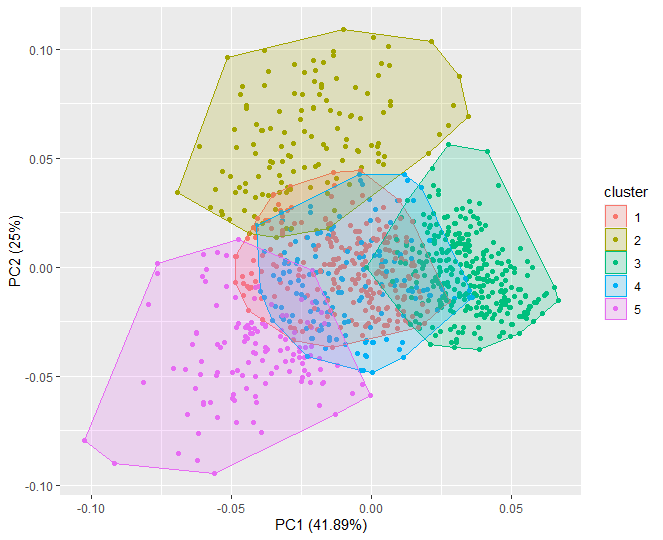
wssplot result for with Quartile attribute:



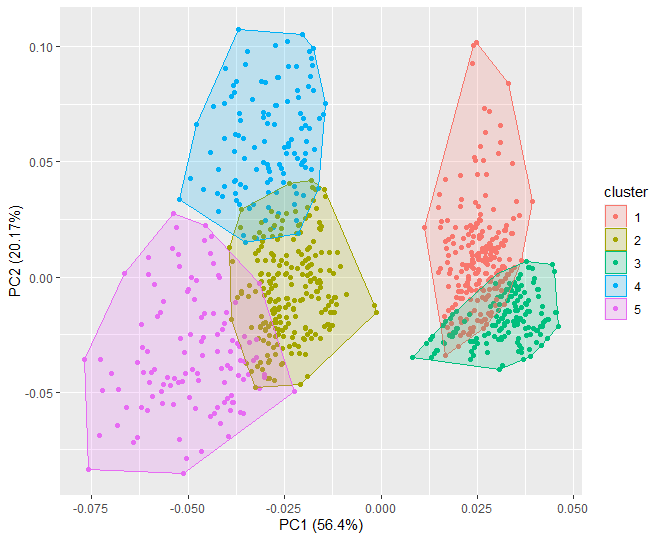
Tree classification result for with Quartile attribute:



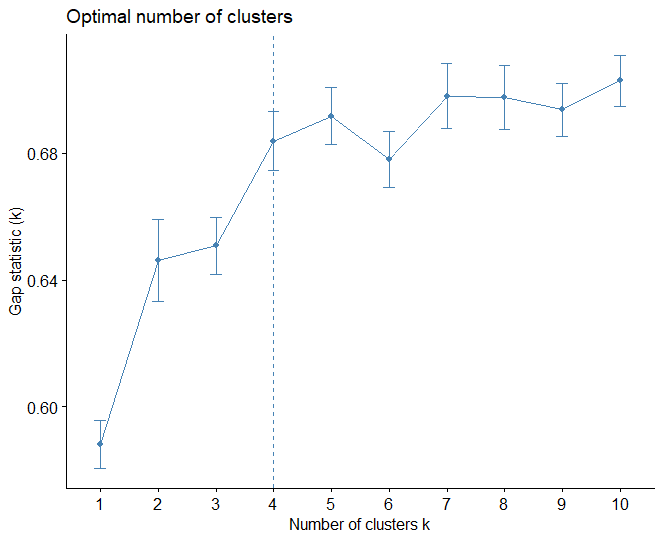
k-means result for without Quartile attribute:



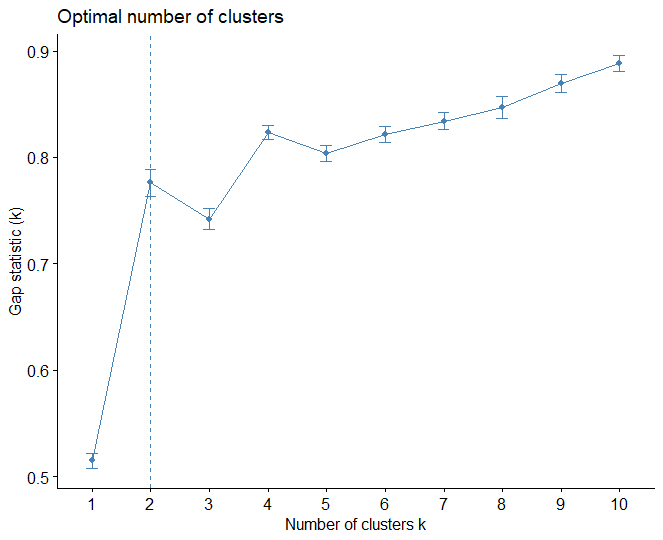
k-means result for with Quartile attribute:



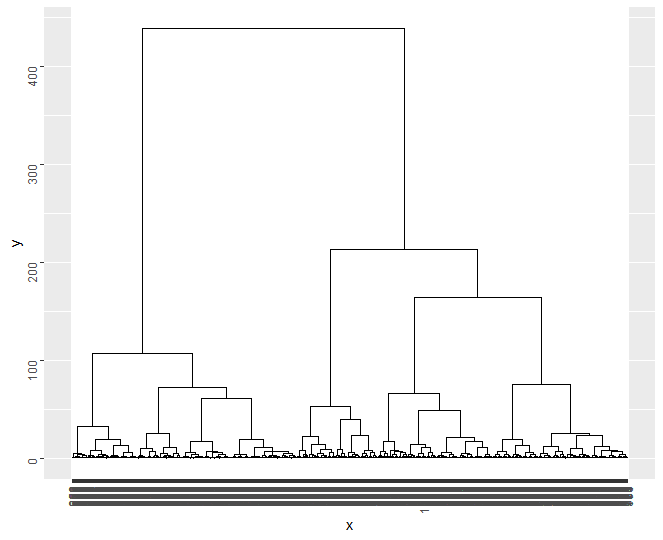
fviz\_nbclust result for without Quartile attribute:



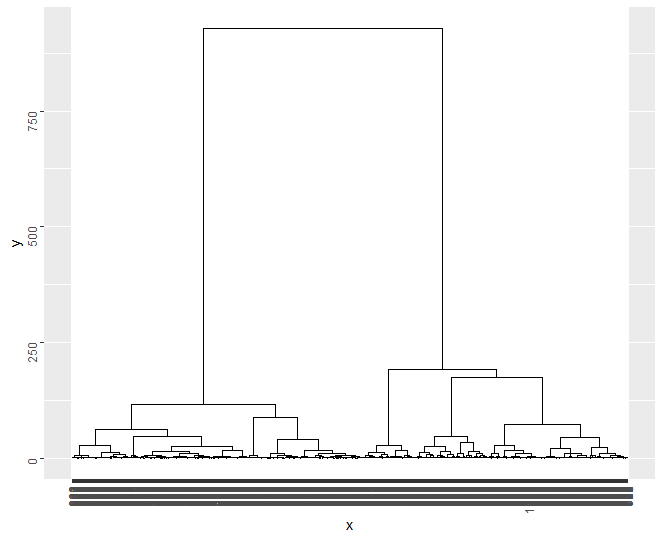
fviz\_nbclust result for with Quartile attribute:



Hclust result for without Quartile attribute:



Hclust result for with Quartile attribute:



1. Conclusion

We made the table to make the comparisons more visible.

K-means cluster for J\_or\_C:

|  |  |  |
| --- | --- | --- |
|  | my\_data\_w \_jc | my\_data\_with\_jc |
| totss | 3996 | 4995 |
| withinss | [1:2] 541 1783 | [1:2] 661 1795 |
| tot.withinss | 2324 | 2456 |
| betweenss | 1672 | 2539 |
| size | [1:2] 525 475 | [1:2] 523 477 |

When we check our results again with the help of the summary () function, we get the following conclusion:



Hclust cluster for J\_or\_C:

|  |  |  |
| --- | --- | --- |
|  | my\_data\_w \_jc | my\_data\_with\_jc |
| size | [1:2] 477 523 | [1:2] 477 523 |
| separation.matrix | [,1] [,2]  [1,] 0.000000 1.077443  [2,] 1.077443 0.000000 | [,1] [,2]  [1,] 0.000000 1.788012  [2,] 1.788012 0.000000 |
| average.between | [,1] [,2]  [1,] 0.000000 3.239491  [2,] 3.239491 0.000000 | [,1] [,2]  [1,] 0.000000 3.806638  [2,] 3.806638 0.000000 |
| n.within | [1] 250029 | [1] 250029 |
| average.distance | [1] 2.492752 1.272840 | [1] 2.492752 1.408755 |

When we check our results again with the help of the summary () function, we get the following conclusion:



K-means cluster for Quartile:

|  |  |  |
| --- | --- | --- |
|  | my\_data\_w \_q | my\_data\_with\_q |
| totss | 3996 | 4995 |
| withinss | [1:5] 358 319 376 261 269 | [1:5] 266 322 225 162 348 |
| tot.withinss | 1582 | 1323 |
| betweenss | 2414 | 3672 |
| size | [1:5] 248 129 339 158 126 | [1:5] 304 242 219 117 118 |

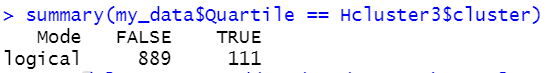
When we check our results again with the help of the summary () function, we get the following conclusion:



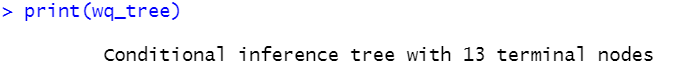
Hclust cluster for Quartile:

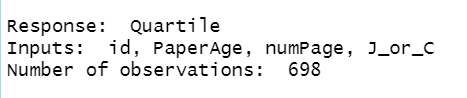
|  |  |  |
| --- | --- | --- |
|  | my\_data\_w \_q | my\_data\_with\_q |
| size | [1] 145 287 118 206 244 | [1] 252 317 206 129 96 |
| separation.matrix | [,1] [,2] [,3] [,4] [,5]  [1,] 0.0000000 0.5266336 0.8606468 0.5922712 0.3950468  [2,] 0.5266336 0.0000000 0.2787948 0.5494582 0.2474866  [3,] 0.8606468 0.2787948 0.0000000 0.8904125 0.3267542  [4,] 0.5922712 0.5494582 0.8904125 0.0000000 0.2084245  [5,] 0.3950468 0.2474866 0.3267542 0.2084245 0.0000000 | [,1] [,2] [,3] [,4] [,5]  [1,] 0.0000000 2.2847442 1.7880116 0.4449239 0.2335205  [2,] 2.2847442 0.0000000 0.1189046 2.4271190 2.2713032  [3,] 1.7880116 0.1189046 0.0000000 2.2029073 2.4195640  [4,] 0.4449239 2.4271190 2.2029073 0.0000000 0.5740274  [5,] 0.2335205 2.2713032 2.4195640 0.5740274 0.0000000 |
| average.between | [,1] [,2] [,3] [,4] [,5]  [1,] 0.000000 3.359271 3.686454 3.566923 2.856741  [2,] 3.359271 0.000000 2.016581 3.271499 2.577186  [3,] 3.686454 2.016581 0.000000 3.300911 2.382442  [4,] 3.566923 3.271499 3.300911 0.000000 2.514870  [5,] 2.856741 2.577186 2.382442 2.514870 0.000000 | [,1] [,2] [,3] [,4] [,5]  [1,] 0.000000 3.211663 3.570514 2.929802 2.817373  [2,] 3.211663 0.000000 1.653474 4.495734 3.696474  [3,] 3.570514 1.653474 0.000000 4.733820 4.328527  [4,] 2.929802 4.495734 4.733820 0.000000 3.634494  [5,] 2.817373 3.696474 4.328527 3.634494 0.000000 |
| n.within | [1] 109145 | [1] 115643 |
| average.distance | [1] 2.303828 1.462966 1.313390 1.986066 1.589929 | [1] 1.548693 1.172738 1.211762 2.297341 1.455452 |

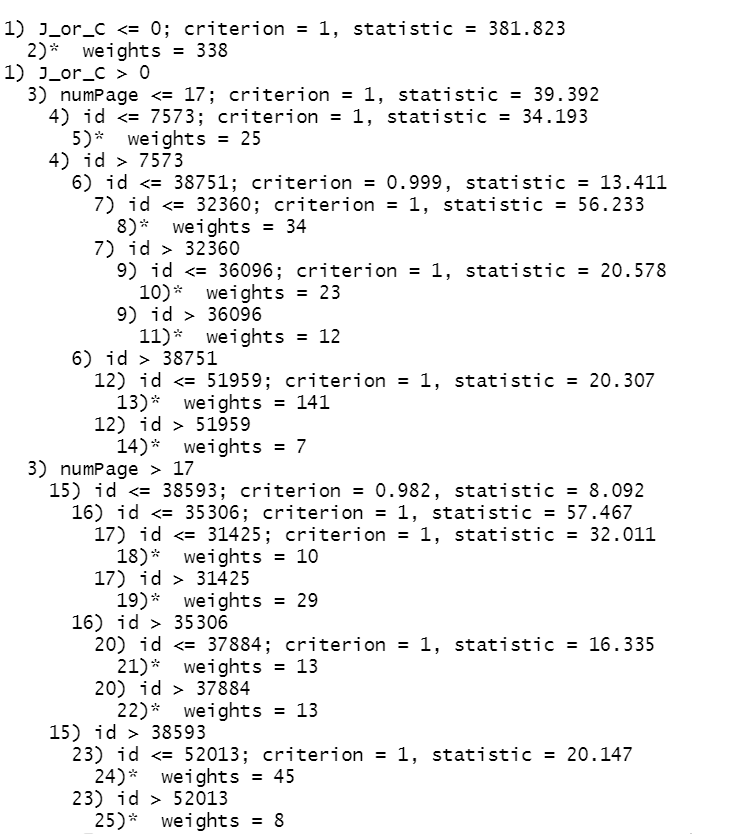
When we check our results again with the help of the summary () function, we get the following conclusion:



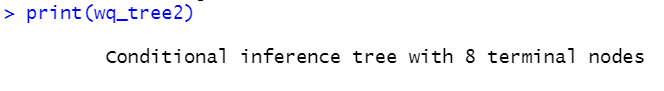
Tree classification results for my\_data\_with\_jc:

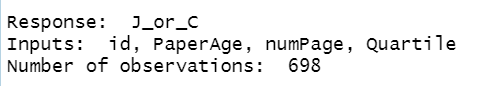


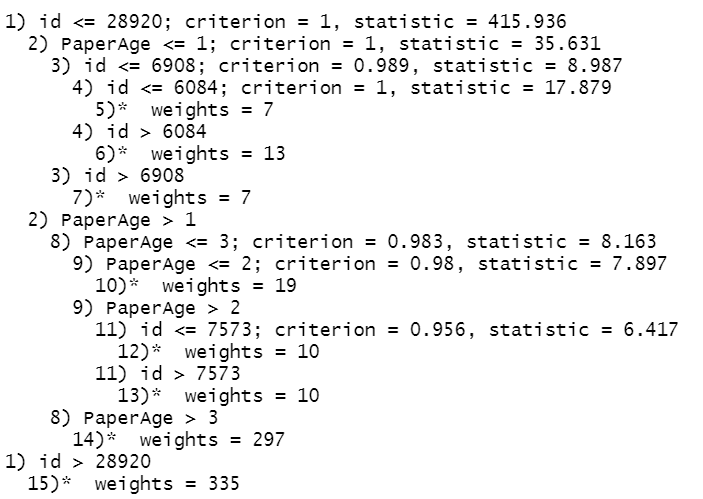




Tree classification results for my\_data\_with\_q:







When the results are compared, it is seen that hclust clustering gives better results. Due to the higher True values in the summary () function output.

Appendix

Correlation result for J\_or\_C:

[,1]

id 0.7695901283

Authors\_Num -0.0536400759

Countries\_Unique\_Num 0.0614238050

FRES\_Title -0.0415767181

FLESCH\_Title 0.0476400330

numCharTitle\_all 0.0568366776

numCharTitle\_onlyAlpha 0.0602109989

numCharTitle\_nonAlpha 0.0426033155

nonAlphaCharTitle\_isExist 0.0122823297

numWordTitle 0.0218441874

numABV\_Title -0.0686486104

binABV\_Title -0.0748833885

numSentAbstract 0.1341003085

numABV\_Abstract -0.0002639689

binABV\_Abstract -0.0748833885

PaperAge 0.2003015217

numLexVerb 0.0146084150

numSylGreaThan2 0.0250839268

numPage 0.4393504852

Year -0.2003015217

Cited by 0.1137294360

FRES\_Abstract -0.0460733295

FLESCH\_Abstract 0.0077471322

isFunding 0.2468480900

keywordsListedAlpha 0.2097446541

numKeywords -0.1670646147

Quartile 0.7408991571

J\_or\_C 1.0000000000

CitationMetric\_1 0.1827314360

CitationMetric\_2 0.1032091493

CitationMetric\_3 0.1838996395

CitationMetric\_4\_CB 0.1347132952

CitationMetric\_4a\_CM2 NA

CitationMetric\_4b\_CM3 0.2047115491

CitationMetric\_5\_CB NA

CitationMetric\_5a\_CM2 NA

CitationMetric\_5b\_CM3 NA

Dominant\_Topic -0.0114268930

single\_quote\_mark 0.0043796993

parenthesis\_mark 0.0587407450

hyphen\_mark 0.0656867158

period\_mark 0.0782819204

slash\_mark 0.0400352103

colon\_mark 0.0899246667

question\_mark 0.1296179903

presenceColon 0.0766965888

avgPunctuation 0.1131239203

numPreposition 0.1191945494

numFreqLexItems\_connectives 0.0566403304

numFreqLexItems\_action\_markers -0.0205420605

numFreqLexItems\_closing -0.0020626981

numTitleSubstantiveWordsWoutStopwords 0.0210552487

numTitleSubstantiveWordsWithStopwords 0.0145262282

numAbstractSubstantiveWordsWoutStopwords 0.1479552017

numAbstractSubstantiveWordsWithStopwords 0.1438372975

presenceInitialPosition\_a 0.0725637833

presenceInitialPosition\_a\_or\_the 0.0717338269

presenceInitialPosition\_ing 0.0124593729

Correlation result for Quartile:

[,1]

id 0.632649508

Authors\_Num -0.041895122

Countries\_Unique\_Num 0.012876833

FRES\_Title -0.056831488

FLESCH\_Title 0.065623774

numCharTitle\_all 0.069849063

numCharTitle\_onlyAlpha 0.072664727

numCharTitle\_nonAlpha 0.040268878

nonAlphaCharTitle\_isExist 0.028163619

numWordTitle 0.038952080

numABV\_Title -0.031661262

binABV\_Title -0.038969138

numSentAbstract 0.157633363

numABV\_Abstract 0.028419542

binABV\_Abstract -0.038969138

PaperAge 0.166901677

numLexVerb 0.032459801

numSylGreaThan2 0.044466404

numPage 0.537468503

Year -0.166901677

Cited by 0.068643861

FRES\_Abstract -0.042544540

FLESCH\_Abstract 0.012725956

isFunding 0.148911190

keywordsListedAlpha 0.205894208

numKeywords -0.150041700

Quartile 1.000000000

J\_or\_C 0.740899157

CitationMetric\_1 0.100630284

CitationMetric\_2 0.037855301

CitationMetric\_3 0.098933609

CitationMetric\_4\_CB 0.070155016

CitationMetric\_4a\_CM2 NA

CitationMetric\_4b\_CM3 0.100676264

CitationMetric\_5\_CB NA

CitationMetric\_5a\_CM2 NA

CitationMetric\_5b\_CM3 NA

Dominant\_Topic -0.020245882

single\_quote\_mark -0.028803605

parenthesis\_mark 0.069004957

hyphen\_mark 0.055982750

period\_mark 0.113892732

slash\_mark 0.057546078

colon\_mark 0.023162837

question\_mark 0.136562048

presenceColon 0.045363532

avgPunctuation 0.119479756

numPreposition 0.166142960

numFreqLexItems\_connectives 0.039462590

numFreqLexItems\_action\_markers -0.031459617

numFreqLexItems\_closing 0.018410147

numTitleSubstantiveWordsWoutStopwords 0.040944063

numTitleSubstantiveWordsWithStopwords 0.038720864

numAbstractSubstantiveWordsWoutStopwords 0.176529513

numAbstractSubstantiveWordsWithStopwords 0.171547042

presenceInitialPosition\_a 0.070655716

presenceInitialPosition\_a\_or\_the 0.071428060

presenceInitialPosition\_ing -0.007684693

>