Document Classification Using Decision Trees

Sarah Morris

Decision Trees To Solve a Mystery in History

We previously used clustering techniques to see if we could determine the authorship of the disputed federalist papers. Now, we will use decision trees to see if they can provide us with further insight into the authorship of the papers.

Data Preparation

```
#importing dataset
papers = read.csv("~/Downloads/week4_resources (2)/fedPapers85_fromClass.csv")
#remove filename
papers = papers[,-2]
head(papers)
```

```
##   author     a   all  also    an   and   any   are    as    at    be  been
## 1  dispt 0.280 0.052 0.009 0.096 0.358 0.026 0.131 0.122 0.017 0.411 0.026
## 2  dispt 0.177 0.063 0.013 0.038 0.393 0.063 0.051 0.139 0.114 0.393 0.165
## 3  dispt 0.339 0.090 0.008 0.030 0.301 0.008 0.068 0.203 0.023 0.474 0.015
## 4  dispt 0.270 0.024 0.016 0.024 0.262 0.056 0.064 0.111 0.056 0.365 0.127
## 5  dispt 0.303 0.054 0.027 0.034 0.404 0.040 0.128 0.148 0.013 0.344 0.047
## 6  dispt 0.245 0.059 0.007 0.067 0.282 0.052 0.111 0.252 0.015 0.297 0.030
##     but    by   can    do  down  even every  for.  from   had   has  have her
## 1 0.009 0.140 0.035 0.026 0.000 0.009 0.044 0.096 0.044 0.035 0.017 0.044   0
## 2 0.000 0.139 0.000 0.013 0.000 0.025 0.000 0.076 0.101 0.101 0.013 0.152   0
## 3 0.038 0.173 0.023 0.000 0.008 0.015 0.023 0.098 0.053 0.008 0.015 0.023   0
## 4 0.032 0.167 0.056 0.000 0.000 0.024 0.040 0.103 0.079 0.016 0.024 0.143   0
## 5 0.061 0.209 0.088 0.000 0.000 0.020 0.027 0.141 0.074 0.000 0.054 0.047   0
## 6 0.037 0.186 0.000 0.000 0.007 0.007 0.007 0.067 0.096 0.022 0.015 0.119   0
##     his   if.   in.  into    is    it   its   may  more  must my    no   not
## 1 0.017 0.000 0.262 0.009 0.157 0.175 0.070 0.035 0.026 0.026  0 0.035 0.114
## 2 0.000 0.025 0.291 0.025 0.038 0.127 0.038 0.038 0.000 0.013  0 0.000 0.127
## 3 0.000 0.023 0.308 0.038 0.150 0.173 0.030 0.120 0.038 0.083  0 0.030 0.068
## 4 0.024 0.040 0.238 0.008 0.151 0.222 0.048 0.056 0.056 0.071  0 0.032 0.087
```

```
## 5 0.020 0.034 0.263 0.013 0.189 0.108 0.013 0.047 0.067 0.013  0 0.047 0.128
## 6 0.067 0.030 0.401 0.037 0.260 0.156 0.015 0.074 0.045 0.015  0 0.059 0.134
##   now    of    on   one  only    or   our shall should    so  some  such  than
## 1   0 0.900 0.140 0.026 0.035 0.096 0.017 0.017  0.017 0.035 0.009 0.026 0.009
## 2   0 0.747 0.139 0.025 0.000 0.114 0.000 0.000  0.013 0.013 0.063 0.000 0.000
## 3   0 0.858 0.150 0.030 0.023 0.060 0.000 0.008  0.068 0.038 0.030 0.045 0.023
## 4   0 0.802 0.143 0.032 0.048 0.064 0.016 0.016  0.032 0.040 0.024 0.008 0.000
## 5   0 0.869 0.054 0.047 0.027 0.081 0.027 0.000  0.000 0.027 0.067 0.027 0.047
## 6   0 0.876 0.141 0.052 0.022 0.074 0.030 0.015  0.030 0.007 0.045 0.015 0.030
##    that   the their  then there things  this    to up  upon   was  were  what
## 1 0.184 1.425 0.114 0.000 0.009  0.009 0.044 0.507  0 0.000 0.009 0.017 0.000
## 2 0.152 1.254 0.165 0.000 0.000  0.000 0.051 0.355  0 0.013 0.051 0.000 0.000
## 3 0.188 1.490 0.053 0.015 0.015  0.000 0.075 0.361  0 0.000 0.008 0.015 0.008
## 4 0.238 1.326 0.071 0.008 0.000  0.000 0.103 0.532  0 0.000 0.087 0.079 0.008
## 5 0.162 1.193 0.027 0.007 0.007  0.000 0.094 0.485  0 0.000 0.027 0.020 0.020
## 6 0.208 1.469 0.089 0.007 0.007  0.000 0.126 0.445  0 0.000 0.007 0.030 0.015
##    when which   who  will  with would your
## 1 0.009 0.175 0.044 0.009 0.087 0.192    0
## 2 0.000 0.114 0.038 0.089 0.063 0.139    0
## 3 0.000 0.105 0.008 0.173 0.045 0.068    0
## 4 0.024 0.167 0.000 0.079 0.079 0.064    0
## 5 0.007 0.155 0.027 0.168 0.074 0.040    0
## 6 0.037 0.186 0.045 0.111 0.089 0.037    0
```

Since we know that Jay is not the author of these papers, we can remove any papers written by him from this dataset. We can also remove papers written jointly by them (HM)

```
#remove Jay papers
papers=papers[papers$author!="Jay",]
#remove HM papers
papers=papers[papers$author!="HM",]
#check to ensure data is clean
head(papers)
```

```
##   author     a   all  also    an   and   any   are    as    at    be  been
## 1  dispt 0.280 0.052 0.009 0.096 0.358 0.026 0.131 0.122 0.017 0.411 0.026
## 2  dispt 0.177 0.063 0.013 0.038 0.393 0.063 0.051 0.139 0.114 0.393 0.165
## 3  dispt 0.339 0.090 0.008 0.030 0.301 0.008 0.068 0.203 0.023 0.474 0.015
## 4  dispt 0.270 0.024 0.016 0.024 0.262 0.056 0.064 0.111 0.056 0.365 0.127
## 5  dispt 0.303 0.054 0.027 0.034 0.404 0.040 0.128 0.148 0.013 0.344 0.047
## 6  dispt 0.245 0.059 0.007 0.067 0.282 0.052 0.111 0.252 0.015 0.297 0.030
##     but    by   can    do  down  even every  for.  from   had   has  have her
## 1 0.009 0.140 0.035 0.026 0.000 0.009 0.044 0.096 0.044 0.035 0.017 0.044   0
## 2 0.000 0.139 0.000 0.013 0.000 0.025 0.000 0.076 0.101 0.101 0.013 0.152   0
## 3 0.038 0.173 0.023 0.000 0.008 0.015 0.023 0.098 0.053 0.008 0.015 0.023   0
## 4 0.032 0.167 0.056 0.000 0.000 0.024 0.040 0.103 0.079 0.016 0.024 0.143   0
## 5 0.061 0.209 0.088 0.000 0.000 0.020 0.027 0.141 0.074 0.000 0.054 0.047   0
## 6 0.037 0.186 0.000 0.000 0.007 0.007 0.007 0.067 0.096 0.022 0.015 0.119   0
##     his   if.   in.  into    is    it   its   may  more  must my    no   not
## 1 0.017 0.000 0.262 0.009 0.157 0.175 0.070 0.035 0.026 0.026  0 0.035 0.114
## 2 0.000 0.025 0.291 0.025 0.038 0.127 0.038 0.038 0.000 0.013  0 0.000 0.127
## 3 0.000 0.023 0.308 0.038 0.150 0.173 0.030 0.120 0.038 0.083  0 0.030 0.068
## 4 0.024 0.040 0.238 0.008 0.151 0.222 0.048 0.056 0.056 0.071  0 0.032 0.087
## 5 0.020 0.034 0.263 0.013 0.189 0.108 0.013 0.047 0.067 0.013  0 0.047 0.128
## 6 0.067 0.030 0.401 0.037 0.260 0.156 0.015 0.074 0.045 0.015  0 0.059 0.134
##   now    of    on   one  only    or   our shall should    so  some  such  than
## 1   0 0.900 0.140 0.026 0.035 0.096 0.017 0.017  0.017 0.035 0.009 0.026 0.009
## 2   0 0.747 0.139 0.025 0.000 0.114 0.000 0.000  0.013 0.013 0.063 0.000 0.000
## 3   0 0.858 0.150 0.030 0.023 0.060 0.000 0.008  0.068 0.038 0.030 0.045 0.023
## 4   0 0.802 0.143 0.032 0.048 0.064 0.016 0.016  0.032 0.040 0.024 0.008 0.000
## 5   0 0.869 0.054 0.047 0.027 0.081 0.027 0.000  0.000 0.027 0.067 0.027 0.047
## 6   0 0.876 0.141 0.052 0.022 0.074 0.030 0.015  0.030 0.007 0.045 0.015 0.030
##    that   the their  then there things  this    to up  upon   was  were  what
## 1 0.184 1.425 0.114 0.000 0.009  0.009 0.044 0.507  0 0.000 0.009 0.017 0.000
## 2 0.152 1.254 0.165 0.000 0.000  0.000 0.051 0.355  0 0.013 0.051 0.000 0.000
## 3 0.188 1.490 0.053 0.015 0.015  0.000 0.075 0.361  0 0.000 0.008 0.015 0.008
## 4 0.238 1.326 0.071 0.008 0.000  0.000 0.103 0.532  0 0.000 0.087 0.079 0.008
```

```
## 5 0.162 1.193 0.027 0.007 0.007  0.000 0.094 0.485  0 0.000 0.027 0.020 0.020
## 6 0.208 1.469 0.089 0.007 0.007  0.000 0.126 0.445  0 0.000 0.007 0.030 0.015
##    when which   who  will  with would your
## 1 0.009 0.175 0.044 0.009 0.087 0.192    0
## 2 0.000 0.114 0.038 0.089 0.063 0.139    0
## 3 0.000 0.105 0.008 0.173 0.045 0.068    0
## 4 0.024 0.167 0.000 0.079 0.079 0.064    0
## 5 0.007 0.155 0.027 0.168 0.074 0.040    0
## 6 0.037 0.186 0.045 0.111 0.089 0.037    0
```

Separate data into training and testing groups

```
#separate the disputed files from other
known<-papers[papers$author!="dispt",]
unknown<-papers[papers$author=="dispt",]
#use 70% of "known" papers as training set and 30% as test set
sample <- sample(c(TRUE, FALSE), nrow(known), replace=TRUE, prob=c(0.7,0.3))
train<-known[sample,]
test<-known[!sample,]
```

Now that our data is split into three parts - the disputed papers(which we will use for testing once our model is complete), 70% of our known data (for preliminary training), and 30% of our known data (for preliminary testing). Now that we have these three subsets, we can proceed to build our decision tree.

Build and Tune Decision Tree

First, load the required packages.

```
library(rpart)
library(rpart.plot)
library(e1071)
library(tree)
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
```

```
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts()
──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
errors
```

Now, create the rpart model.

Rpart model1

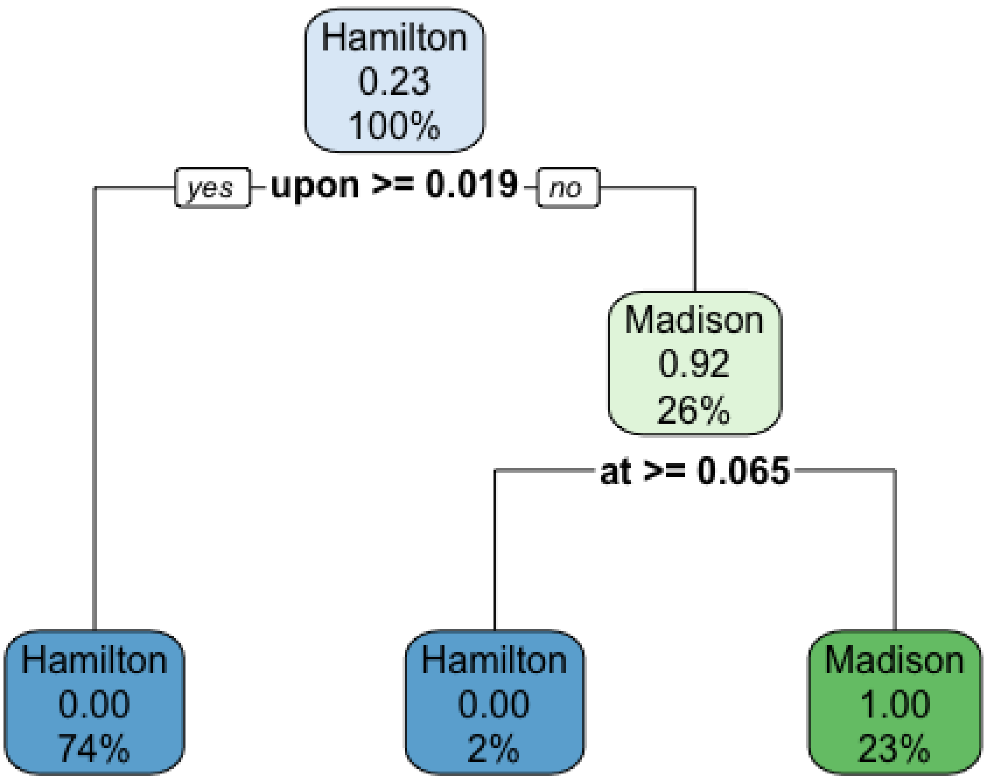
```
#rpart
train_rpart = rpart(author ~ ., data = train,
                    minsplit=3, cp=0)
summary(train_rpart)
```

```
## Call:
## rpart(formula = author ~ ., data = train, minsplit = 3, cp = 0)
##   n= 47
##
##           CP nsplit  rel error    xerror      xstd
## 1 0.90909091      0 1.00000000 1.0000000 0.2638797
## 2 0.09090909      1 0.09090909 0.1818182 0.1257997
## 3 0.00000000      2 0.00000000 0.2727273 0.1523510
##
## Variable importance
##  upon there    on    by    to    an    at
##    29    22    14    12    12     7     4
##
## Node number 1: 47 observations,    complexity param=0.9090909
##   predicted class=Hamilton  expected loss=0.2340426  P(node) =1
##     class counts:    36    11
```

```
##    probabilities: 0.766 0.234
##   left son=2 (35 obs) right son=3 (12 obs)
##   Primary splits:
##       upon  < 0.019  to the right, improve=15.017730, (0 missing)
##       there < 0.0145 to the right, improve=10.294500, (0 missing)
##       on    < 0.0825 to the left,  improve= 8.579635, (0 missing)
##       to    < 0.4885 to the right, improve= 7.547896, (0 missing)
##       by    < 0.14   to the left,  improve= 5.901064, (0 missing)
##   Surrogate splits:
##       there < 0.0115 to the right, agree=0.936, adj=0.750, (0 split)
##       on    < 0.0745 to the left,  agree=0.872, adj=0.500, (0 split)
##       by    < 0.14   to the left,  agree=0.851, adj=0.417, (0 split)
##       to    < 0.4885 to the right, agree=0.851, adj=0.417, (0 split)
##       an    < 0.064  to the right, agree=0.809, adj=0.250, (0 split)
##
## Node number 2: 35 observations
##   predicted class=Hamilton  expected loss=0  P(node) =0.7446809
##     class counts:    35     0
##    probabilities: 1.000 0.000
##
## Node number 3: 12 observations,    complexity param=0.09090909
##   predicted class=Madison   expected loss=0.08333333  P(node) =0.2553191
##     class counts:     1    11
##    probabilities: 0.083 0.917
##   left son=6 (1 obs) right son=7 (11 obs)
##   Primary splits:
##       at   < 0.065  to the right, improve=1.833333, (0 missing)
##       be   < 0.375  to the right, improve=1.833333, (0 missing)
##       even < 0.0215 to the right, improve=1.833333, (0 missing)
##       if.  < 0.0025 to the left,  improve=1.833333, (0 missing)
##       may  < 0.026  to the left,  improve=1.833333, (0 missing)
##
## Node number 6: 1 observations
```

```
##   predicted class=Hamilton  expected loss=0  P(node) =0.0212766
##     class counts:     1     0
##    probabilities: 1.000 0.000
##
## Node number 7: 11 observations
##   predicted class=Madison   expected loss=0  P(node) =0.2340426
##     class counts:     0    11
##    probabilities: 0.000 1.000

rpart.plot(train_rpart)
```



Rpart Model 2

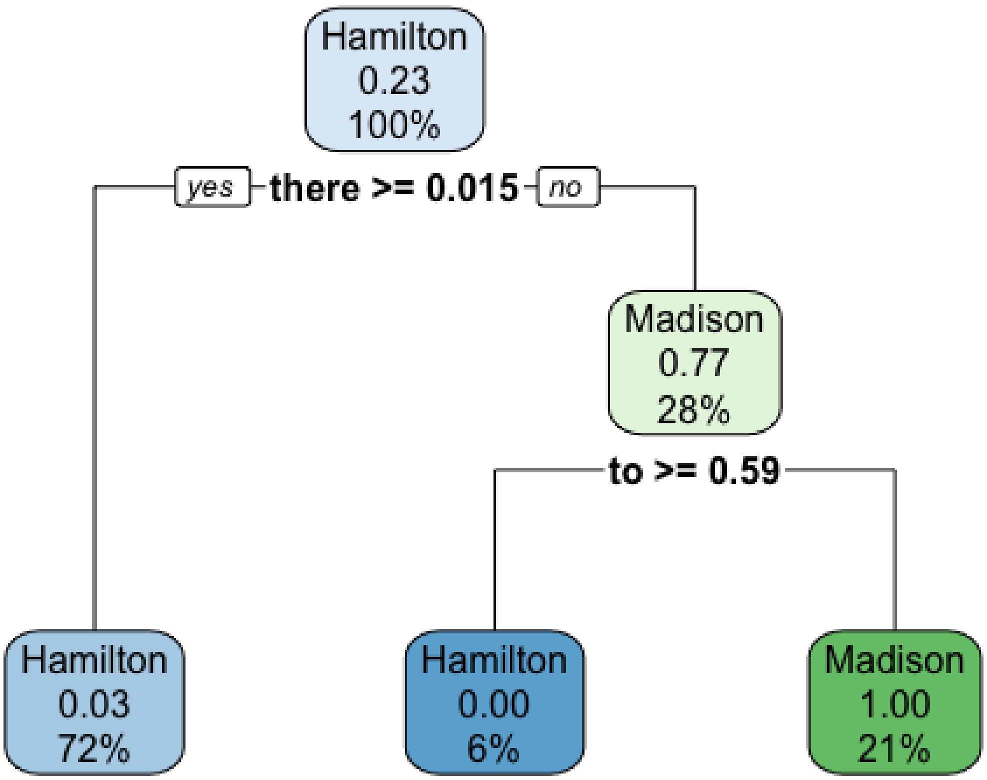
Now we will build a decision tree with more nodes. We know from our tree above, that only certain variables are being used as predictors (the words upon, on, there, to, by, and should). Upon is a very strong predictor which seems to override the model. Therefore, I am removing it from this tree so we can see more nodes.

```r
train_rpart2 <- rpart(author ~  on + there + to + by + should,
          data=train,
          method="class",
          control=rpart.control(minsplit=3, cp=0.002))
summary(train_rpart2)
```

```
## Call:
## rpart(formula = author ~ on + there + to + by + should, data = train,
##     method = "class", control = rpart.control(minsplit = 3, cp = 0.002))
##   n= 47
##
##          CP nsplit  rel error    xerror      xstd
## 1 0.6363636      0 1.00000000 1.0000000 0.2638797
## 2 0.2727273      1 0.36363636 0.6363636 0.2218898
## 3 0.0020000      2 0.09090909 0.3636364 0.1739092
##
## Variable importance
##  there     on     to     by should
##     31     23     23     12     12
##
## Node number 1: 47 observations,    complexity param=0.6363636
##   predicted class=Hamilton  expected loss=0.2340426  P(node) =1
##     class counts:    36    11
##    probabilities: 0.766 0.234
##   left son=2 (34 obs) right son=3 (13 obs)
##   Primary splits:
##       there  < 0.0145 to the right, improve=10.294500, (0 missing)
##       on     < 0.0825 to the left,  improve= 8.579635, (0 missing)
##       to     < 0.4885 to the right, improve= 7.547896, (0 missing)
##       by     < 0.14   to the left,  improve= 5.901064, (0 missing)
##       should < 0.015  to the right, improve= 2.820761, (0 missing)
##   Surrogate splits:
##       on     < 0.0915 to the left,  agree=0.851, adj=0.462, (0 split)
##       to     < 0.4755 to the right, agree=0.809, adj=0.308, (0 split)
```

```
##       by     < 0.14   to the left,  agree=0.787, adj=0.231, (0 split)
##       should < 0.0445 to the left,  agree=0.787, adj=0.231, (0 split)
##
## Node number 2: 34 observations
##   predicted class=Hamilton  expected loss=0.02941176  P(node) =0.7234043
##     class counts:    33     1
##    probabilities: 0.971 0.029
##
## Node number 3: 13 observations,    complexity param=0.2727273
##   predicted class=Madison   expected loss=0.2307692  P(node) =0.2765957
##     class counts:     3    10
##    probabilities: 0.231 0.769
##   left son=6 (3 obs) right son=7 (10 obs)
##   Primary splits:
##       to     < 0.5875 to the right, improve=4.6153850, (0 missing)
##       on     < 0.0535 to the left,  improve=2.7972030, (0 missing)
##       should < 0.0285 to the right, improve=1.6153850, (0 missing)
##       by     < 0.1215 to the left,  improve=1.4820510, (0 missing)
##       there  < 0.006  to the right, improve=0.4153846, (0 missing)
##   Surrogate splits:
##       on     < 0.0535 to the left,  agree=0.923, adj=0.667, (0 split)
##       by     < 0.1215 to the left,  agree=0.846, adj=0.333, (0 split)
##       should < 0.048  to the right, agree=0.846, adj=0.333, (0 split)
##
## Node number 6: 3 observations
##   predicted class=Hamilton  expected loss=0  P(node) =0.06382979
##     class counts:     3     0
##    probabilities: 1.000 0.000
##
## Node number 7: 10 observations
##   predicted class=Madison   expected loss=0  P(node) =0.212766
##     class counts:     0    10
##    probabilities: 0.000 1.000
```

```r
#plot new train model
rpart.plot(train_rpart2)
```



Rpart model 3

Checking the variable importance before continuing to tune our decision tree:

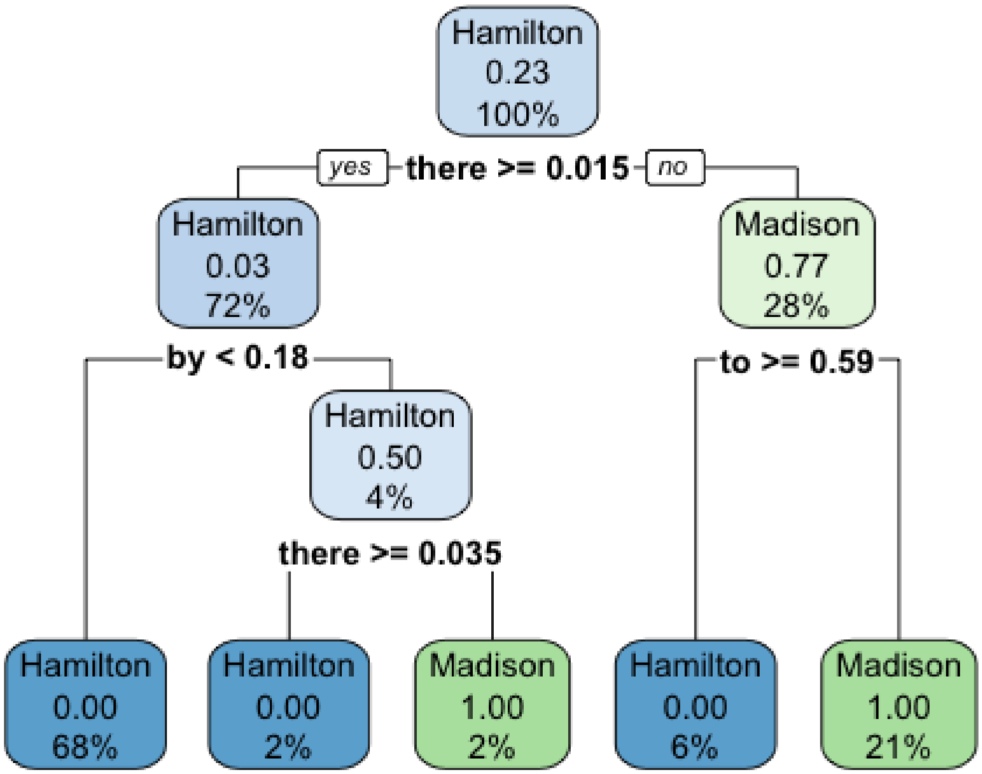
```r
train_rpart$variable.importance

##     upon     there       on       by       to       an       at
## 15.017730 11.263298  7.508865  6.257388  6.257388  3.754433  1.833333
```

For the last model, we will leave out upon and on to see the new results of the decision tree.

```r
train_rpart3 <- rpart(author ~  there + to + by + should,
          data=train,
          method="class",
```

```
          control=rpart.control(cp=0.002,minsplit=2))
rpart.plot(train_rpart3)
```



Calculating correct classification rate:

```
#first model
ccr<-predict(train_rpart, type="class") # Model Predictions
sum(ccr != train$author)/nrow(train) # CCR

## [1] 0

# the CCR of 0 for our first model shows that the model is 100% accurate.

#second model
ccr<-predict(train_rpart2, type="class") # Model Predictions
sum(ccr != train$author)/nrow(train) # CCR

## [1] 0.0212766
```

```
#the 0 shows that this model is also 100% correct

#third model
ccr<-predict(train_rpart3, type="class") # Model Predictions
sum(ccr != train$author)/nrow(train) # CCR

## [1] 0

#this model is also 100% correct.
```

Prediction

Now we can use our models to predict on our test data:

Model 1 Prediction(train_rpart)

```
test1<-predict(train_rpart, test, type='class')
#confusion matrix
table(authorship=test1, true=test$author)

##           true
## authorship Hamilton Madison
##   Hamilton       15       0
##   Madison         0       4
```

Our first model is correct the majority of the time, so we can use this on the disputed papers.

```
dispt1<-predict(train_rpart, unknown, type='class')
table(authorship=dispt1, true=unknown$author)

##           true
## authorship dispt
##   Hamilton     2
##   Madison      9
```

This model predicted that all of the disputed papers were written by Madison. Now, let's check our other two models to see what result they give us.

Model 2 Prediction (train_rpart2)

```
test2<-predict(train_rpart2, test, type='class')
#confusion matrix
table(authorship=test2, true=test$author)

##           true
## authorship Hamilton Madison
##   Hamilton       14       0
##   Madison         1       4
```

This test was slightly less accurate - 3 papers were misclassified. Let's run predict on our unknown data anyways:

```
dispt2<-predict(train_rpart2, unknown, type='class')
table(authorship=dispt2, true=unknown$author)

##           true
## authorship dispt
##   Hamilton     5
##   Madison      6
```

The algorithm predicted that 3 of the papers are attributed to Hamilton, though the majority of them are attributed to Madison. However, at this point, we should doubt the function of this model since it performed poorly in preliminary testing (this model left out "upon" as a deciding factor).

Model 3 Prediction

```
test3<-predict(train_rpart3, test, type='class')
#confusion matrix
table(authorship=test3, true=test$author)

##           true
## authorship Hamilton Madison
##   Hamilton       14       0
##   Madison         1       4
```

This model performed even worse, as it misclassified 5/24 papers (20.83% error rate)- this model left out upon and on, which were both strong predictors. We can test it on our disputed papers anyways to see the result:

```
dispt3<-predict(train_rpart3, unknown, type='class')
table(authorship=dispt3, true=unknown$author)

##          true
## authorship dispt
##   Hamilton     4
##   Madison      7
```

This model attributed 8 of the papers to Madison, and 3 to Hamilton.

Conclusion

Ultimately, the model 1 (which included all words in the DFM) performed the best with a 95.83% accuracy rate. This model attributed all of the papers to Madison, so I would predict with the most certainty that Madison is the author of the disputed papers. This is also supported by the results of our clustering algorithm, which also predicted that Madison was the author of the papers.

We also created 2 more models that left out the strongest predictor (upon), and then left out the two strongest predictor words (upon and on). These were understandably less accurate, but did provide us with more nodes in our decision trees relating to the other predictor words “there”, “to”, “should”, and “by”. However, we still relied on our first model for its accuracy to tell us the true authorship of the disputed papers.