Victoria_Haley_HW5

Victoria Haley

2023-05-05

HW 5: Using Supervised and Unsupervised Learning Techniques to Solve a Mystery in History

Introduction/Context

In this assignment, I used clustering methods to solve a mystery in history: who wrote the disputed essays, Hamilton or Madison? The Federalist Papers were a series of eighty-five essays urging the citizens of New York to ratify the new United States Constitution and were originally published anonymously in New York newspapers in 1787 and 1788 under the pen name "Publius." However, the authors of each essay were not identified until the 1818 edition published by Jacob Gideon. Among the essays, 11 were written by "Hamilton or Madison," and the authorship remains disputed to this day.

To solve this mystery, I used the Federalist Paper data set and applied clustering algorithms k-Means, and Hierarchical Clustering (HAC). In addition to the clustering methods, I have also included decision tree modeling to solve the mystery of who wrote the disputed Federalist Papers. Using the same Federalist Paper data set, I applied decision tree algorithms to analyze the distribution of function words and their importance in determining the authorship of the papers.

The results of this analysis were compared with the clustering results to provide a more comprehensive understanding of the disputed papers' authorship. Together, these methods provide compelling evidence that James Madison wrote most of the disputed papers. Specifically, papers # 50, 53, 56, 57, and 62 were likely written by Madison.

Loading and quickly viewing the data

```
library(readr)
setwd("/Users/victoriahaley/")
fedPapers <- read_csv("Desktop/IST 707/fedPapers85.csv")
```

```
## Rows: 85 Columns: 72
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): author, filename
## dbl (70): a, all, also, an, and, any, are, as, at, be, been, but, by, can, d...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(fedPapers)
```

```
## # A tibble: 6 x 72
##   author filename        a   all  also    an   and   any   are    as    at    be
##   <chr>  <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 dispt  dispt_fed_~ 0.28  0.052 0.009 0.096 0.358 0.026 0.131 0.122 0.017 0.411
```

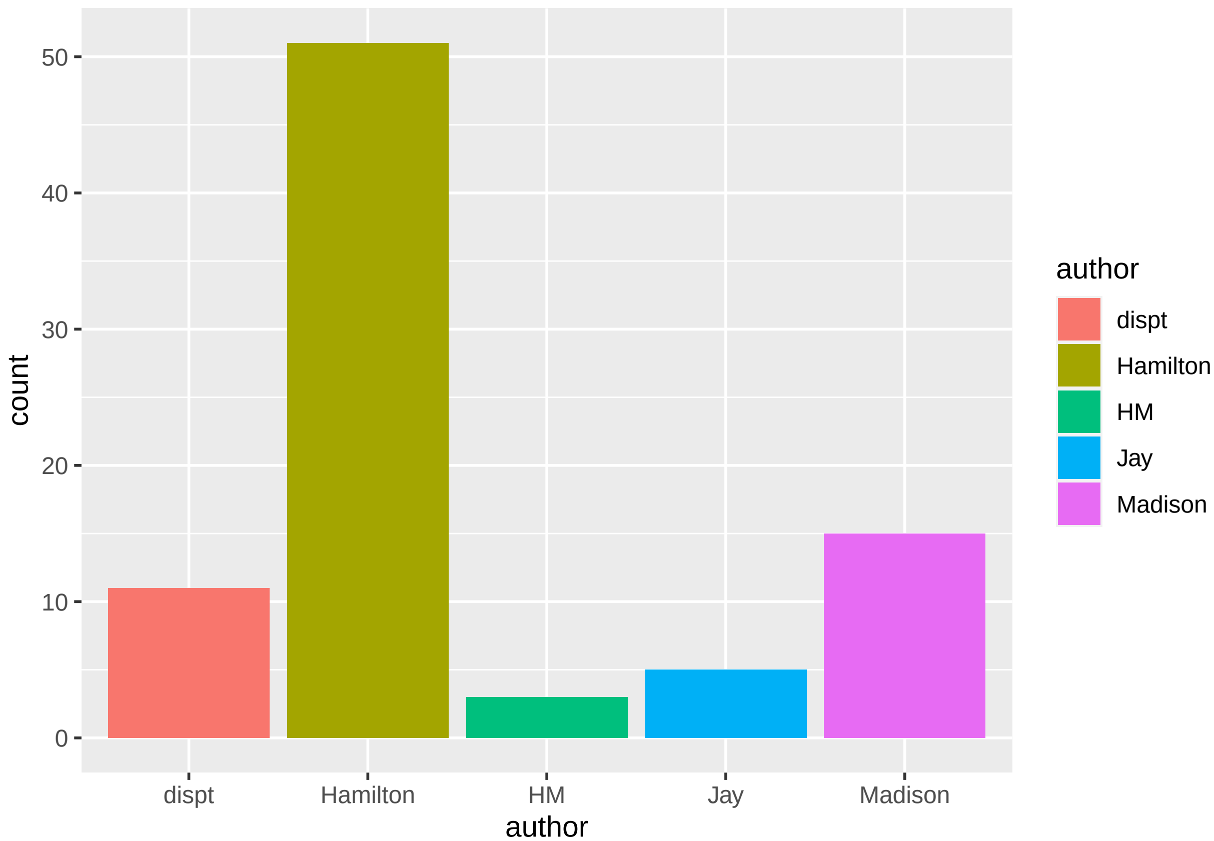
```
## 2 dispt  dispt_fed_~ 0.177 0.063 0.013 0.038 0.393 0.063 0.051 0.139 0.114 0.393
## 3 dispt  dispt_fed_~ 0.339 0.09  0.008 0.03  0.301 0.008 0.068 0.203 0.023 0.474
## 4 dispt  dispt_fed_~ 0.27  0.024 0.016 0.024 0.262 0.056 0.064 0.111 0.056 0.365
## 5 dispt  dispt_fed_~ 0.303 0.054 0.027 0.034 0.404 0.04  0.128 0.148 0.013 0.344
## 6 dispt  dispt_fed_~ 0.245 0.059 0.007 0.067 0.282 0.052 0.111 0.252 0.015 0.297
## # ... with 60 more variables: been <dbl>, but <dbl>, by <dbl>, can <dbl>,
## #   do <dbl>, down <dbl>, even <dbl>, every <dbl>, `for` <dbl>, from <dbl>,
## #   had <dbl>, has <dbl>, have <dbl>, her <dbl>, his <dbl>, `if` <dbl>,
## #   `in` <dbl>, into <dbl>, is <dbl>, it <dbl>, its <dbl>, may <dbl>,
## #   more <dbl>, must <dbl>, my <dbl>, no <dbl>, not <dbl>, now <dbl>, of <dbl>,
## #   on <dbl>, one <dbl>, only <dbl>, or <dbl>, our <dbl>, shall <dbl>,
## #   should <dbl>, so <dbl>, some <dbl>, such <dbl>, than <dbl>, that <dbl>, ...
```

```
#Quick look at the first 6 rows of each column
sum(is.na(fedPapers))
```

```
## [1] 0
```

```
#No missing values
```

```
library(ggplot2)
ggplot(data = fedPapers) + geom_bar(aes(x=author, fill=author))
```




```
#This barplot shows how much of a difference there is between authors, and how Hamilton left everybody
```

Unsupervised Learning

k-Means Clustering

```r
#First, we remove the author names for clustering purposes
fedPapers_km <- fedPapers[,2:72]
#Then, the row names are updated to match the file names so that the dataframe can be numerical
fedPapers_km <- fedPapers_km[,c(2:71)]
row.names(fedPapers_km)<- fedPapers$filename
```

Data prep

```
## Warning: Setting row names on a tibble is deprecated.
```

```r
#Set seed for fixed random seed
set.seed(35)

#run k-means
clusters <- kmeans(fedPapers_km,9)
fedPapers_km$clusters <- as.factor(clusters$cluster)
str(clusters)
```

```
## List of 9
##  $ cluster     : Named int [1:85] 6 1 6 1 1 6 3 7 6 3 ...
##   ..- attr(*, "names")= chr [1:85] "dispt_fed_49.txt" "dispt_fed_50.txt" "dispt_fed_51.txt" "dispt_f
##  $ centers     : num [1:9, 1:70] 0.276 0.16 0.328 0.213 0.261 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:70] "a" "all" "also" "an" ...
##  $ totss       : num 12.6
##  $ withinss    : num [1:9] 0.685 0.599 0.897 0.171 1.148 ...
##  $ tot.withinss: num 6.19
##  $ betweenss   : num 6.38
##  $ size        : int [1:9] 12 5 12 3 15 11 10 6 11
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

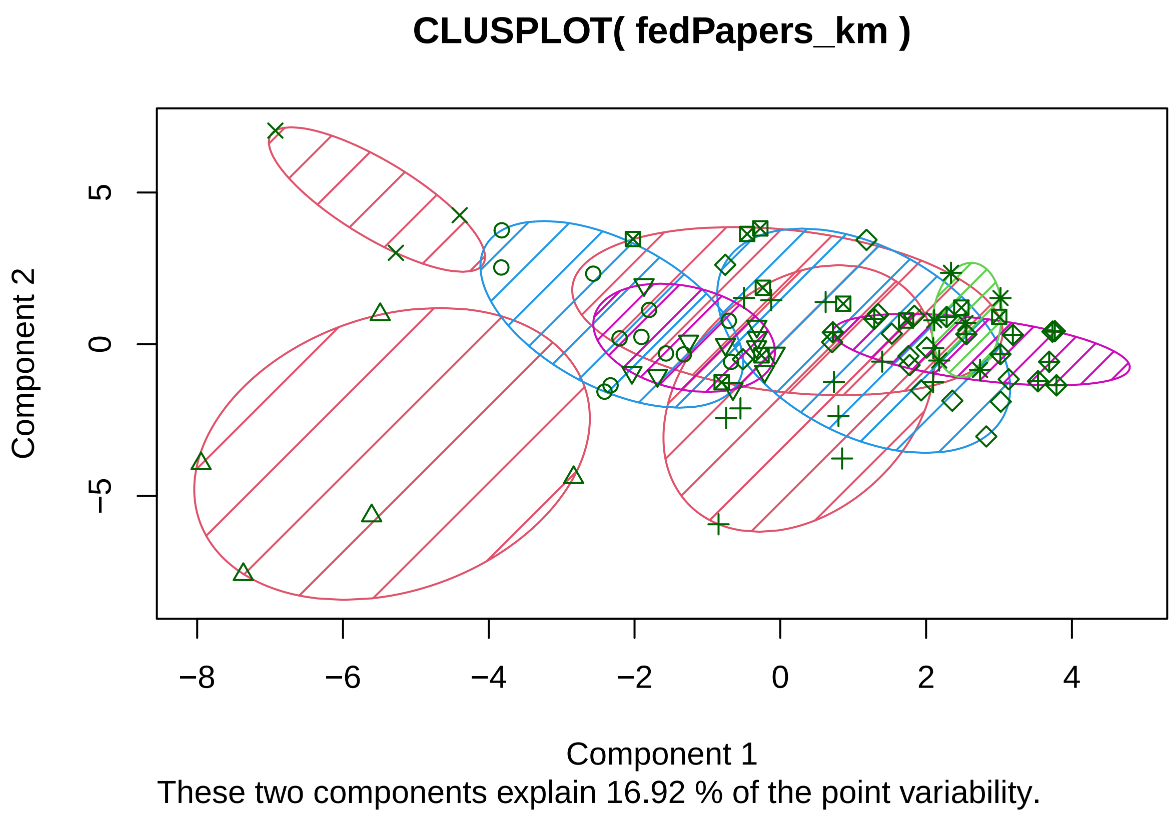
```r
clusters$centers
```

```
##           a        all        also         an       and        any        are
## 1 0.2757500 0.04683333 0.011166667 0.06108333 0.4164167 0.03300000 0.07300000
## 2 0.1598000 0.03600000 0.019800000 0.02520000 0.7152000 0.03760000 0.08520000
## 3 0.3278333 0.04391667 0.004166667 0.06250000 0.3521667 0.04983333 0.08083333
## 4 0.2133333 0.04266667 0.006000000 0.04700000 0.5306667 0.01800000 0.08233333
## 5 0.2614667 0.06273333 0.003133333 0.08093333 0.3469333 0.04386667 0.07686667
## 6 0.2614545 0.06345455 0.012181818 0.05181818 0.3616364 0.03400000 0.09200000
## 7 0.3506000 0.06000000 0.004000000 0.07330000 0.3839000 0.02900000 0.08350000
## 8 0.3646667 0.04683333 0.014166667 0.08566667 0.3055000 0.03816667 0.04716667
## 9 0.3410000 0.05218182 0.004000000 0.09381818 0.3134545 0.06818182 0.06809091
##          as         at         be       been        but         by         can
## 1 0.1320833 0.03808333 0.29458333 0.08858333 0.03500000 0.17841667 0.036416667
## 2 0.1568000 0.03600000 0.27540000 0.02680000 0.04920000 0.13620000 0.033000000
## 3 0.1103333 0.05458333 0.36141667 0.04600000 0.02475000 0.10633333 0.044250000
## 4 0.0800000 0.04433333 0.06666667 0.02133333 0.02833333 0.17500000 0.005333333
## 5 0.1269333 0.04366667 0.33326667 0.05646667 0.02913333 0.09206667 0.051200000
## 6 0.1542727 0.02900000 0.33372727 0.04890909 0.03263636 0.15990909 0.027727273
## 7 0.1111000 0.06400000 0.24370000 0.06890000 0.03230000 0.11590000 0.028500000
## 8 0.1223333 0.04066667 0.28866667 0.06150000 0.01750000 0.08650000 0.036666667
```

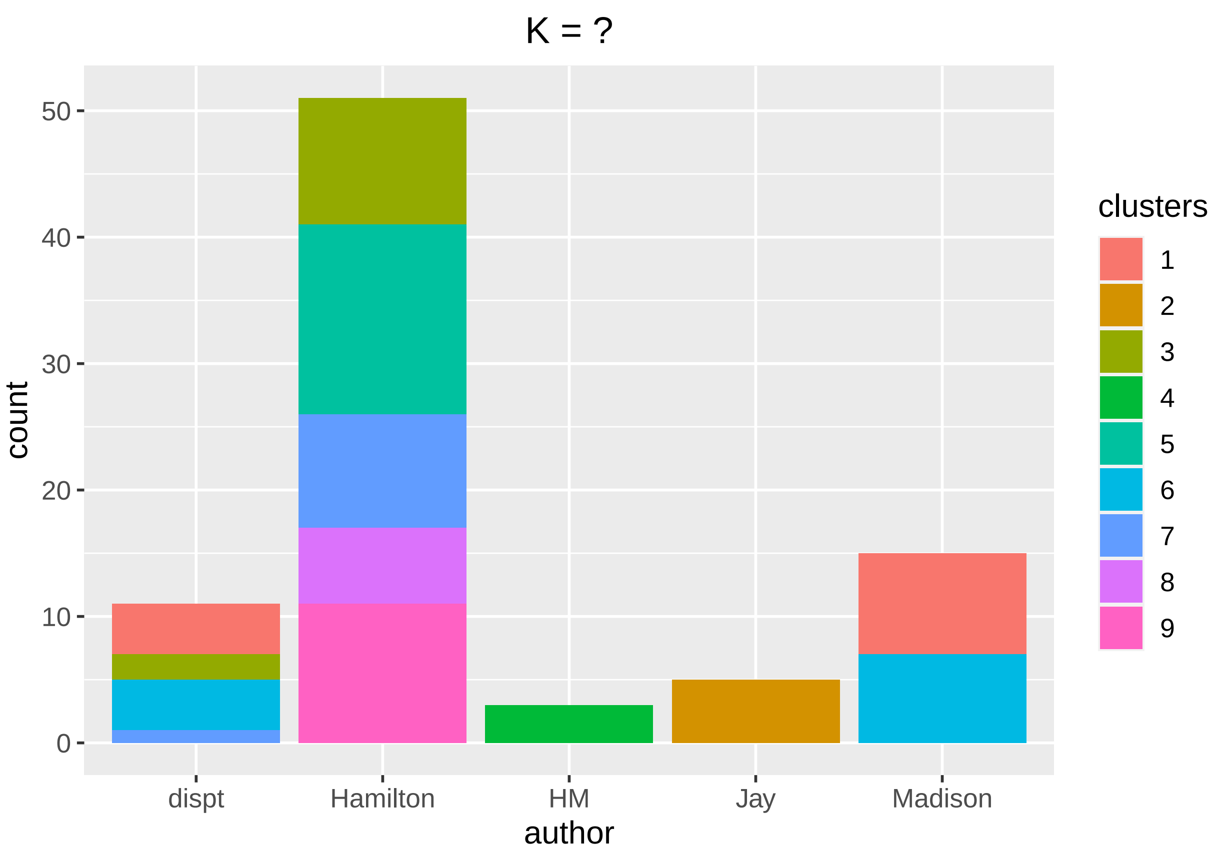
```
## 9 0.1070909 0.04363636 0.30090909 0.07418182 0.04318182 0.12427273 0.027000000
##            do         down        even       every         for       from
## 1 0.003416667 0.0003333333 0.013666667 0.024250000 0.10858333 0.07791667
## 2 0.008200000 0.0000000000 0.007600000 0.006000000 0.09600000 0.09100000
## 3 0.007500000 0.0023333333 0.014833333 0.033166667 0.10716667 0.08433333
## 4 0.002333333 0.0000000000 0.002333333 0.007666667 0.08566667 0.10866667
## 5 0.005733333 0.0045333333 0.014466667 0.015800000 0.08833333 0.08160000
## 6 0.005727273 0.0022727273 0.009272727 0.038727273 0.08581818 0.06245455
## 7 0.008200000 0.0005000000 0.013300000 0.024100000 0.07280000 0.09130000
## 8 0.006666667 0.0000000000 0.005833333 0.014500000 0.11633333 0.07183333
## 9 0.007454545 0.0000000000 0.008636364 0.027181818 0.08627273 0.07254545
##          had        has       have         her        his          if        in
## 1 0.03283333 0.04458333 0.10658333 0.010250000 0.02041667 0.023833333 0.2816667
## 2 0.01640000 0.02880000 0.08680000 0.014800000 0.00900000 0.052600000 0.2714000
## 3 0.01541667 0.04683333 0.07683333 0.018166667 0.04616667 0.027166667 0.3055000
## 4 0.08133333 0.04600000 0.06666667 0.020333333 0.08700000 0.007666667 0.2490000
## 5 0.01473333 0.03773333 0.09440000 0.001800000 0.03180000 0.028533333 0.3135333
## 6 0.01509091 0.03754545 0.08472727 0.003454545 0.01690909 0.019727273 0.2929091
## 7 0.01940000 0.06000000 0.11330000 0.013400000 0.01370000 0.025100000 0.3094000
## 8 0.02050000 0.04600000 0.09750000 0.000000000 0.03550000 0.031000000 0.3881667
## 9 0.01727273 0.04927273 0.10472727 0.001181818 0.02863636 0.031181818 0.4070000
##         into        is        it        its        may       more       must
## 1 0.02658333 0.1655833 0.1575833 0.04391667 0.06741667 0.03941667 0.03775000
## 2 0.04460000 0.0936000 0.2048000 0.03340000 0.05680000 0.08680000 0.02120000
## 3 0.02341667 0.1520833 0.1354167 0.04908333 0.06708333 0.05250000 0.04175000
## 4 0.02966667 0.1040000 0.1086667 0.05400000 0.01733333 0.04200000 0.01366667
## 5 0.01686667 0.1530000 0.1608667 0.06206667 0.06313333 0.03800000 0.03533333
## 6 0.02327273 0.1732727 0.1450909 0.04636364 0.05990909 0.05281818 0.03272727
## 7 0.02760000 0.1257000 0.1136000 0.04340000 0.04950000 0.04210000 0.03020000
## 8 0.01883333 0.1661667 0.1770000 0.03650000 0.06000000 0.04200000 0.02183333
## 9 0.02163636 0.2032727 0.2045455 0.05200000 0.07663636 0.03545455 0.03500000
##            my         no        not         now        of         on        one
## 1 0.001166667 0.03441667 0.10350000 0.004250000 0.8265833 0.09733333 0.03858333
## 2 0.001800000 0.01500000 0.10800000 0.006600000 0.6390000 0.07460000 0.08140000
## 3 0.005833333 0.03375000 0.09600000 0.004583333 0.8840000 0.06350000 0.04641667
## 4 0.005333333 0.02400000 0.03466667 0.012666667 0.8386667 0.09366667 0.04400000
## 5 0.001800000 0.02860000 0.08580000 0.003933333 0.9318667 0.03086667 0.03266667
## 6 0.001272727 0.04372727 0.09645455 0.003454545 0.9105455 0.12672727 0.04500000
## 7 0.002500000 0.02400000 0.08790000 0.007800000 1.0272000 0.05690000 0.03130000
## 8 0.005500000 0.02550000 0.07816667 0.006333333 1.1218333 0.03533333 0.03733333
## 9 0.006272727 0.04390909 0.10245455 0.011181818 0.9145455 0.06045455 0.03509091
##         only         or         our       shall      should         so
## 1 0.02641667 0.08033333 0.023250000 0.013000000 0.022166667 0.03491667
## 2 0.04340000 0.16080000 0.066000000 0.017400000 0.041400000 0.04460000
## 3 0.01416667 0.09858333 0.032250000 0.027166667 0.023500000 0.02608333
## 4 0.01500000 0.05366667 0.006000000 0.004666667 0.002333333 0.01733333
## 5 0.02133333 0.10646667 0.013666667 0.018266667 0.034933333 0.03193333
## 6 0.02409091 0.09309091 0.008727273 0.020272727 0.023636364 0.02227273
## 7 0.01810000 0.09680000 0.047500000 0.011700000 0.024000000 0.02910000
## 8 0.02833333 0.09033333 0.005500000 0.023500000 0.029333333 0.03150000
## 9 0.02363636 0.08909091 0.012000000 0.023272727 0.026909091 0.02945455
##         some       such       than      that      the      their        then
## 1 0.03583333 0.02666667 0.03250000 0.2086667 1.241667 0.09908333 0.006416667
## 2 0.02140000 0.05120000 0.06280000 0.2434000 0.854400 0.14160000 0.008000000
```

```
## 3 0.01841667 0.03000000 0.06383333 0.2369167 1.159583 0.08991667 0.007166667
## 4 0.02133333 0.03100000 0.03766667 0.1016667 1.267000 0.13233333 0.009666667
## 5 0.01440000 0.02973333 0.03693333 0.2070000 1.413267 0.08046667 0.005066667
## 6 0.01936364 0.02663636 0.04000000 0.1910000 1.507455 0.08127273 0.006363636
## 7 0.01840000 0.02110000 0.03410000 0.1691000 1.123700 0.08210000 0.003200000
## 8 0.01633333 0.02466667 0.06016667 0.2466667 1.438000 0.04966667 0.004166667
## 9 0.01436364 0.03245455 0.04163636 0.2524545 1.302909 0.06154545 0.007454545
##         there      things       this        to           up        upon
## 1 0.008500000 0.000750000 0.08341667 0.4692500 0.0019166667 0.003833333
## 2 0.014000000 0.001400000 0.05320000 0.4834000 0.0000000000 0.001800000
## 3 0.033083333 0.002833333 0.08608333 0.6158333 0.0097500000 0.041333333
## 4 0.004333333 0.000000000 0.06500000 0.3826667 0.0090000000 0.005333333
## 5 0.036600000 0.003733333 0.08006667 0.6593333 0.0016000000 0.046733333
## 6 0.009181818 0.002363636 0.08027273 0.4366364 0.0005454545 0.000000000
## 7 0.032900000 0.006000000 0.09700000 0.4821000 0.0036000000 0.036500000
## 8 0.034833333 0.001666667 0.11566667 0.5585000 0.0013333333 0.052333333
## 9 0.042909091 0.002181818 0.10481818 0.5537273 0.0050000000 0.048818182
##          was       were        what        when     which        who       will
## 1 0.03458333 0.02875000 0.010500000 0.007750000 0.1725833 0.02883333 0.09141667
## 2 0.02480000 0.02880000 0.018400000 0.021000000 0.0986000 0.05160000 0.12600000
## 3 0.01541667 0.01016667 0.010583333 0.018666667 0.1555833 0.03858333 0.11191667
## 4 0.11400000 0.04733333 0.002333333 0.015666667 0.1373333 0.04733333 0.01666667
## 5 0.02306667 0.01593333 0.013600000 0.006533333 0.1774000 0.03646667 0.11073333
## 6 0.02100000 0.01454545 0.011272727 0.009909091 0.1511818 0.02445455 0.12527273
## 7 0.02040000 0.02550000 0.010700000 0.005200000 0.1696000 0.02020000 0.07930000
## 8 0.02283333 0.01316667 0.019333333 0.028833333 0.1203333 0.02583333 0.06333333
## 9 0.01927273 0.02118182 0.017272727 0.008818182 0.1658182 0.03481818 0.09572727
##         with      would        your
## 1 0.07025000 0.05183333 0.002833333
## 2 0.09500000 0.12520000 0.006400000
## 3 0.07225000 0.12533333 0.006166667
## 4 0.09700000 0.02566667 0.000000000
## 5 0.08500000 0.09160000 0.000000000
## 6 0.07763636 0.08372727 0.000000000
## 7 0.10050000 0.12930000 0.001000000
## 8 0.08516667 0.15650000 0.000000000
## 9 0.05927273 0.11727273 0.002000000
```

```
#Add clusters to original dataframe with author names
fedPapers_km2 <- fedPapers
fedPapers_km2$clusters <- as.factor(clusters$cluster)
#Plotted results
library(cluster)
clusplot(fedPapers_km, fedPapers_km$clusters, color=TRUE, shade=TRUE, labels=0, lines=0)
```




```
ggplot(data=fedPapers_km2, aes(x=author, fill=clusters)) +
  geom_bar(stat="count") +
  labs(title = "K = ?") +
  theme(plot.title = element_text(hjust=0.5), text=element_text(size=12))
```



With 9 clusters, it looks as though the disputed papers were written neither by Jay nor HM and that about half of the disputed papers were written by Hamilton and the other half by Madison.

Hierarchical Clustering (HAC)

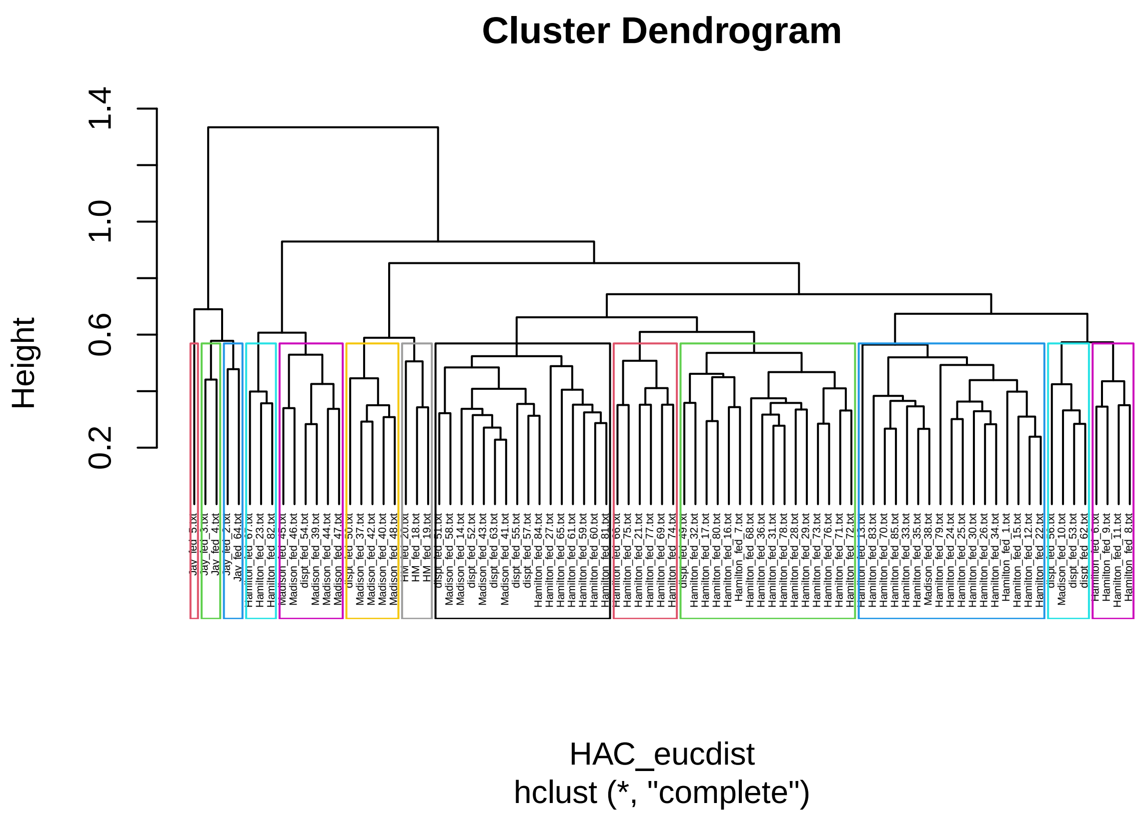
```
#Same data prep steps above, except now for HAC
#First, we remove the author names for clustering purposes
fedPapers_HAC <- fedPapers[,2:72]
#Then, the row names are updated to match the file names so that the dataframe can be numerical
fedPapers_HAC <- fedPapers_HAC[,c(2:71)]
row.names(fedPapers_HAC)<- fedPapers$filename
```

```
## Warning: Setting row names on a tibble is deprecated.
```

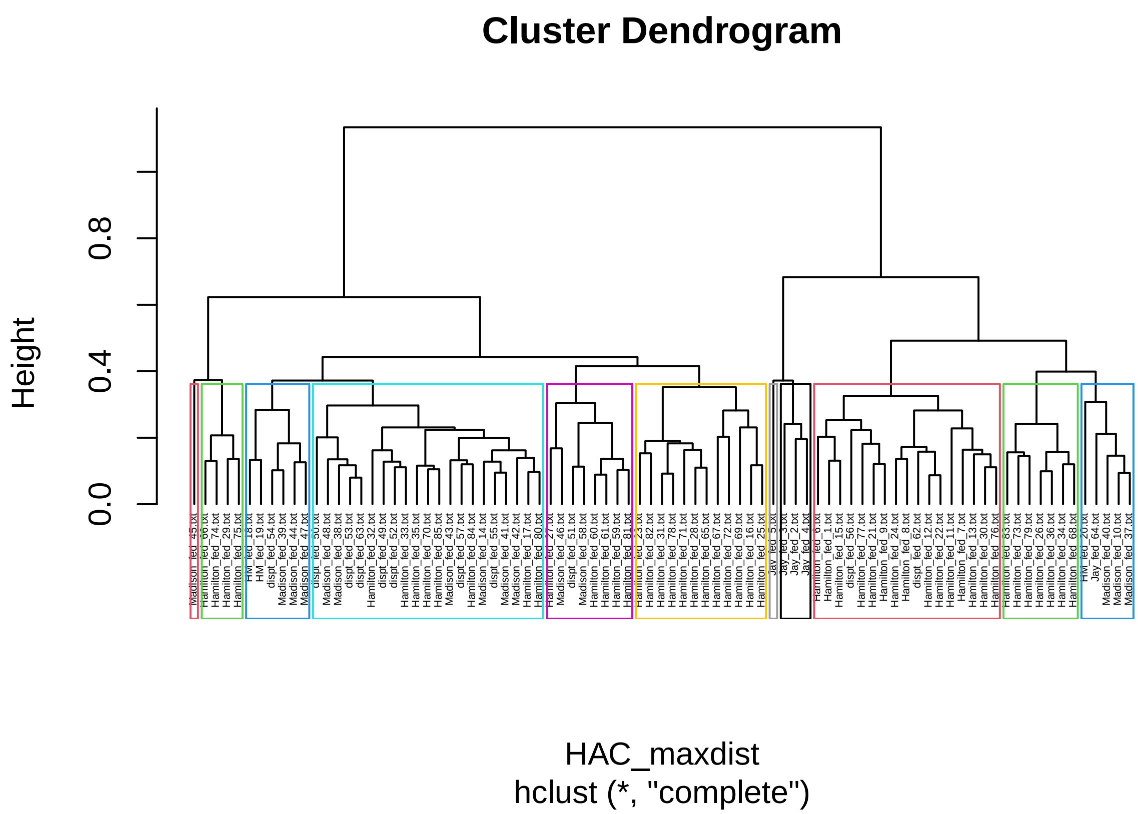
```
#distance calculations
HAC_eucdist <- dist(fedPapers_HAC, method = "euclidean")
HAC_maxdist <- dist(fedPapers_HAC, method = "maximum")
HAC_mandist <- dist(fedPapers_HAC, method = "manhattan")
HAC_candist <- dist(fedPapers_HAC, method = "canberra")
HAC_bindist <- dist(fedPapers_HAC, method = "binary")
HAC_minkdist <- dist(fedPapers_HAC, method = "minkowski")
```

Results of the HAC clusters

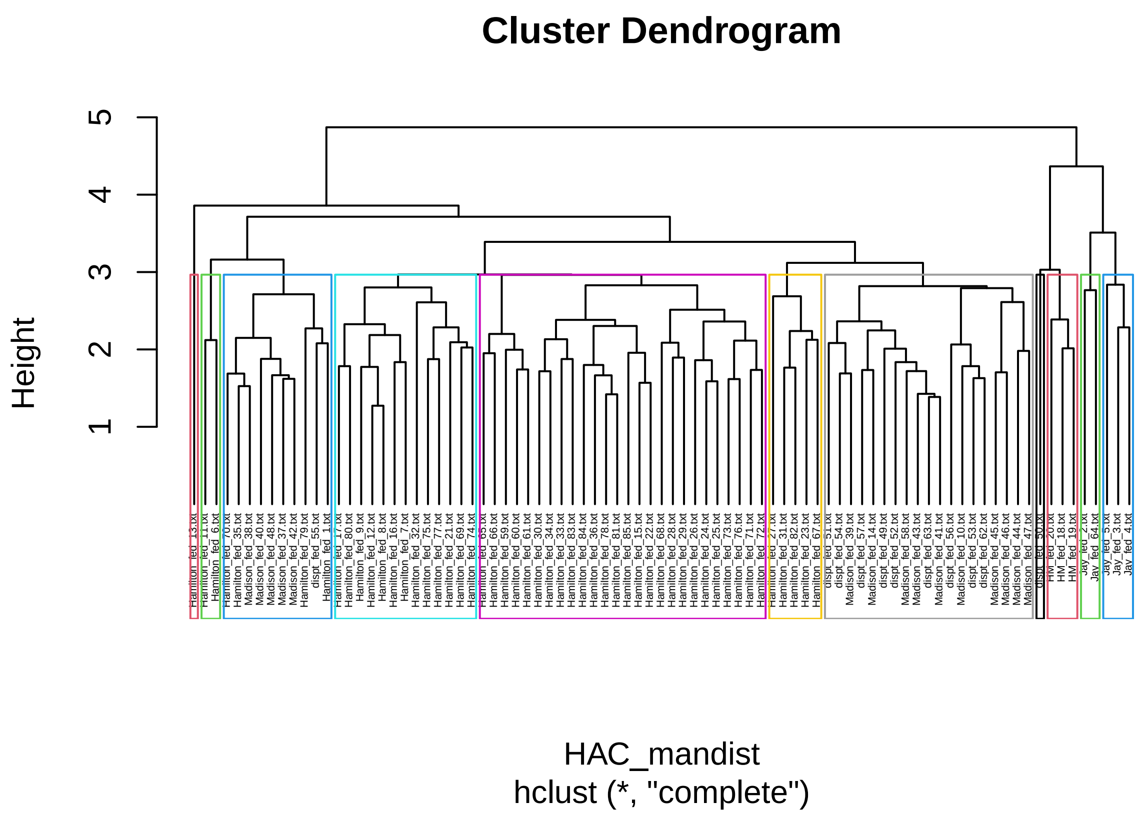
```
#euclidean
HAC <- hclust(HAC_eucdist, method = "complete")
plot(HAC, cex=0.3, hang=-1)
rect.hclust(HAC, k = 13, border=2:15)
```



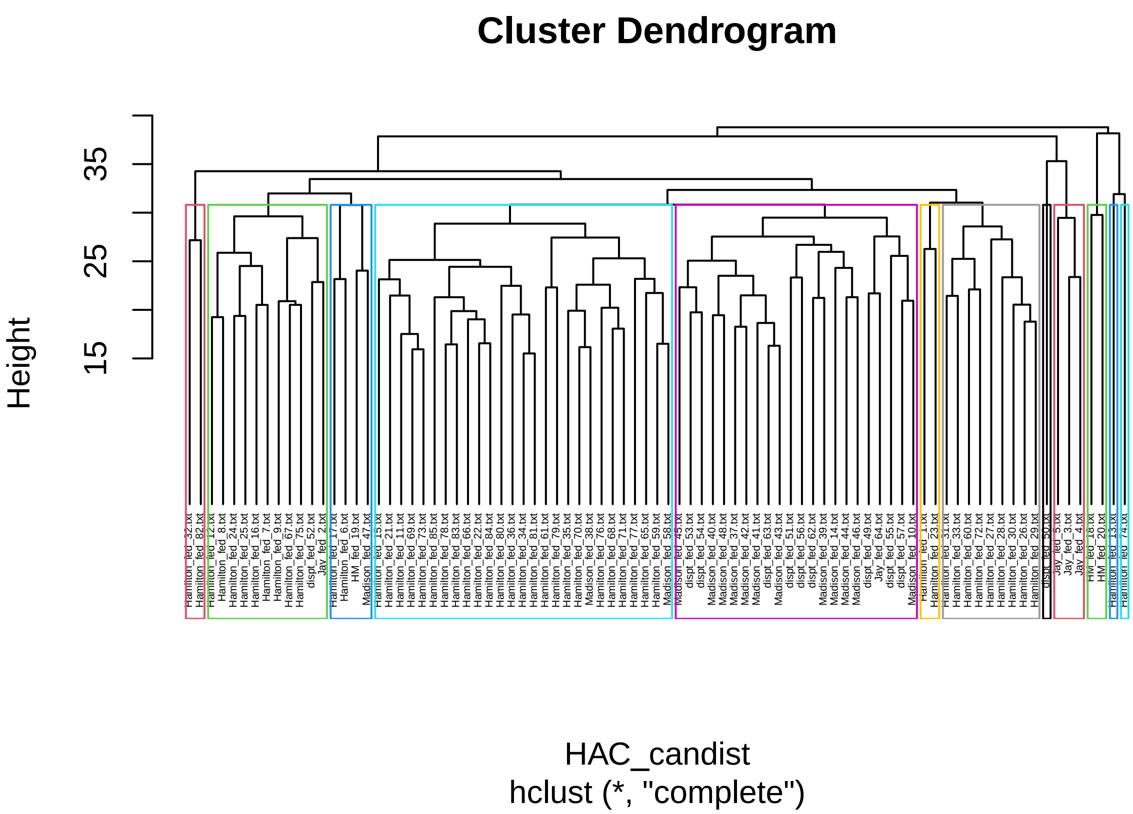

```r
#maximum
HAC2 <- hclust(HAC_maxdist, method = "complete")
plot(HAC2, cex=0.3, hang=-1)
rect.hclust(HAC2, k = 11, border=2:15)
```



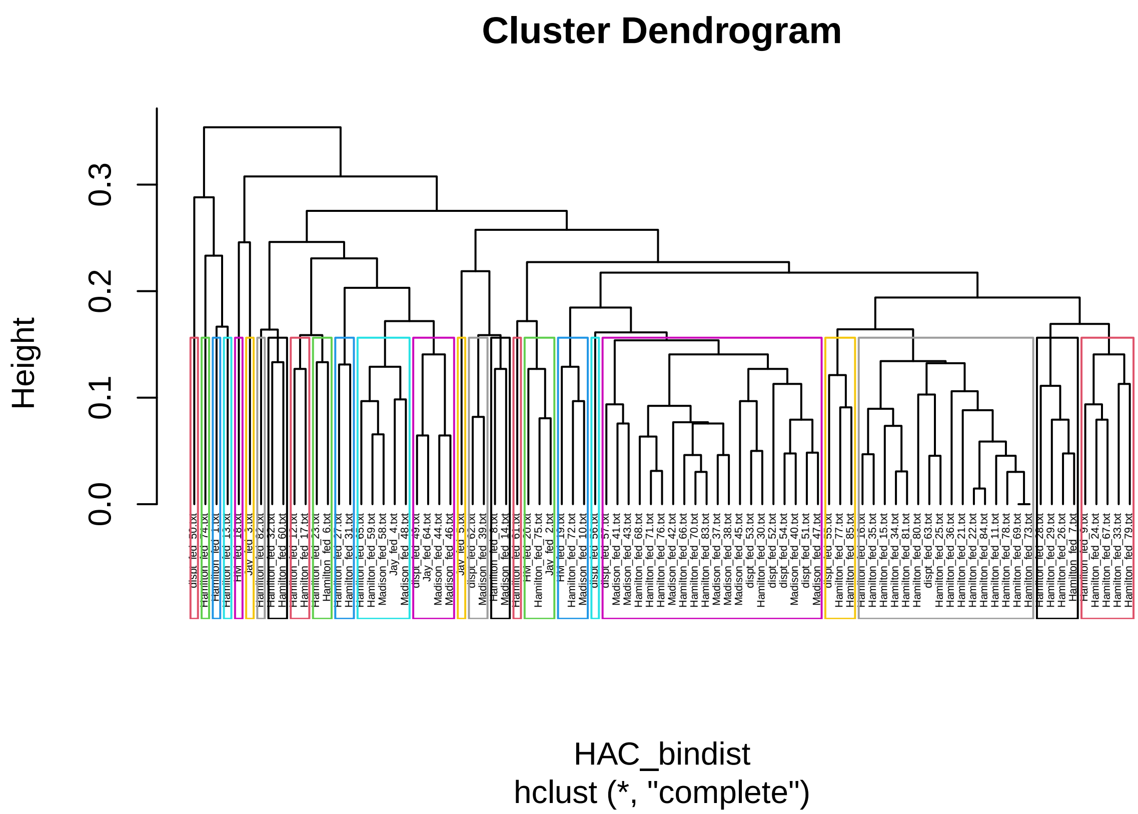

```r
#manhattan
HAC3 <- hclust(HAC_mandist, method = "complete")
plot(HAC3, cex=0.33, hang=-1)
rect.hclust(HAC3, k = 11, border=2:15)
```



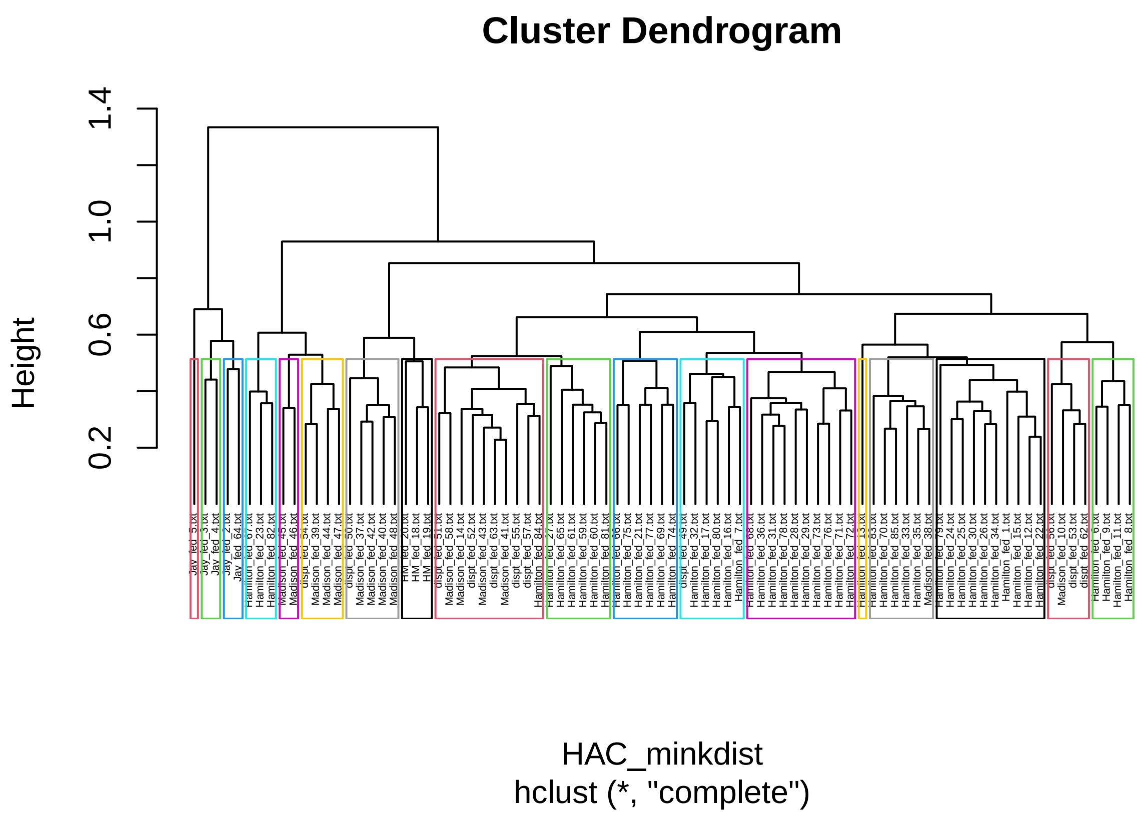

```r
#canberra
HAC4 <- hclust(HAC_candist, method = "complete")
plot(HAC4, cex=0.3, hang=-1)
rect.hclust(HAC4, k = 12, border=2:15)
```




```
#binary
HAC5 <- hclust(HAC_bindist, method = "complete")
plot(HAC5, cex=0.3, hang=-1)
rect.hclust(HAC5, k = 25, border=2:35)
```




```r
#minkowski
HAC6 <- hclust(HAC_minkdist, method = "complete")
plot(HAC6, cex=0.3, hang=-1)
rect.hclust(HAC6, k = 18, border=2:25)
```



Each cluster dendrogram appears to come to the same conclusion as the k-means plot above: that the disputed papers were written by either Hamilton or Madison.

Supervised Learning

First steps, loading packages and adjusting data.

```
library(rpart)
library(rpart.plot)

#create a data frame (df) for each author
hamPapers <- fedPapers[fedPapers$author == "Hamilton",]
HamMadPapers <- fedPapers[fedPapers$author =="HM",]
jayPapers <- fedPapers[fedPapers$author == "Jay",]
madPapers <- fedPapers[fedPapers$author =="Madison",]

#function that finds the average tf-dif of a word for a specific author
createWordMean <- function(x) {
  y <- ncol(x)
  x <- colMeans(x[,3:y])
  newVec_1 <- c()
  for (i in 1:length(x)){
    newVec_1[i] <- x[[i]]
  }
  print(newVec_1)
}
#run the function on new dfs from above
hamVector <- createWordMean(hamPapers)
```

```
##  [1] 0.315607843 0.053764706 0.004784314 0.080803922 0.339490196 0.046745098
##  [7] 0.072549020 0.117725490 0.048823529 0.308372549 0.061921569 0.030156863
## [13] 0.104588235 0.038235294 0.006607843 0.001980392 0.012294118 0.020803922
## [19] 0.092450980 0.080215686 0.017431373 0.048117647 0.098941176 0.007058824
## [25] 0.031921569 0.029019608 0.343901961 0.021313725 0.159411765 0.158588235
## [31] 0.052549020 0.062196078 0.039960784 0.033705882 0.004254902 0.031882353
## [37] 0.090294118 0.006549020 0.957176471 0.047431373 0.035686275 0.020313725
## [43] 0.098019608 0.022666667 0.021705882 0.029392157 0.028960784 0.016019608
## [49] 0.028941176 0.044549020 0.221176471 1.289941176 0.074686275 0.005196078
## [55] 0.037098039 0.003372549 0.093490196 0.591078431 0.004568627 0.047313725
## [61] 0.020607843 0.017450980 0.013627451 0.012254902 0.160490196 0.032921569
## [67] 0.092392157 0.078960784 0.122725490 0.002078431
```

```
HamMadVector <- createWordMean(HamMadPapers)
```

```
##  [1] 0.213333333 0.042666667 0.006000000 0.047000000 0.530666667 0.018000000
##  [7] 0.082333333 0.080000000 0.044333333 0.066666667 0.021333333 0.028333333
## [13] 0.175000000 0.005333333 0.002333333 0.000000000 0.002333333 0.007666667
## [19] 0.085666667 0.108666667 0.081333333 0.046000000 0.066666667 0.020333333
## [25] 0.087000000 0.007666667 0.249000000 0.029666667 0.104000000 0.108666667
## [31] 0.054000000 0.017333333 0.042000000 0.013666667 0.005333333 0.024000000
## [37] 0.034666667 0.012666667 0.838666667 0.093666667 0.044000000 0.015000000
## [43] 0.053666667 0.006000000 0.004666667 0.002333333 0.017333333 0.021333333
## [49] 0.031000000 0.037666667 0.101666667 1.267000000 0.132333333 0.009666667
## [55] 0.004333333 0.000000000 0.065000000 0.382666667 0.009000000 0.005333333
## [61] 0.114000000 0.047333333 0.002333333 0.015666667 0.137333333 0.047333333
## [67] 0.016666667 0.097000000 0.025666667 0.000000000
```

```
jayVector <- createWordMean(jayPapers)
```

```
##  [1] 0.1598 0.0360 0.0198 0.0252 0.7152 0.0376 0.0852 0.1568 0.0360 0.2754
## [11] 0.0268 0.0492 0.1362 0.0330 0.0082 0.0000 0.0076 0.0060 0.0960 0.0910
## [21] 0.0164 0.0288 0.0868 0.0148 0.0090 0.0526 0.2714 0.0446 0.0936 0.2048
## [31] 0.0334 0.0568 0.0868 0.0212 0.0018 0.0150 0.1080 0.0066 0.6390 0.0746
## [41] 0.0814 0.0434 0.1608 0.0660 0.0174 0.0414 0.0446 0.0214 0.0512 0.0628
## [51] 0.2434 0.8544 0.1416 0.0080 0.0140 0.0014 0.0532 0.4834 0.0000 0.0018
## [61] 0.0248 0.0288 0.0184 0.0210 0.0986 0.0516 0.1260 0.0950 0.1252 0.0064
```

```
madVector <- createWordMean(madPapers)
```

```
##  [1] 0.2698000000 0.0553333333 0.0110666667 0.0594666667 0.4196666667
##  [6] 0.0298000000 0.0748666667 0.1340666667 0.0295333333 0.2895333333
## [11] 0.0740666667 0.0354000000 0.1660666667 0.0283333333 0.0040000000
## [16] 0.0009333333 0.0098000000 0.0310666667 0.0945333333 0.0685333333
## [21] 0.0224666667 0.0510666667 0.1004666667 0.0078000000 0.0188666667
## [26] 0.0214666667 0.2864000000 0.0268666667 0.1714000000 0.1498000000
## [31] 0.0480666667 0.0656000000 0.0495333333 0.0338000000 0.0018666667
## [36] 0.0428000000 0.0950000000 0.0051333333 0.8687333333 0.1050666667
## [41] 0.0428000000 0.0236000000 0.0842000000 0.0178666667 0.0198666667
## [46] 0.0237333333 0.0290000000 0.0213333333 0.0287333333 0.0410000000
## [51] 0.2015333333 1.3752666667 0.0891333333 0.0062000000 0.0077333333
## [56] 0.0014000000 0.0831333333 0.4568666667 0.0012666667 0.0022000000
## [61] 0.0257333333 0.0208666667 0.0116000000 0.0060000000 0.1638000000
## [66] 0.0265333333 0.1066666667 0.0770666667 0.0606666667 0.0022666667
```

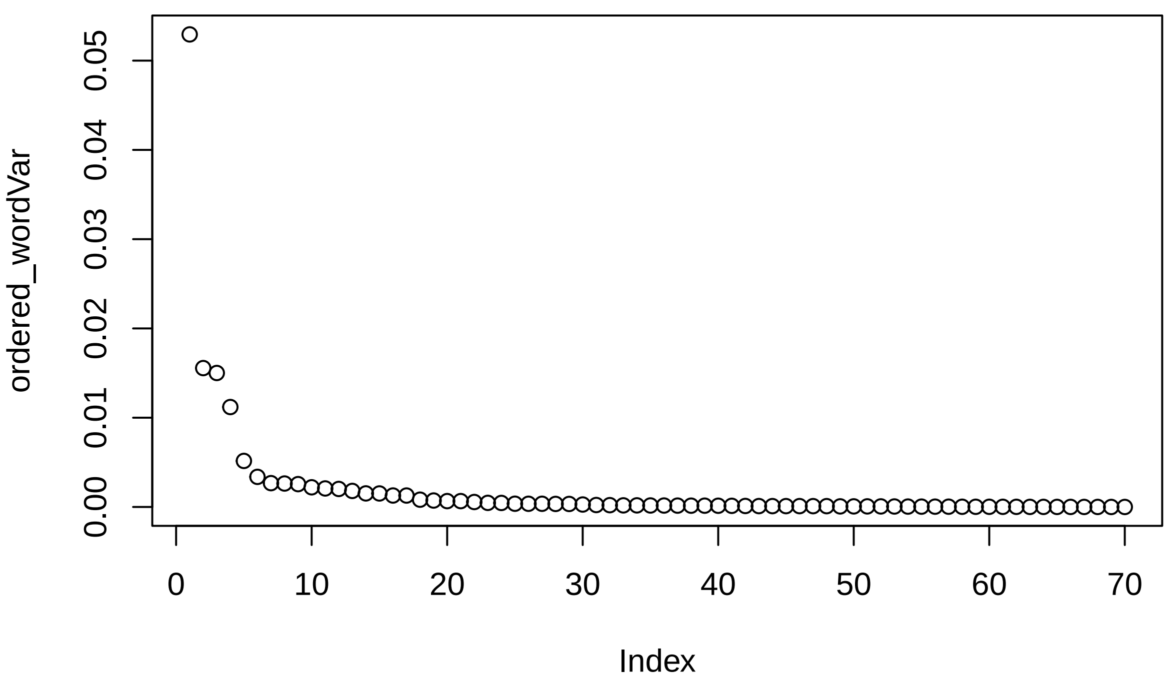
```
columns <- colnames(hamPapers)
```

```r
#new df with results from above
newFedPapers <- hamPapers
newFedPapers <- data.frame(rbind(HamMadVector,HamMadVector, jayVector, madVector))

#make column names the words so its easy to identify
colnames(newFedPapers) <-columns[3:length(columns)]

#lets take a look
newFedPapers[,1:3]
```

```
##                        a        all       also
## HamMadVector   0.2133333 0.04266667 0.00600000
## HamMadVector.1 0.2133333 0.04266667 0.00600000
## jayVector      0.1598000 0.03600000 0.01980000
## madVector      0.2698000 0.05533333 0.01106667
```

Next, calculate the variance in frequency of words. The words with the most variance among use suggests that they are likely unique to a specific author's writing style. For this analysis, the words in the top 45% will be selected.

```r
#find variance
wordVar <- sapply(newFedPapers, var)
ordered_wordVar <- sort(wordVar, decreasing=TRUE)
plot(ordered_wordVar)
```




```r
#select top 45% of words
uniqueWords <- data.frame(ordered_wordVar[1:(length(ordered_wordVar)*0.45)])
```

Take a look at what words are unique

```r
# number of words chosen
length(rownames(uniqueWords))
```

```
## [1] 31
```

```r
#words chosen
rownames(uniqueWords)
```

```
##  [1] "the"    "be"     "and"    "of"     "that"   "will"   "to"     "was"
##  [9] "or"     "would"  "it"     "a"      "his"    "not"    "as"     "had"
## [17] "is"     "our"    "which"  "been"   "may"    "their"  "more"   "if"
## [25] "from"   "one"    "should" "by"     "in"     "have"   "can"
```

New data frame is created with the unique words

```r
uniqueFedPapers <- fedPapers[,c("author", "filename", rownames(uniqueWords))]

rownames(uniqueFedPapers) <- uniqueFedPapers$filename
```

```
## Warning: Setting row names on a tibble is deprecated.
```

Decision Tree Modeling

Time to create a train model

```r
set.seed(330)
setwd("/Users/victoriahaley/")
fedDT <- read_csv("Desktop/IST 707/fedPapers85.csv")
```

```
## Rows: 85 Columns: 72
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): author, filename
## dbl (70): a, all, also, an, and, any, are, as, at, be, been, but, by, can, d...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
#fedDT <- fedDT[,-1]

fedDT <- fedDT[-c(which(fedDT$author =='Jay'), which(fedDT$author =="HM")),]
fedDT <- droplevels(fedDT)
fedDT <- fedDT[,-2]


fedDT_disputed <- fedDT[1:11,]
fedDT_not_disputed <- fedDT[-c(1:11),]
```

Indexes make sure that an equal amount of Hamilton and an Equal amount of Madison's papers are included in the training data.

```r
indexes <- sample(1:55, .65*(length(1:55)))
indexes <- c(indexes, sample(56:nrow(fedDT_not_disputed), .65*length(56:nrow(fedDT_not_disputed))))
```

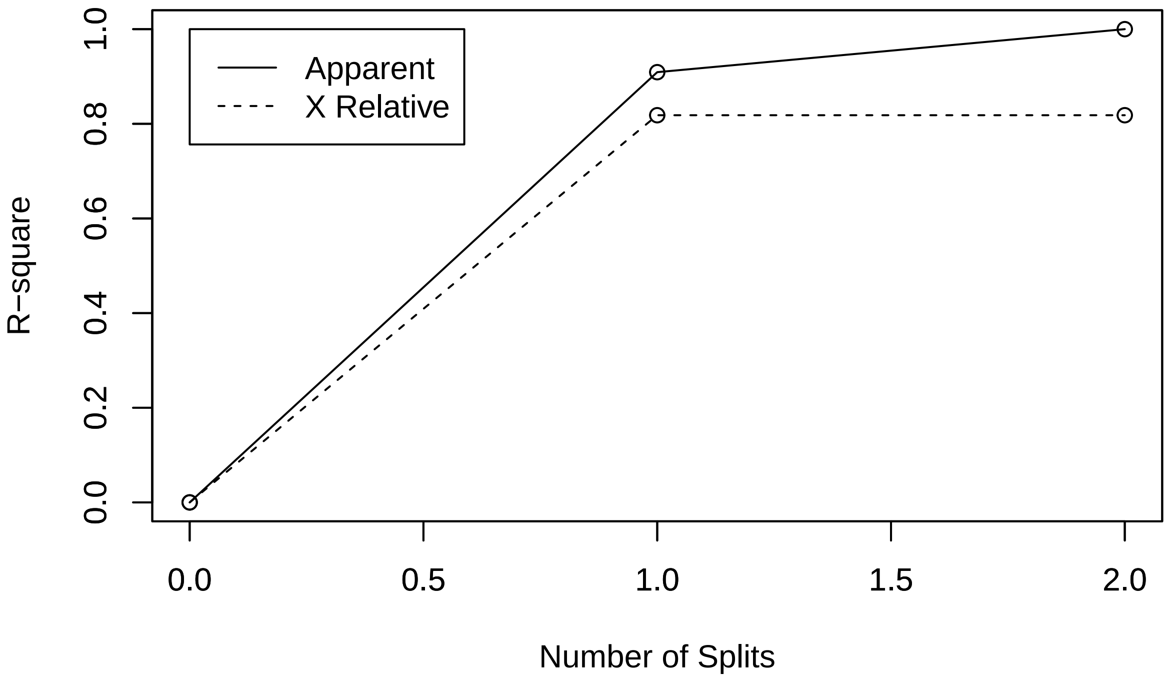
Let's plant some trees

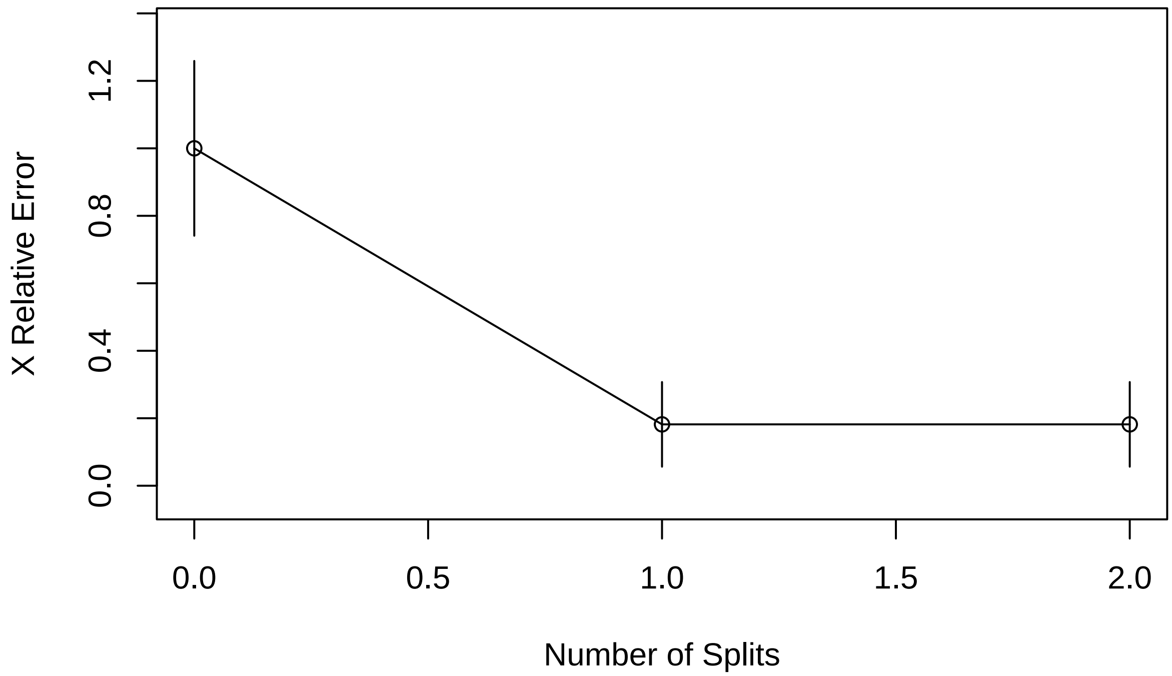
```r
fed_DT1 <- rpart(author ~ . , data= fedDT_not_disputed[indexes,], method = 'class', control = rpart.con
rsq.rpart(fed_DT1)
```

```
##
## Classification tree:
## rpart(formula = author ~ ., data = fedDT_not_disputed[indexes,
##     ], method = "class", model = T, control = rpart.control(minbucket = 1,
```

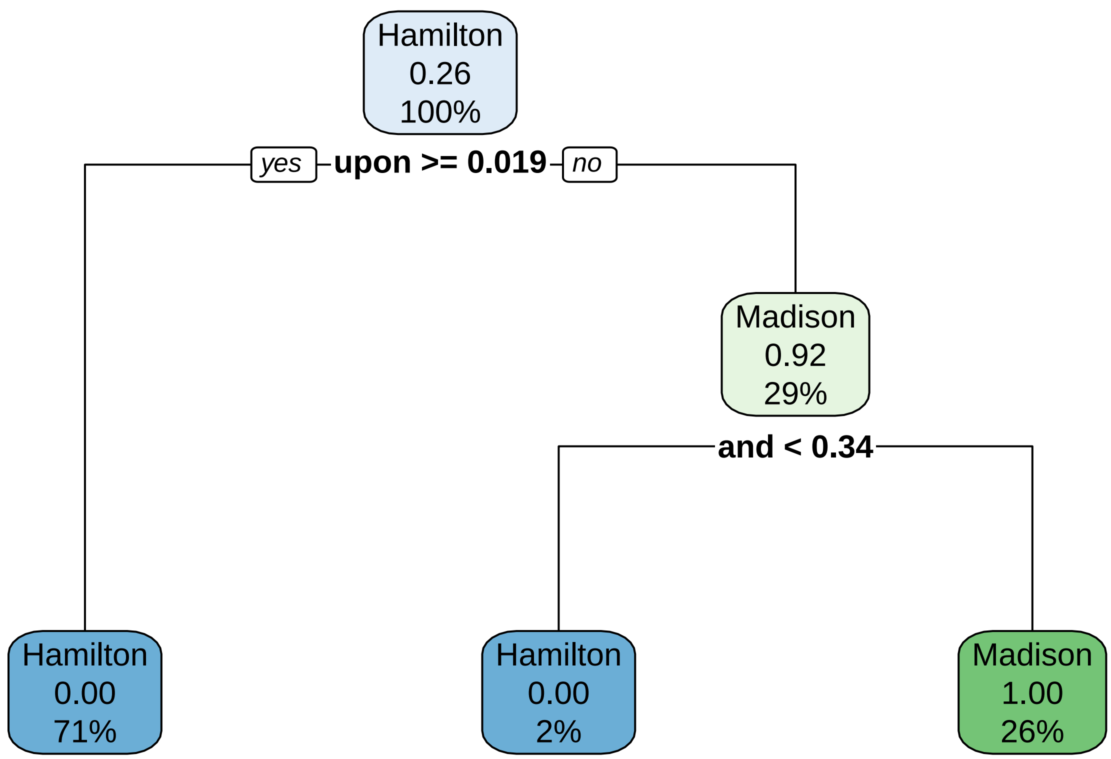
```
##     minsplit = 1, cp = -1))
##
## Variables actually used in tree construction:
## [1] and  upon
##
## Root node error: 11/42 = 0.2619
##
## n= 42
##
##          CP nsplit rel error  xerror    xstd
## 1  0.909091      0  1.000000 1.00000 0.25904
## 2  0.090909      1  0.090909 0.18182 0.12547
## 3 -1.000000      2  0.000000 0.18182 0.12547
```

```
## Warning in rsq.rpart(fed_DT1): may not be applicable for this method
```






```
rpart.plot(fed_DT1)
```




```
#matrix to see how well the model predicted
fedDT1_preds <- predict(fed_DT1, fedDT_not_disputed[-indexes,], type='class')
table(fedDT_not_disputed$author[-indexes],fedDT1_preds)
```

```
##             fedDT1_preds
##              Hamilton Madison
##    Hamilton        20       0
```

```
##   Madison         2       2
```

This model correctly predicted the authors in all but 22 out of 24 papers. Let's see how it does compared to the test data.

```
predict(fed_DT1, fedDT_disputed, type='class')
```

```
##        1        2        3        4        5        6        7        8
##  Madison  Madison Hamilton Hamilton  Madison Hamilton Hamilton  Madison
##        9       10       11
##  Madison  Madison Hamilton
## Levels: Hamilton Madison
```

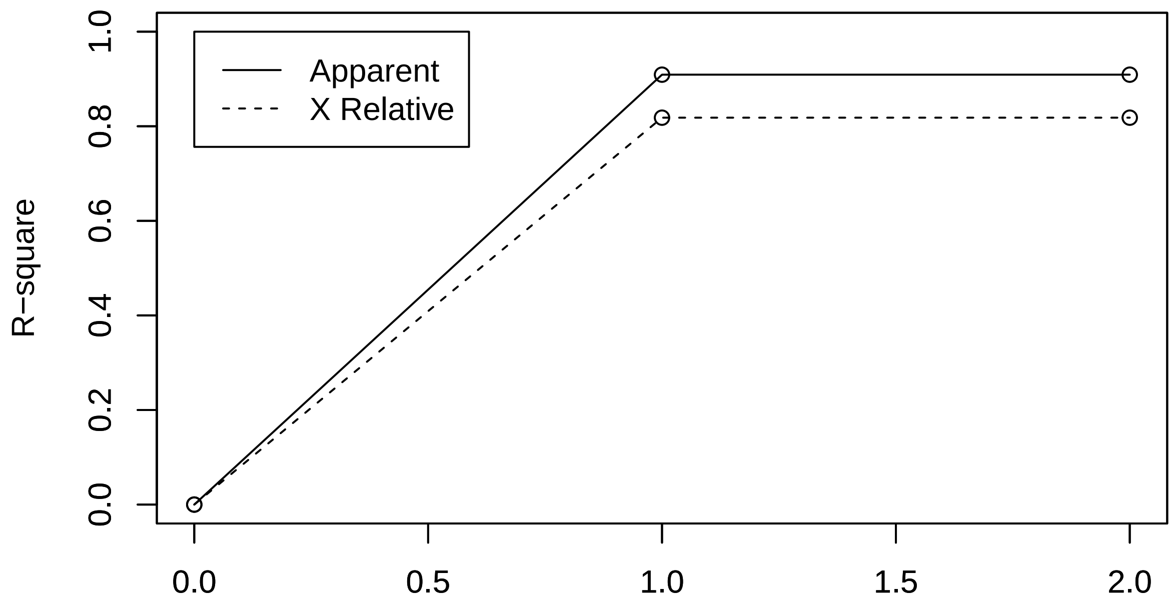
This model indicates that Madison wrote 6 out of the 11 disputed papers, specifically paper #49, 50, 53, 56, 57, and 62. This is pretty close to what was seen above with the clusters.

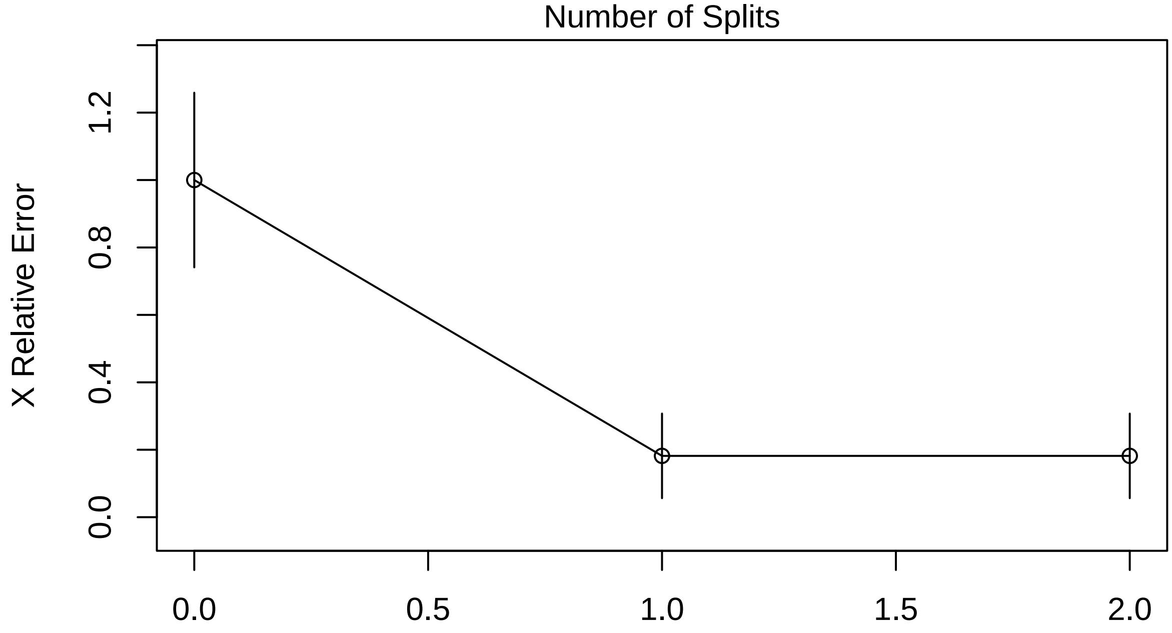
Let's try another tree

```
fed_DT2 <- rpart(author ~ . , data= fedDT_not_disputed[indexes,], method = 'class', control = rpart.con
rsq.rpart(fed_DT2)
```

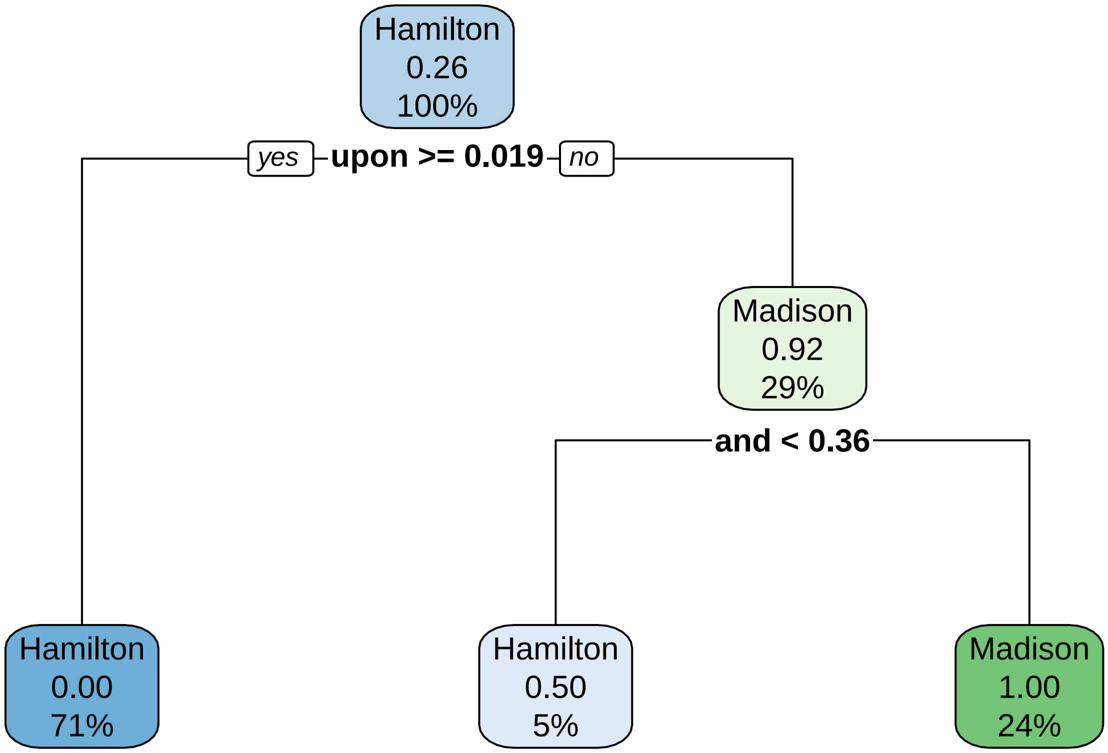
```
##
## Classification tree:
## rpart(formula = author ~ ., data = fedDT_not_disputed[indexes,
##     ], method = "class", model = T, control = rpart.control(minbucket = 2,
##     minsplit = 2, cp = -2))
##
## Variables actually used in tree construction:
## [1] and  upon
##
## Root node error: 11/42 = 0.2619
##
## n= 42
##
##         CP nsplit rel error  xerror    xstd
## 1  0.90909      0  1.000000 1.00000 0.25904
## 2  0.00000      1  0.090909 0.18182 0.12547
## 3 -2.00000      2  0.090909 0.18182 0.12547
```

```
## Warning in rsq.rpart(fed_DT2): may not be applicable for this method
```






```
rpart.plot(fed_DT2)
```




```
predict(fed_DT2, fedDT_disputed, type='class')
```

```
##        1        2        3        4        5        6        7        8
##  Madison  Madison Hamilton Hamilton  Madison Hamilton Hamilton  Madison
##        9       10       11
## Hamilton  Madison Hamilton
## Levels: Hamilton Madison
```

This model, with adjusted controls, has Hamilton being the author of 6 out of the 11 papers. These parameters are not good and should be thrown out considering that it does not align with what the rest of the data is showing.

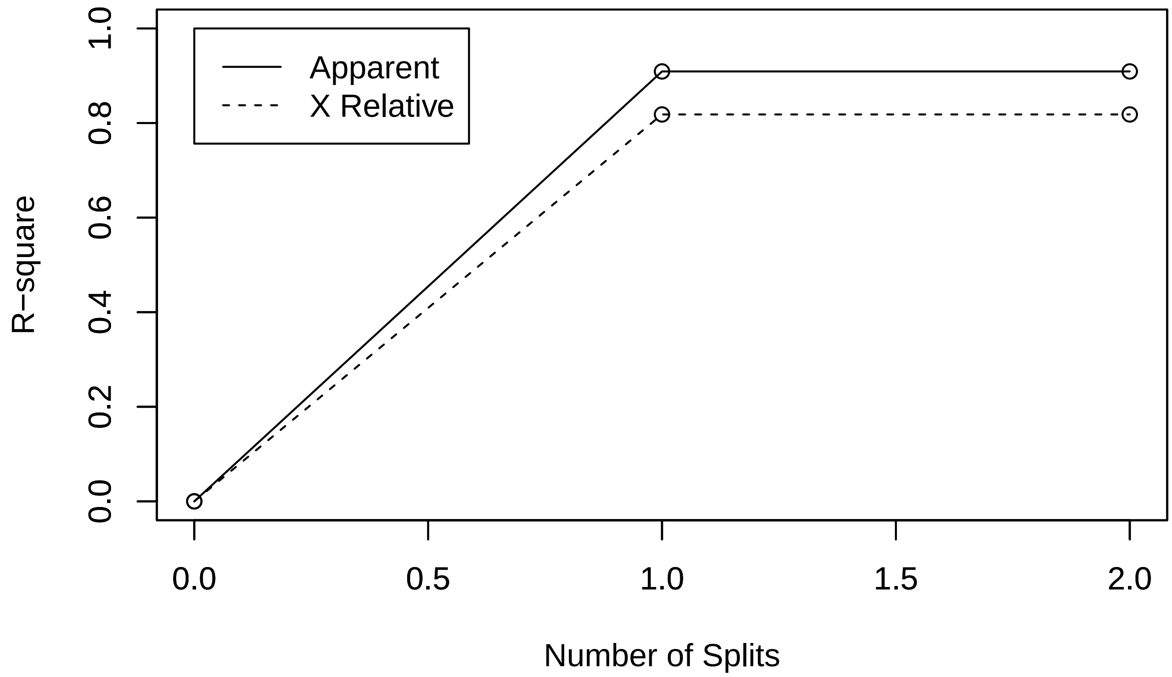
Tree #3:

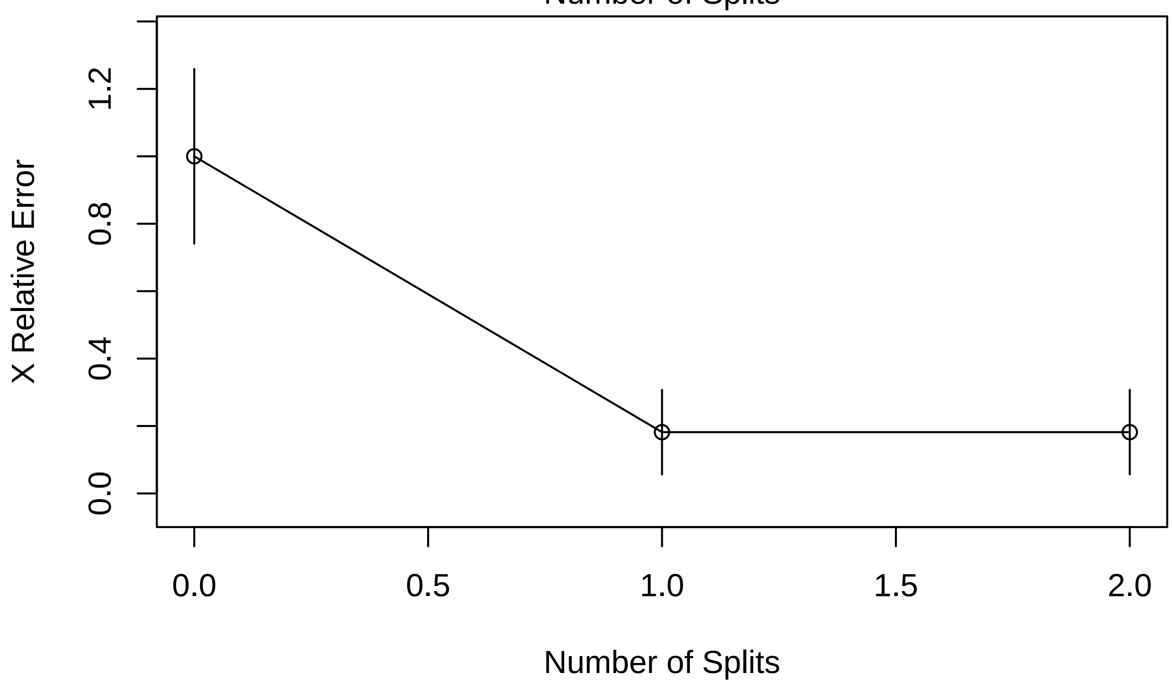
```
fed_DT3 <- rpart(author ~ . , data= fedDT_not_disputed[indexes,], method = 'class', control = rpart.con
rsq.rpart(fed_DT3)
```

```
##
## Classification tree:
## rpart(formula = author ~ ., data = fedDT_not_disputed[indexes,
##     ], method = "class", control = rpart.control(minbucket = 3,
##     minsplit = 2, cp = -1, model = TRUE))
##
## Variables actually used in tree construction:
## [1] a    upon
##
## Root node error: 11/42 = 0.2619
##
## n= 42
##
##          CP nsplit rel error  xerror    xstd
## 1  0.90909      0  1.000000 1.00000 0.25904
```

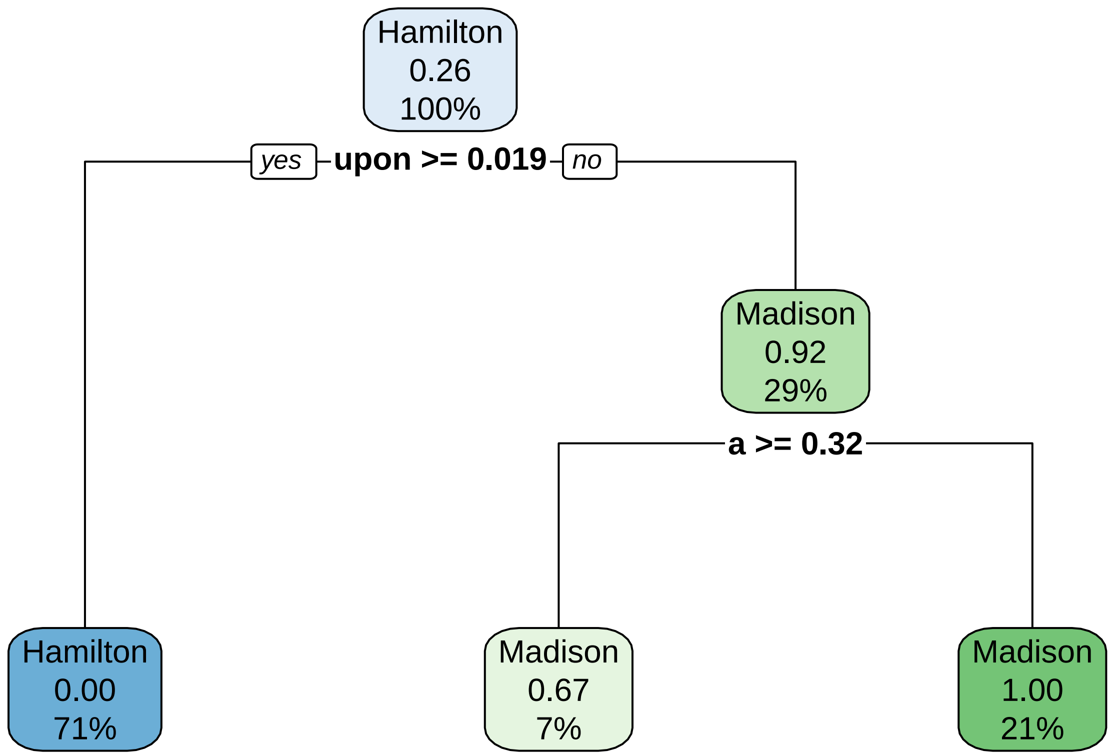
```
## 2  0.00000      1  0.090909 0.18182 0.12547
## 3 -1.00000      2  0.090909 0.18182 0.12547
```

```
## Warning in rsq.rpart(fed_DT3): may not be applicable for this method
```






```
rpart.plot(fed_DT3)
```




```
predict(fed_DT3, fedDT_disputed, type='class')
```

```
##       1       2       3       4       5       6       7       8       9      10
## Madison Madison Madison Madison Madison Madison Madison Madison Madison Madison
##      11
## Madison
## Levels: Hamilton Madison
```

This is the worst one, it predicted all of them to be written by Madison.

Conclusion

After analyzing the Federalist Papers dataset using k-Means and HAC clustering methods, as well as through decision tree modeling, it is now possible to shed light on the disputed authorship of the Federalist Papers. Specifically, the analysis suggests that James Madison wrote most of the disputed papers, specifically papers #50, 53, 56, 57, and 62.

The clustering techniques provided evidence that suggests that Madison wrote most of the disputed papers. The k-Means and HAC produced similar clustering results, indicating which of the disputed papers were written by Hamilton and which were written by Madison.

The decision tree model was helpful in interpreting the results of the analysis, as it provided a clear visual "tree branches" of the decision-making process. In particular, the decision tree showed that certain function words, specifically 'upon' and 'and', were the most important predictors of authorship attribution.

Overall, the clustering and decision tree models seem to agree that Madison was the most likely author of the disputed papers. Although it is possible that some of the papers were co-authored by Hamilton and Madison, the analysis provides evidence to suggest that Madison was the author of most of these papers.