Clustering

Sudhanshu Kulkarni

# Background

We have been provided with a data set of the Federal Papers which consisted of 85 essays which were written by Alexander Hamilton, James Madison, and John Jay.Out of these 85 essays, 74 essays were identified with their authors, i.e. 51 essays written by Hamilton, 15 by Madison, 3 by Hamilton and Madison, 5 by Jay. The remaining 11 essays did not have an atuhor attributed to them, hence termed as disputed papers.

In the data set, we are provided of function words. The occurance of these function words vary with different authors. So our task is, by using clustering algorithms like the k-Means and HAC, to develop clusters of these function words and find the authors of these disputed papers.

# Solution

## Importing Libraries

First, we begin with importing libraries. We have installed and imported the “dendextend” library to help us build dendrogram using the HAC model. Also, we have imported the “ggplot2” to visualize the k-Means clusters.

library(dendextend)

## Warning: package 'dendextend' was built under R version 3.5.2

##   
## ---------------------  
## Welcome to dendextend version 1.9.0  
## Type citation('dendextend') for how to cite the package.  
##   
## Type browseVignettes(package = 'dendextend') for the package vignette.  
## The github page is: https://github.com/talgalili/dendextend/  
##   
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues  
## Or contact: <tal.galili@gmail.com>  
##   
## To suppress this message use: suppressPackageStartupMessages(library(dendextend))  
## ---------------------

##   
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':  
##   
## cutree

library(ggplot2)

## 

## Loading Data

Then, we load the csv data into a data set and check its structure to know if any data preprocessing is required.

setwd("C:\\Sudhanshu\\SU\\Semester 2\\707\\Assignment 4")  
fedPaperDS <- read.csv("fedPapers85.csv")  
View(fedPaperDS)  
str(fedPaperDS)

## 'data.frame': 85 obs. of 72 variables:  
## $ author : Factor w/ 5 levels "dispt","Hamilton",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ filename: Factor w/ 85 levels "dispt\_fed\_49.txt",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ a : num 0.28 0.177 0.339 0.27 0.303 0.245 0.349 0.414 0.248 0.442 ...  
## $ all : num 0.052 0.063 0.09 0.024 0.054 0.059 0.036 0.083 0.04 0.062 ...  
## $ also : num 0.009 0.013 0.008 0.016 0.027 0.007 0.007 0.009 0.007 0.006 ...  
## $ an : num 0.096 0.038 0.03 0.024 0.034 0.067 0.029 0.018 0.04 0.075 ...  
## $ and : num 0.358 0.393 0.301 0.262 0.404 0.282 0.335 0.478 0.356 0.423 ...  
## $ any : num 0.026 0.063 0.008 0.056 0.04 0.052 0.058 0.046 0.034 0.037 ...  
## $ are : num 0.131 0.051 0.068 0.064 0.128 0.111 0.087 0.11 0.154 0.093 ...  
## $ as : num 0.122 0.139 0.203 0.111 0.148 0.252 0.073 0.074 0.161 0.1 ...  
## $ at : num 0.017 0.114 0.023 0.056 0.013 0.015 0.116 0.037 0.047 0.031 ...  
## $ be : num 0.411 0.393 0.474 0.365 0.344 0.297 0.378 0.331 0.289 0.379 ...  
## $ been : num 0.026 0.165 0.015 0.127 0.047 0.03 0.044 0.046 0.027 0.025 ...  
## $ but : num 0.009 0 0.038 0.032 0.061 0.037 0.007 0.055 0.027 0.037 ...  
## $ by : num 0.14 0.139 0.173 0.167 0.209 0.186 0.102 0.092 0.168 0.174 ...  
## $ can : num 0.035 0 0.023 0.056 0.088 0 0.058 0.037 0.047 0.056 ...  
## $ do : num 0.026 0.013 0 0 0 0 0.015 0.028 0 0 ...  
## $ down : num 0 0 0.008 0 0 0.007 0 0 0 0 ...  
## $ even : num 0.009 0.025 0.015 0.024 0.02 0.007 0.007 0.018 0 0.006 ...  
## $ every : num 0.044 0 0.023 0.04 0.027 0.007 0.087 0.064 0.081 0.05 ...  
## $ for. : num 0.096 0.076 0.098 0.103 0.141 0.067 0.116 0.055 0.127 0.1 ...  
## $ from : num 0.044 0.101 0.053 0.079 0.074 0.096 0.08 0.083 0.074 0.124 ...  
## $ had : num 0.035 0.101 0.008 0.016 0 0.022 0.015 0.009 0.007 0 ...  
## $ has : num 0.017 0.013 0.015 0.024 0.054 0.015 0.036 0.037 0.02 0.019 ...  
## $ have : num 0.044 0.152 0.023 0.143 0.047 0.119 0.044 0.074 0.074 0.044 ...  
## $ her : num 0 0 0 0 0 0 0.007 0 0.034 0.025 ...  
## $ his : num 0.017 0 0 0.024 0.02 0.067 0 0.018 0.02 0.05 ...  
## $ if. : num 0 0.025 0.023 0.04 0.034 0.03 0.029 0 0 0.025 ...  
## $ in. : num 0.262 0.291 0.308 0.238 0.263 0.401 0.189 0.267 0.248 0.274 ...  
## $ into : num 0.009 0.025 0.038 0.008 0.013 0.037 0 0.037 0.013 0.037 ...  
## $ is : num 0.157 0.038 0.15 0.151 0.189 0.26 0.167 0.083 0.208 0.23 ...  
## $ it : num 0.175 0.127 0.173 0.222 0.108 0.156 0.102 0.165 0.134 0.131 ...  
## $ its : num 0.07 0.038 0.03 0.048 0.013 0.015 0 0.046 0.02 0.019 ...  
## $ may : num 0.035 0.038 0.12 0.056 0.047 0.074 0.08 0.092 0.027 0.106 ...  
## $ more : num 0.026 0 0.038 0.056 0.067 0.045 0.08 0.064 0.06 0.081 ...  
## $ must : num 0.026 0.013 0.083 0.071 0.013 0.015 0.044 0.018 0.027 0.068 ...  
## $ my : num 0 0 0 0 0 0 0.007 0 0 0 ...  
## $ no : num 0.035 0 0.03 0.032 0.047 0.059 0.022 0.018 0.02 0.044 ...  
## $ not : num 0.114 0.127 0.068 0.087 0.128 0.134 0.102 0.101 0.094 0.106 ...  
## $ now : num 0 0 0 0 0 0 0.007 0 0.007 0.012 ...  
## $ of : num 0.9 0.747 0.858 0.802 0.869 ...  
## $ on : num 0.14 0.139 0.15 0.143 0.054 0.141 0.051 0.083 0.127 0.118 ...  
## $ one : num 0.026 0.025 0.03 0.032 0.047 0.052 0.073 0.046 0.06 0.031 ...  
## $ only : num 0.035 0 0.023 0.048 0.027 0.022 0.007 0.046 0.02 0.012 ...  
## $ or : num 0.096 0.114 0.06 0.064 0.081 0.074 0.153 0.037 0.154 0.081 ...  
## $ our : num 0.017 0 0 0.016 0.027 0.03 0.051 0 0.007 0.025 ...  
## $ shall : num 0.017 0 0.008 0.016 0 0.015 0.007 0 0.02 0 ...  
## $ should : num 0.017 0.013 0.068 0.032 0 0.03 0.007 0 0 0.012 ...  
## $ so : num 0.035 0.013 0.038 0.04 0.027 0.007 0.051 0.018 0.04 0.05 ...  
## $ some : num 0.009 0.063 0.03 0.024 0.067 0.045 0.007 0.028 0.027 0.025 ...  
## $ such : num 0.026 0 0.045 0.008 0.027 0.015 0.015 0 0.013 0.031 ...  
## $ than : num 0.009 0 0.023 0 0.047 0.03 0.109 0.055 0.067 0.044 ...  
## $ that : num 0.184 0.152 0.188 0.238 0.162 0.208 0.233 0.165 0.208 0.218 ...  
## $ the : num 1.42 1.25 1.49 1.33 1.19 ...  
## $ their : num 0.114 0.165 0.053 0.071 0.027 0.089 0.109 0.083 0.154 0.081 ...  
## $ then : num 0 0 0.015 0.008 0.007 0.007 0.015 0.009 0.007 0.012 ...  
## $ there : num 0.009 0 0.015 0 0.007 0.007 0.036 0.028 0.02 0 ...  
## $ things : num 0.009 0 0 0 0 0 0 0 0 0.012 ...  
## $ this : num 0.044 0.051 0.075 0.103 0.094 0.126 0.08 0.11 0.067 0.093 ...  
## $ to : num 0.507 0.355 0.361 0.532 0.485 0.445 0.56 0.34 0.49 0.498 ...  
## $ up : num 0 0 0 0 0 0 0.007 0 0 0 ...  
## $ upon : num 0 0.013 0 0 0 0 0 0 0 0 ...  
## $ was : num 0.009 0.051 0.008 0.087 0.027 0.007 0.015 0.018 0.027 0 ...  
## $ were : num 0.017 0 0.015 0.079 0.02 0.03 0.029 0.009 0.007 0 ...  
## $ what : num 0 0 0.008 0.008 0.02 0.015 0.015 0.009 0.02 0.025 ...  
## $ when : num 0.009 0 0 0.024 0.007 0.037 0.007 0 0.02 0.012 ...  
## $ which : num 0.175 0.114 0.105 0.167 0.155 0.186 0.211 0.175 0.201 0.199 ...  
## $ who : num 0.044 0.038 0.008 0 0.027 0.045 0.022 0.018 0.04 0.031 ...  
## $ will : num 0.009 0.089 0.173 0.079 0.168 0.111 0.145 0.267 0.154 0.106 ...  
## $ with : num 0.087 0.063 0.045 0.079 0.074 0.089 0.073 0.129 0.027 0.081 ...  
## $ would : num 0.192 0.139 0.068 0.064 0.04 0.037 0.073 0.037 0.04 0.031 ...  
## $ your : num 0 0 0 0 0 0 0 0 0 0 ...

## 

## Data Preprocessing

Since, there are factor fields in the data set we remove the author and filename columns.

fedPapersCleanedDS <- fedPaperDS[,c(3:72)]

## 

## Clustering Results

### k-Means Algorithm

We have used the k-Means algorithm to find the author names of the disputed papers. The initial cluster assignments are done by setting a seed value and dividing the data into 4 clusters and setting nstart = 20. Thus, R will perform 20 random starting assignments and later select the one with the lowest within cluster variation. We can see the cluster centroids, the clusters that each data point was assigned to, and the variation within clusters.

set.seed(100)  
modelKMeans <- kmeans(fedPapersCleanedDS,4, nstart = 20)  
modelKMeans

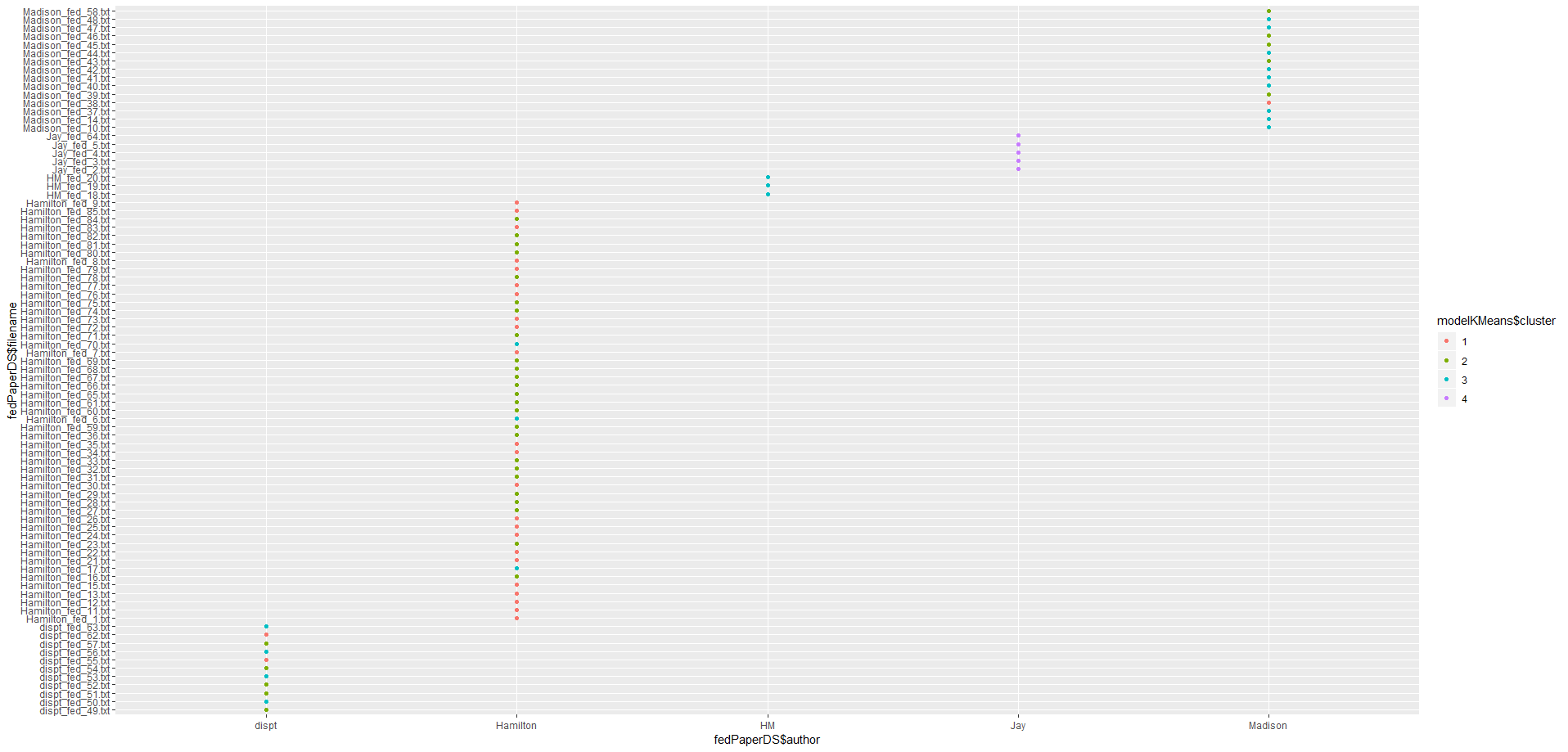
## K-means clustering with 4 clusters of sizes 26, 35, 19, 5  
##   
## Cluster means:  
## a all also an and any  
## 1 0.3452308 0.05169231 0.005076923 0.07503846 0.3461923 0.04500000  
## 2 0.2904000 0.05502857 0.006628571 0.07471429 0.3284571 0.04554286  
## 3 0.2624211 0.05478947 0.009894737 0.05900000 0.4536842 0.03078947  
## 4 0.1598000 0.03600000 0.019800000 0.02520000 0.7152000 0.03760000  
## are as at be been but  
## 1 0.07650000 0.1101154 0.05750000 0.3051538 0.06057692 0.03019231  
## 2 0.07422857 0.1362857 0.03788571 0.3382571 0.05811429 0.03305714  
## 3 0.08094737 0.1126316 0.04010526 0.2341053 0.06994737 0.02942105  
## 4 0.08520000 0.1568000 0.03600000 0.2754000 0.02680000 0.04920000  
## by can do down even every  
## 1 0.1200385 0.03750000 0.007153846 0.0012692308 0.01350000 0.02673077  
## 2 0.1139429 0.03865714 0.006000000 0.0023714286 0.01100000 0.02474286  
## 3 0.1588421 0.02794737 0.005000000 0.0007368421 0.01026316 0.02321053  
## 4 0.1362000 0.03300000 0.008200000 0.0000000000 0.00760000 0.00600000  
## for. from had has have her  
## 1 0.09530769 0.08330769 0.01819231 0.05603846 0.09469231 0.014730769  
## 2 0.09031429 0.07585714 0.01525714 0.03680000 0.09314286 0.001857143  
## 3 0.09742105 0.07921053 0.03736842 0.04668421 0.09984211 0.008736842  
## 4 0.09600000 0.09100000 0.01640000 0.02880000 0.08680000 0.014800000  
## his if. in. into is it  
## 1 0.03911538 0.02880769 0.3257692 0.02288462 0.1606923 0.1417308  
## 2 0.02125714 0.02720000 0.3341714 0.01925714 0.1686571 0.1761143  
## 3 0.03300000 0.01889474 0.2869474 0.02926316 0.1438947 0.1289474  
## 4 0.00900000 0.05260000 0.2714000 0.04460000 0.0936000 0.2048000  
## its may more must my no  
## 1 0.04661538 0.05819231 0.04615385 0.03788462 0.005961538 0.03365385  
## 2 0.05422857 0.06845714 0.03900000 0.03431429 0.001971429 0.03434286  
## 3 0.04389474 0.05584211 0.04621053 0.02721053 0.002315789 0.03152632  
## 4 0.03340000 0.05680000 0.08680000 0.02120000 0.001800000 0.01500000  
## not now of on one only  
## 1 0.09300000 0.007000000 0.9373462 0.05934615 0.04053846 0.01603846  
## 2 0.09514286 0.005485714 0.9473714 0.06574286 0.03645714 0.02582857  
## 3 0.08278947 0.005578947 0.8721579 0.08789474 0.03842105 0.02142105  
## 4 0.10800000 0.006600000 0.6390000 0.07460000 0.08140000 0.04340000  
## or our shall should so some  
## 1 0.09576923 0.03584615 0.02030769 0.02673077 0.02953846 0.01957692  
## 2 0.09382857 0.01057143 0.02005714 0.03057143 0.02762857 0.01540000  
## 3 0.08657895 0.01700000 0.01457895 0.01505263 0.03036842 0.02821053  
## 4 0.16080000 0.06600000 0.01740000 0.04140000 0.04460000 0.02140000  
## such than that the their then  
## 1 0.02915385 0.04984615 0.2198462 1.161423 0.08130769 0.005653846  
## 2 0.02948571 0.04000000 0.2227714 1.434971 0.07508571 0.006085714  
## 3 0.02305263 0.03826316 0.1733158 1.272263 0.09578947 0.006157895  
## 4 0.05120000 0.06280000 0.2434000 0.854400 0.14160000 0.008000000  
## there things this to up upon  
## 1 0.03280769 0.003846154 0.09430769 0.5707308 0.007653846 0.044576923  
## 2 0.03042857 0.002742857 0.09040000 0.5723714 0.001342857 0.033571429  
## 3 0.01336842 0.001210526 0.07968421 0.4343684 0.002631579 0.007421053  
## 4 0.01400000 0.001400000 0.05320000 0.4834000 0.000000000 0.001800000  
## was were what when which who  
## 1 0.01753846 0.01800000 0.01200000 0.013692308 0.1598077 0.03211538  
## 2 0.02254286 0.01702857 0.01382857 0.011600000 0.1608286 0.02985714  
## 3 0.04352632 0.02689474 0.01078947 0.006894737 0.1648421 0.03300000  
## 4 0.02480000 0.02880000 0.01840000 0.021000000 0.0986000 0.05160000  
## will with would your  
## 1 0.08838462 0.07946154 0.13323077 0.004076923  
## 2 0.11137143 0.07688571 0.10220000 0.000000000  
## 3 0.08205263 0.08110526 0.05152632 0.001789474  
## 4 0.12600000 0.09500000 0.12520000 0.006400000  
##   
## Clustering vector:  
## [1] 2 3 2 2 3 2 1 3 2 1 3 1 1 1 1 1 2 3 1 1 2 1 1 1 2 2 2 1 2 2 2 1 1 2 2  
## [36] 3 2 2 2 2 2 2 2 1 3 2 1 1 2 2 1 1 2 1 1 2 2 2 1 2 1 1 3 3 3 4 4 4 4 4  
## [71] 3 3 3 1 2 3 3 3 2 3 2 2 3 3 2  
##   
## Within cluster sum of squares by cluster:  
## [1] 2.173745 3.474834 1.803998 0.598822  
## (between\_SS / total\_SS = 36.0 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss"   
## [5] "tot.withinss" "betweenss" "size" "iter"   
## [9] "ifault"

table(modelKMeans$cluster, fedPaperDS$author)

##   
## dispt Hamilton HM Jay Madison  
## 1 2 23 0 0 1  
## 2 5 25 0 0 5  
## 3 4 3 3 0 9  
## 4 0 0 0 5 0

We can also plot the data of the clusters as shown below:

modelKMeans$cluster <- as.factor(modelKMeans$cluster)  
ggplot(fedPapersCleanedDS, aes(fedPaperDS$author, fedPaperDS$filename, color = modelKMeans$cluster)) + geom\_point()

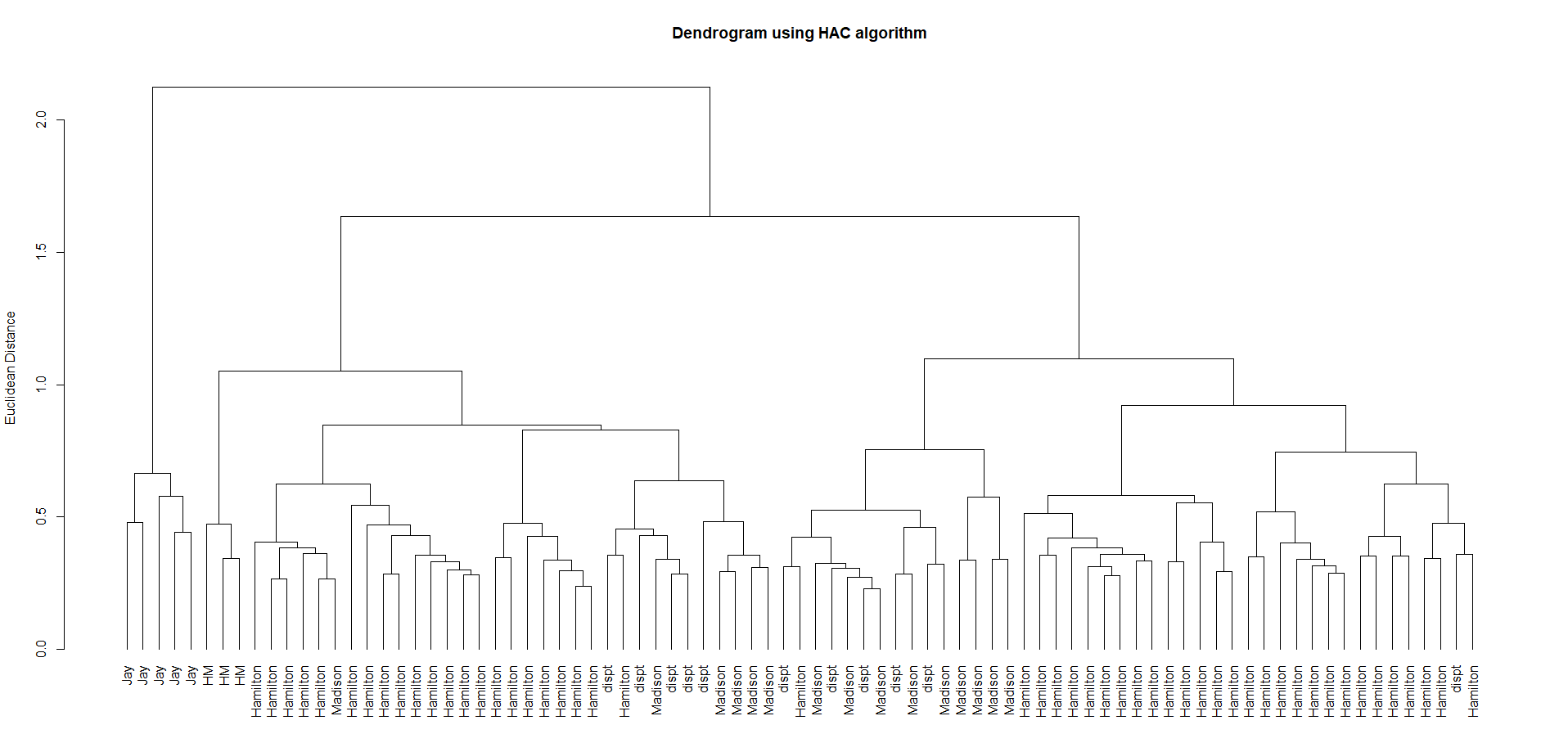


From this visualization we can say that two of the disputed essays (orange dots) were written by Hamilton and the remaining nine were written by Madison.

### HAC Algorithm

The HAC algorithm assists us to identify the clusters with the help of a dendrogram. Below is the how we plot the dendrogram:

distance = dist(as.matrix(fedPapersCleanedDS))  
hacModel <- as.dendrogram(hclust(distance, method = "ward.D2"))  
labels(hacModel) <- fedPaperDS[,1][order.dendrogram(hacModel)]  
plot(hacModel, main="Dendrogram using HAC algorithm",xlab = "", ylab = "Euclidean Distance")



From this dendrogram we can conclude that, two of the disputed essays (dendrogram legs shared with Hamilton) were written by Hamilton and the remaining nine were written by Madison.

## 

## Conclusion

Hence, we can conclude that two disputed essays were written by Hamilton and remaining nine essays were written by Madison. Also, from the k-Means cluster plot and the dendrogram we can conclude that the essays written by both Hamiton and Madison are clustered along with Hamilton’s essays.