

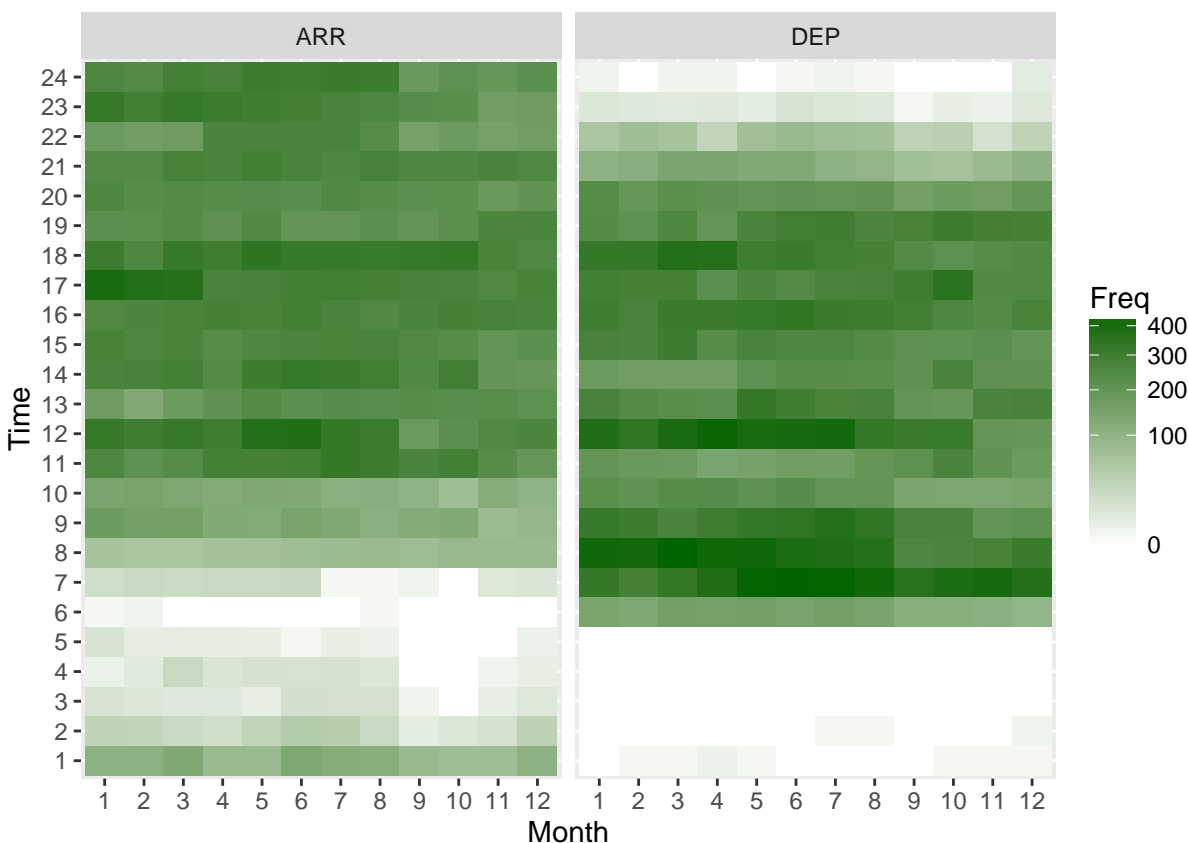
STAT380_Exer2

Anthony Garino, Anqi Huang, Olivia Hong, Yun Guo

Summer 2016

Flights at ABIA

With this very interesting dataset, let's first run some exploratory analysis to figure out which time period witnesses the most air traffic into/out of Austin.



From this heat map, we can clearly see that 7-9 am is the busiest **departure** window throughout the year. However, whether the busiest hour is 7-8 am or 8-9 am varies by month: winter months (if there are any) tend to witness highest volume within the 8-9 am window, whereas relatively warm months (May-Aug.) the highest volume kicks in an hour earlier (7-8 am). This is actually a very interesting insight, one that speaks to the humaneness of airport schedulers: on winter months they give us an extra hour of shut-eye before we are reluctantly whisked to the airport.

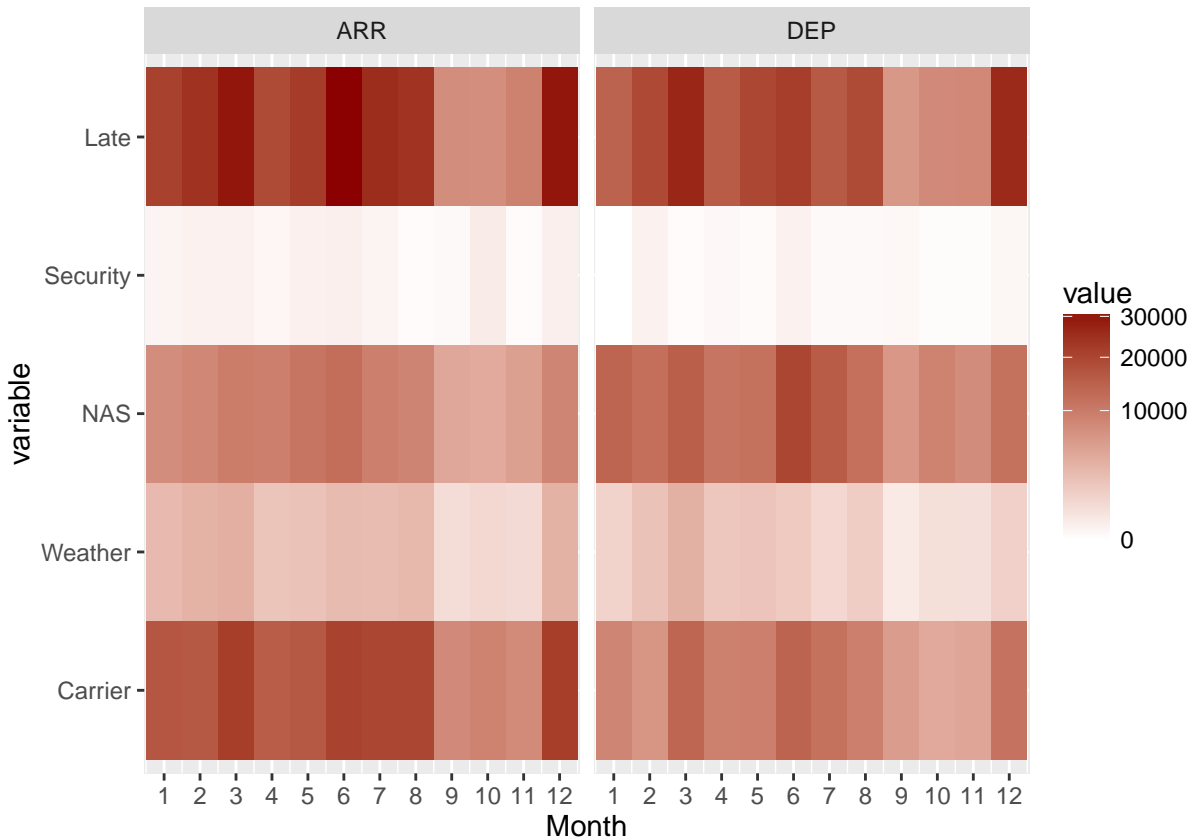
Next, as concerned citizens and taxpayers we naturally wanted to understand what it is that kept on delaying our innocuous flights. Is it security concerns? That universally applicable cop-out (weather)? Or something more wicked (unknown)?

According to the FAA's website, There are five types of delays: Late Arrival Delay ("Late" hereinafter): Arrival delay at an airport due to the late arrival of the same aircraft at a previous airport; Security Delay ("Security"): Evacuation of a terminal or concourse, re-boarding of aircraft because of security breach; NAS Delay ("NAS"): Airport operations, heavy traffic volume, air traffic control reasons; Weather Delay

(“Weather”): Self-evident; Carrier Delay (“Carrier”): Airline logistics reasons, like aircraft cleaning, aircraft damage, bird strike / crew strike...

Next, we seek to plot the incidence of each category by month.

```
##  
## Attaching package: 'dplyr'  
  
## The following objects are masked from 'package:stats':  
##  
##   filter, lag  
  
## The following objects are masked from 'package:base':  
##  
##   intersect, setdiff, setequal, union
```



According to the plot, most of the delays are attributable to late arrival of inbound aircrafts. But there are multiple reasons for this. On closer scrutiny, it appears that this category correlates heavily with the “Carrier” delay group (whenever ‘Carrier’ delays are high, late arrives are more common). This could mean that airlines are more sloppy in certain months and are more prone to aircraft lateness. Note that these months include March and December, the colder months. This could mean that blizzards in northeast are affecting on time departures of flights bound for Austin. Finally, security is not the major reason for delays.

Author Attribution

In this question, we first tried Naive Bayes. This wraps another function around readPlan to read plain text documents in English. And we set the number of author as 50 because there are 50 authors totally.

```
## Loading required package: NLP
```

```
##
```

```
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
```

```
##
```

```
##      annotate
```

```
author_dirs = Sys.glob('./data/ReutersC50/C50train/*')
author_dirs = author_dirs[1:num_authors]
file_list = NULL # This is for all files, including both train/test.
labels = NULL
for(author in author_dirs) {
  author_name = substring(author, first=29)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  labels = append(labels, rep(author_name, length(files_to_add)))
}
test_dirs = Sys.glob('./data/ReutersC50/C50test/*')
test_dirs = test_dirs[1:num_authors]
test_labels = NULL
for(author in test_dirs) {
  author_name = substring(author, first=28)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  test_labels = append(test_labels, rep(author_name, length(files_to_add)))
}
```

Here, we first creat a blank lists of ‘file_list’ and ‘labels’ to store all files including both the training and testing articles later.

Then read in all the file in ‘C50train’ and ‘C50test’. Note: the reason that we read all the files, instead of only the training articles, at the very begining is that we will effeciently avoid any words that only show in the testing set.

once you have documents in a vector, you create a text mining ‘corpus’ with:

```
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

my_corpus = Corpus(VectorSource(all_docs))
names(my_corpus) = file_list
```

Some pre-processing/tokenization steps. tm_map just maps some function to every document in the corpus

```

# Preprocessing
my_corpus = tm_map(my_corpus, content_transformer(tolower)) # make everything lowercase
my_corpus = tm_map(my_corpus, content_transformer(removeNumbers)) # remove numbers
my_corpus = tm_map(my_corpus, content_transformer(removePunctuation)) # remove punctuation
my_corpus = tm_map(my_corpus, content_transformer(stripWhitespace)) ## remove excess white-space
my_corpus = tm_map(my_corpus, content_transformer(removeWords), stopwords("SMART"))

DTM = DocumentTermMatrix(my_corpus)
DTM = removeSparseTerms(DTM, 0.975)
X=as.matrix(DTM)
X_train = X[1:2500, ]
X_test = X[2501:5000, ]
num_rows = nrow(X_train)
num_cols = ncol(X_train)

```

In this way, we will divide the matrix into training and testing set as we know that the first 2500 (50 50) come from the training set and the later 2500 (50 50) come from testing set.

Following, we calculate every author's multinomial probability vector with the smoothing factor.

```

smooth_count = 1/nrow(X_train)
author_weights <- list()
author_classes = NULL
for (i in 0 : (num_authors-1)) {
  author_train = X_train[(i*50 + 1): ((i+1)*50), ]
  author_weight = colSums(author_train + smooth_count)
  author_weight = author_weight/sum(author_weight)
  author_weights[[i+1]] <- author_weight
  author_classes = append(author_classes, labels[i*50+1])
}

```

Then we evaluate each article in the test set and find the author with maximum log_probs, so it is the evaluate result of each article using the model.

```

evaluate = function(test_instance){
  log_probs <- vector()
  for (i in 1 : num_authors) {
    log_prob = sum(test_instance*log(author_weights[[i]]))
    log_probs[i] <- log_prob
  }
  author_classes[which.max(log_probs)]
}
result = apply(X_test, 1, evaluate)

```

At last, we made a matrix to compare the predicting results from the model and the real answer.

```
print(accuracy)
```

```
## [1] 0.6024
```

So the accuracy of the Naive Bayes model is 60.24%. This is the average accuracy of all the 2500 articles (50 articles from each of the 50 authors).

```
print(confusion_matrix)
```

```
##          test_labels
## result    amuelPerry anEeLyn aneMacartney anLopatka arahDavison
## amuelPerry      32      0      0      0      0
## anEeLyn          0     20      0      0      8
## aneMacartney      0      0     18      0      1
## anLopatka         0      0      0     26      0
## arahDavison       0      8      0      0     27
## arcelMichelson    0      0      0      0      0
## arkBendeich        0      0      0      0      0
## arlPenhaul         0      0      0      0      0
## aronPressman       1      0      0      0      0
## arrenSchuettler    0      0      0      0      0
## artinWolk          2      0      0      0      0
## atriciaCommins     0      0      0      0      0
## atthewBunce        0      0      0      0      0
## avidLawder         0      0      0      0      0
## cottHillis         0      1     14      0      1
## dnaFernandes        0      0      0      0      0
## eatherScoffield    0      0      0      0      0
## eithWeir           0      0      0      0      0
## enjaminKangLim      0      0     10      0      0
## ernardHickey        0      1      0      0      0
## eterHumphrey        0     18      0      0     11
## evinDrawbaugh       2      0      0      0      0
## evinMorrison        0      0      0      0      0
## heresePoletti       6      0      0      0      0
## ichaelConnor        1      0      0      1      0
## ickLouth            0      0      0      0      0
## ierreTran           0      0      0      0      0
## illiamKazer         0      1      2      0      0
## imFarrand           0      0      0      0      0
## imGilchrist         0      0      0      0      1
## imonCowell          0      0      0      0      0
## irstinRidley        0      0      0      0      0
## lanCrosby           0      0      0      0      0
## lexanderSmith       0      0      0      0      0
## obinSidel           0      0      0      0      0
## oddNissen           0      0      0      0      0
## oeOrtiz             0      0      0      0      0
## ogerFillion         0      0      0      0      0
## ohnMastrini         0      0      0     23      0
## onathanBirt         0      0      0      0      0
## ouroshKarimkhany    1      0      0      0      0
## oWinterbottom       0      0      0      0      0
## radDorfman          0      0      0      0      0
## rahamEarnshaw       0      1      2      0      1
## ricAuchard          5      0      0      0      0
## umikoFujisaki       0      0      0      0      0
## ureDickie           0      0      4      0      0
## ydiaZajc            0      0      0      0      0
## ynnneO'Donnell      0      0      0      0      0
```

##	ynnleyBrowning	0	0	0	0	0
##	test_labels					
##	result	arcelMichelson	arkBendeich	arlPenhaul	aronPressman	
##	amuelPerry	1	0	0	3	
##	anEeLyn	0	0	0	0	
##	aneMacartney	0	1	0	0	
##	anLopatka	0	0	0	0	
##	arahDavison	0	5	0	0	
##	arcelMichelson	30	0	0	0	
##	arkBendeich	0	21	0	0	
##	arlPenhaul	0	0	47	0	
##	aronPressman	0	0	0	42	
##	arrenSchuettler	0	0	0	1	
##	artinWolk	0	0	0	0	
##	atriciaCommins	0	0	0	0	
##	atthewBunce	0	0	2	0	
##	avidLawder	0	0	0	0	
##	cottHillis	0	1	1	0	
##	dnaFernandes	0	0	0	0	
##	eatherScoffield	0	2	0	0	
##	eithWeir	1	0	0	0	
##	enjaminKangLim	0	0	0	0	
##	ernardHickey	0	4	0	0	
##	eterHumphrey	0	0	0	0	
##	evinDrawbaugh	2	0	0	0	
##	evinMorrison	0	8	0	0	
##	heresePoletti	2	0	0	0	
##	ichaelConnor	0	0	0	0	
##	ickLouth	0	0	0	0	
##	ierreTran	3	2	0	0	
##	illiamKazer	0	0	0	2	
##	imFarrand	0	0	0	0	
##	imGilchrist	0	0	0	0	
##	imonCowell	0	0	0	0	
##	irstinRidley	1	2	0	0	
##	lanCrosby	0	0	0	0	
##	lexanderSmith	0	0	0	0	
##	obinSidel	0	0	0	0	
##	oddNissen	0	2	0	0	
##	oeOrtiz	1	1	0	1	
##	ogerFillion	0	0	0	0	
##	ohnMastrini	0	0	0	0	
##	onathanBirt	5	0	0	0	
##	ouroshKarimkhany	0	0	0	0	
##	oWinterbottom	0	0	0	0	
##	radDorfman	0	0	0	1	
##	rahamEarnshaw	0	0	0	0	
##	ricAuchard	0	0	0	0	
##	umikoFujisaki	0	0	0	0	
##	ureDickie	0	1	0	0	
##	ydiaZajc	0	0	0	0	
##	ynneO'Donnell	0	0	0	0	
##	ynnleyBrowning	4	0	0	0	
##	test_labels					

## result	arrenSchuettler	artinWolk	atriciaCommins	atthewBunce
## amuelPerry	0	7	0	0
## anEeLyn	0	0	0	1
## aneMacartney	0	0	0	3
## anLopatka	0	0	0	0
## arahDavison	0	1	0	0
## arcelMichelson	0	0	0	0
## arkBendeich	0	0	0	0
## arlPenhaul	0	0	0	0
## aronPressman	0	0	0	0
## arrenSchuettler	14	0	0	0
## artinWolk	0	25	0	0
## atriciaCommins	0	1	31	0
## atthewBunce	0	0	0	42
## avidLawder	0	0	0	0
## cottHillis	0	1	0	0
## dnaFernandes	0	0	0	0
## eatherScoffield	36	0	0	2
## eithWeir	0	0	0	0
## enjaminKangLim	0	0	0	0
## ernardHickey	0	0	0	0
## eterHumphrey	0	0	0	0
## evinDrawbaugh	0	2	6	0
## evinMorrison	0	0	0	0
## heresePoletti	0	7	1	0
## ichaelConnor	0	0	1	0
## ickLouth	0	0	3	0
## ierreTran	0	0	0	0
## illiamKazer	0	0	0	0
## imFarrand	0	0	0	0
## imGilchrist	0	0	0	0
## imonCowell	0	0	1	0
## irstinRidley	0	0	0	0
## lanCrosby	0	0	0	0
## lexanderSmith	0	0	0	0
## obinSidel	0	0	0	0
## oddNissen	0	2	0	0
## oeOrtiz	0	0	0	0
## ogerFillion	0	0	0	0
## ohnMastrini	0	0	0	0
## onathanBirt	0	0	1	0
## ouroshKarimkhany	0	1	0	0
## oWinterbottom	0	0	0	0
## radDorfman	0	3	4	0
## rahamEarnshaw	0	0	0	0
## ricAuchard	0	0	2	0
## umikoFujisaki	0	0	0	0
## ureDickie	0	0	0	0
## ydiaZajc	0	0	0	0
## ynnneO'Donnell	0	0	0	0
## ynnleyBrowning	0	0	0	2
##	test_labels			
## result	avidLawder	cottHillis	dnaFernandes	eatherScoffield
## amuelPerry	0	0	0	0

##	anEeLyn	0	1	0	0
##	aneMacartney	1	28	0	12
##	anLopatka	0	0	0	1
##	arahDavison	0	0	0	0
##	arcelMichelson	0	0	0	0
##	arkBendeich	0	0	1	4
##	arlPenhaul	1	0	0	0
##	aronPressman	0	0	0	0
##	arrenSchuettler	0	0	0	7
##	artinWolk	2	0	0	0
##	atriciaCommins	0	0	0	0
##	atthewBunce	0	0	0	0
##	avidLawder	7	0	0	0
##	cottHillis	0	13	0	1
##	dnaFernandes	0	0	20	0
##	eatherScoffield	0	0	0	17
##	eithWeir	0	0	1	0
##	enjaminKangLim	0	0	0	0
##	ernardHickey	0	0	0	0
##	eterHumphrey	0	4	0	1
##	evinDrawbaugh	3	0	0	0
##	evinMorrison	0	0	0	0
##	heresePoletti	0	0	0	0
##	ichaelConnor	0	0	0	0
##	ickLouth	0	0	0	0
##	ierreTran	0	0	4	0
##	illiamKazer	0	0	0	2
##	imFarrand	0	0	6	0
##	imGilchrist	0	0	0	0
##	imonCowell	0	0	0	0
##	irstinRidley	0	0	5	0
##	lanCrosby	0	0	0	0
##	lexanderSmith	0	0	3	0
##	obinSidel	4	0	0	0
##	oddNissen	23	0	6	0
##	oeOrtiz	1	0	0	0
##	ogerFillion	0	0	0	1
##	ohnMastrini	0	0	0	3
##	onathanBirt	0	0	1	0
##	ouroshKarimkhany	0	0	0	0
##	oWinterbottom	0	0	3	0
##	radDorfman	8	0	0	0
##	rahamEarnshaw	0	1	0	0
##	ricAuchard	0	0	0	0
##	umikoFujisaki	0	0	0	0
##	ureDickie	0	2	0	0
##	ydiaZajc	0	0	0	0
##	ynneO'Donnell	0	1	0	0
##	ynnleyBrowning	0	0	0	1
##	test_labels				
##	result	eithWeir	enjaminKangLim	ernardHickey	eterHumphrey
##	amuelPerry	0	0	0	0
##	anEeLyn	0	0	0	11
##	aneMacartney	0	13	0	1

##	anLopatka	0	0	0	0
##	arahDavison	0	0	4	3
##	arcelMichelson	0	0	0	0
##	arkBendeich	0	0	3	0
##	arlPenhaul	0	0	0	0
##	aronPressman	0	0	0	0
##	arrenSchuettler	0	0	0	0
##	artinWolk	0	0	0	0
##	atriciaCommins	0	0	0	0
##	atthewBunce	0	0	0	0
##	avidLawder	0	0	0	0
##	cottHillis	0	7	0	1
##	dnaFernandes	5	0	0	0
##	eatherScoffield	0	0	3	0
##	eithWeir	37	0	0	0
##	enjaminKangLim	0	11	0	0
##	ernardHickey	0	0	27	0
##	eterHumphrey	0	3	0	33
##	evinDrawbaugh	0	0	0	0
##	evinMorrison	0	0	5	0
##	heresePoletti	0	0	0	0
##	ichaelConnor	0	0	2	0
##	ickLouth	0	0	0	0
##	ierreTran	0	0	0	0
##	illiamKazer	0	11	0	0
##	imFarrand	0	0	2	0
##	imGilchrist	0	0	0	1
##	imonCowell	0	0	0	0
##	irstinRidley	5	0	0	0
##	lanCrosby	0	0	0	0
##	lexanderSmith	0	0	2	0
##	obinSidel	0	0	0	0
##	oddNissen	0	0	0	0
##	oeOrtiz	0	0	1	0
##	ogerFillion	0	0	0	0
##	ohnMastrini	0	0	0	0
##	onathanBirt	1	0	0	0
##	ouroshKarimkhany	0	0	0	0
##	oWinterbottom	1	0	0	0
##	radDorfman	0	0	0	0
##	rahamEarnshaw	0	2	0	0
##	ricAuchard	0	0	0	0
##	umikoFujisaki	0	0	0	0
##	ureDickie	1	3	0	0
##	ydiaZajc	0	0	0	0
##	ynneO'Donnell	0	0	0	0
##	ynnleyBrowning	0	0	1	0
##	test_labels				
##	result	evinDrawbaugh evinMorrison heresePoletti ichaelConnor			
##	amuelPerry	4	0	5	0
##	anEeLyn	0	0	0	0
##	aneMacartney	0	2	0	0
##	anLopatka	0	0	0	0
##	arahDavison	0	2	0	0

##	arcelMichelson	0	0	0	0	
##	arkBendeich	0	4	0	0	
##	arlPenhaul	0	0	0	0	
##	aronPressman	0	0	4	0	
##	arrenSchuettler	0	0	0	0	
##	artinWolk	0	0	1	0	
##	atriciaCommins	10	0	0	1	
##	atthewBunce	0	0	0	0	
##	avidLawder	0	0	0	0	
##	cottHillis	0	0	0	0	
##	dnaFernandes	0	0	0	0	
##	eatherScoffield	0	0	0	0	
##	eithWeir	1	1	0	2	
##	enjaminKangLim	0	0	0	0	
##	ernardHickey	0	6	0	0	
##	eterHumphrey	0	0	0	0	
##	evinDrawbaugh	26	0	4	1	
##	evinMorrison	0	28	0	0	
##	heresePoletti	1	0	26	1	
##	ichaelConnor	2	0	0	39	
##	ickLouth	0	1	0	0	
##	ierreTran	0	0	0	0	
##	illiamKazer	0	3	0	0	
##	imFarrand	0	1	0	0	
##	imGilchrist	0	0	0	0	
##	imonCowell	0	0	0	0	
##	irstinRidley	0	2	0	0	
##	lanCrosby	0	0	0	0	
##	lexanderSmith	0	0	0	1	
##	obinSidel	0	0	0	0	
##	oddNissen	0	0	0	1	
##	oeOrtiz	0	0	0	4	
##	ogerFillion	0	0	0	0	
##	ohnMastrini	0	0	0	0	
##	onathanBirt	0	0	0	0	
##	ouroshKarimkhany	0	0	1	0	
##	oWinterbottom	0	0	0	0	
##	radDorfman	6	0	1	0	
##	rahamEarnshaw	0	0	0	0	
##	ricAuchard	0	0	8	0	
##	umikoFujisaki	0	0	0	0	
##	ureDickie	0	0	0	0	
##	ydiaZajc	0	0	0	0	
##	ynneO'Donnell	0	0	0	0	
##	ynnleyBrowning	0	0	0	0	
##	test_labels					
##	result	ickLouth	ierreTran	illiamKazer	imFarrand	imGilchrist
##	amuelPerry	1	0	0	0	0
##	anEeLyn	0	0	4	0	0
##	aneMacartney	0	0	9	0	0
##	anLopatka	0	0	0	0	0
##	arahDavison	0	0	1	0	0
##	arcelMichelson	0	13	0	0	0
##	arkBendeich	0	0	0	0	0

##	arlPenhaul	0	0	0	0	0
##	aronPressman	0	0	0	0	0
##	arrenSchuettler	0	0	0	0	0
##	artinWolk	0	0	0	0	0
##	atriciaCommins	1	0	0	0	0
##	atthewBunce	0	0	0	0	0
##	avidLawder	0	0	0	0	0
##	cottHillis	0	0	5	0	0
##	dnaFernandes	0	1	0	2	0
##	eatherScofield	0	0	0	0	0
##	eithWeir	0	0	0	2	0
##	enjaminKangLim	0	0	4	0	0
##	ernardHickey	0	0	0	0	0
##	eterHumphrey	0	0	1	0	0
##	evinDrawbaugh	0	0	0	0	0
##	evinMorrison	0	0	0	0	0
##	heresePoletti	0	0	0	0	0
##	ichaelConnor	0	0	0	1	0
##	ickLouth	41	0	0	0	0
##	ierreTran	0	34	0	0	0
##	illiamKazer	0	0	17	0	0
##	imFarrand	0	1	0	39	0
##	imGilchrist	0	0	0	0	50
##	imonCowell	0	0	0	5	0
##	irstinRidley	2	0	0	0	0
##	lanCrosby	0	0	0	0	0
##	lexanderSmith	0	0	0	0	0
##	obinSidel	0	0	0	0	0
##	oddNissen	0	0	0	0	0
##	oeOrtiz	0	0	0	0	0
##	ogerFillion	1	0	0	0	0
##	ohnMastrini	0	0	0	0	0
##	onathanBirt	0	1	0	1	0
##	ouroshKarimkhany	0	0	0	0	0
##	oWinterbottom	0	0	0	0	0
##	radDorfman	1	0	0	0	0
##	rahamEarnshaw	0	0	4	0	0
##	ricAuchard	3	0	0	0	0
##	umikoFujisaki	0	0	2	0	0
##	ureDickie	0	0	2	0	0
##	ydiaZajc	0	0	0	0	0
##	ynneO'Donnell	0	0	1	0	0
##	ynnleyBrowning	0	0	0	0	0
##	test_labels					
##	result	imonCowell	irstinRidley	lanCrosby	lexanderSmith	
##	amuelPerry	0	0	0	1	
##	anEeLyn	0	0	0	0	
##	aneMacartney	0	0	0	0	
##	anLopatka	0	0	8	0	
##	arahDavison	0	1	0	0	
##	arcelMichelson	0	0	0	0	
##	arkBendeich	0	0	0	0	
##	arlPenhaul	0	0	0	0	
##	aronPressman	0	2	0	0	

##	arrenSchuettler	0	0	0	0	
##	artinWolk	0	0	0	0	
##	atriciaCommins	0	0	0	0	
##	atthewBunce	0	0	0	0	
##	avidLawder	0	0	0	0	
##	cottHillis	0	0	0	0	
##	dnaFernandes	0	0	0	0	
##	eatherScoffield	0	0	0	0	
##	eithWeir	0	7	0	0	
##	enjaminKangLim	0	0	0	0	
##	ernardHickey	0	0	0	0	
##	eterHumphrey	0	0	0	0	
##	evinDrawbaugh	0	0	0	0	
##	evinMorrison	0	0	0	0	
##	heresePoletti	0	0	0	0	
##	ichaelConnor	0	0	0	0	
##	ickLouth	0	0	0	0	
##	ierreTran	0	0	0	0	
##	illiamKazer	0	0	0	0	
##	imFarrand	1	1	0	0	
##	imGilchrist	0	0	0	0	
##	imonCowell	28	0	0	0	
##	irstinRidley	0	34	0	0	
##	lanCrosby	0	0	25	0	
##	lexanderSmith	16	0	0	20	
##	obinSidel	0	0	0	0	
##	oddNissen	0	0	0	0	
##	oeOrtiz	4	1	0	28	
##	ogerFillion	0	0	0	0	
##	ohnMastrini	0	0	17	0	
##	onathanBirt	0	2	0	1	
##	ouroshKarimkhany	0	0	0	0	
##	oWinterbottom	0	2	0	0	
##	radDorfman	0	0	0	0	
##	rahamEarnshaw	0	0	0	0	
##	ricAuchard	0	0	0	0	
##	umikoFujisaki	1	0	0	0	
##	ureDickie	0	0	0	0	
##	ydiaZajc	0	0	0	0	
##	ynneO'Donnell	0	0	0	0	
##	ynnleyBrowning	0	0	0	0	
##	test_labels					
##	result	obinSidel	oddNissen	oeOrtiz	ogerFillion	ohnMastrini
##	amuelPerry	0	1	0	1	0
##	anEeLyn	0	0	0	0	0
##	aneMacartney	0	1	0	1	0
##	anLopatka	0	0	0	0	11
##	arahDavison	0	1	1	0	0
##	arcelMichelson	1	0	0	0	0
##	arkBendeich	0	0	0	0	0
##	arlPenhaul	0	7	0	0	2
##	aronPressman	0	0	0	2	0
##	arrenSchuettler	0	0	0	0	0
##	artinWolk	1	0	0	0	0

##	atriciaCommins	0	0	0	0	0
##	atthewBunce	0	0	0	0	0
##	avidLawder	0	13	0	0	0
##	cottHillis	0	0	0	2	1
##	dnaFernandes	0	0	0	0	0
##	eatherScofield	0	0	0	0	0
##	eithWeir	0	0	0	2	0
##	enjaminKangLim	0	0	0	0	0
##	ernardHickey	0	0	0	0	0
##	eterHumphrey	0	0	0	0	1
##	evinDrawbaugh	1	3	0	0	0
##	evinMorrison	1	0	0	0	0
##	heresePoletti	0	0	0	0	0
##	ichaelConnor	1	1	0	2	0
##	ickLouth	0	0	0	2	0
##	ierreTran	0	0	0	0	1
##	illiamKazer	0	0	0	0	0
##	imFarrand	0	0	1	0	0
##	imGilchrist	0	1	0	0	0
##	imonCowell	0	0	4	0	0
##	irstinRidley	0	0	0	0	0
##	lanCrosby	0	0	0	0	1
##	lexanderSmith	0	0	5	0	0
##	obinSidel	40	0	0	0	0
##	oddNissen	0	20	0	0	1
##	oeOrtiz	0	0	39	0	0
##	ogerFillion	0	0	0	38	0
##	ohnMastrini	0	0	0	0	32
##	onathanBirt	2	0	0	0	0
##	ouroshKarimkhany	0	0	0	0	0
##	oWinterbottom	0	0	0	0	0
##	radDorfman	2	2	0	0	0
##	rahamEarnshaw	0	0	0	0	0
##	ricAuchard	1	0	0	0	0
##	umikoFujisaki	0	0	0	0	0
##	ureDickie	0	0	0	0	0
##	ydiaZajc	0	0	0	0	0
##	ynneO'Donnell	0	0	0	0	0
##	ynnleyBrowning	0	0	0	0	0
##	test_labels					
##	result	onathanBirt	ouroshKarimkhany	oWinterbottom	radDorfman	
##	amuelPerry	0	1	0	1	
##	anEeLyn	0	0	0	0	
##	aneMacartney	0	0	0	0	
##	anLopatka	0	0	0	0	
##	arahDavison	0	0	0	0	
##	arcelMichelson	0	0	0	0	
##	arkBendeich	0	0	0	0	
##	arlPenhaul	0	0	0	0	
##	aronPressman	0	0	0	0	
##	arrenSchuettler	0	0	0	0	
##	artinWolk	0	0	0	1	
##	atriciaCommins	0	1	0	0	
##	atthewBunce	0	0	0	0	

##	avidLawder	0	0	0	0
##	cottHillis	0	0	0	0
##	dnaFernandes	1	0	3	0
##	eatherScoffield	0	0	0	0
##	eithWeir	1	0	0	0
##	enjaminKangLim	0	0	0	0
##	ernardHickey	0	0	0	0
##	eterHumphrey	0	0	0	0
##	evinDrawbaugh	0	1	0	1
##	evinMorrison	0	0	0	0
##	heresePoletti	0	4	0	0
##	ichaelConnor	0	0	0	0
##	ickLouth	0	1	0	0
##	ierreTran	0	0	0	0
##	illiamKazer	0	0	0	0
##	imFarrand	3	0	8	0
##	imGilchrist	0	0	0	0
##	imonCowell	2	0	0	0
##	irstinRidley	1	0	0	0
##	lanCrosby	0	0	0	0
##	lexanderSmith	2	0	3	0
##	obinSidel	0	0	0	1
##	oddNissen	0	0	0	1
##	oeOrtiz	0	0	1	0
##	ogerFillion	0	0	0	0
##	ohnMastrini	0	0	0	0
##	onathanBirt	37	0	2	0
##	ouroshKarimkhany	0	32	0	0
##	oWinterbottom	2	0	33	0
##	radDorfman	0	0	0	43
##	rahamEarnshaw	0	0	0	0
##	ricAuchard	0	10	0	2
##	umikoFujisaki	0	0	0	0
##	ureDickie	0	0	0	0
##	ydiaZajc	0	0	0	0
##	ynneO'Donnell	0	0	0	0
##	ynnleyBrowning	1	0	0	0
##	test_labels				
##	result	rahamEarnshaw ricAuchard umikoFujisaki ureDickie			
##	amuelPerry	0	3	0	0
##	anEeLyn	0	0	0	1
##	aneMacartney	3	0	0	9
##	anLopatka	0	0	0	0
##	arahDavison	4	0	1	0
##	arcelMichelson	0	0	0	0
##	arkBendeich	0	0	0	0
##	arlPenhaul	0	0	0	0
##	aronPressman	0	0	0	0
##	arrenSchuettler	0	0	0	0
##	artinWolk	0	1	0	0
##	atriciaCommins	0	0	0	0
##	atthewBunce	0	0	0	0
##	avidLawder	0	0	0	0
##	cottHillis	1	0	0	8

##	dnaFernandes	0	0	0	0
##	eatherScoffield	0	0	0	0
##	eithWeir	0	0	0	0
##	enjaminKangLim	0	0	0	1
##	ernardHickey	0	0	0	0
##	eterHumphrey	0	0	0	3
##	evinDrawbaugh	0	3	0	0
##	evinMorrison	0	0	0	0
##	heresePoletti	0	2	0	0
##	ichaelConnor	0	0	0	0
##	ickLouth	0	10	0	0
##	ierreTran	0	0	0	1
##	illiamKazer	2	0	0	8
##	imFarrand	0	0	0	0
##	imGilchrist	1	0	0	0
##	imonCowell	0	0	0	0
##	irstinRidley	0	2	0	0
##	lanCrosby	0	0	0	0
##	lexanderSmith	0	0	0	0
##	obinSidel	0	1	0	0
##	oddNissen	0	0	0	0
##	oeOrtiz	0	0	0	0
##	ogerFillion	0	1	0	0
##	ohnMastrini	0	0	0	0
##	onathanBirt	0	0	0	0
##	ouroshKarimkhany	0	1	0	0
##	oWinterbottom	0	0	0	0
##	radDorfman	0	1	0	0
##	rahamEarnshaw	39	0	0	4
##	ricAuchard	0	25	0	0
##	umikoFujisaki	0	0	49	0
##	ureDickie	0	0	0	13
##	ydiaZajc	0	0	0	0
##	ynneO'Donnell	0	0	0	1
##	ynnleyBrowning	0	0	0	1
##	test_labels				
##	result	ydiaZajc	ynneO'Donnell	ynnleyBrowning	
##	amuelPerry	2	0	0	
##	anEeLyn	0	1	0	
##	aneMacartney	0	4	1	
##	anLopatka	0	0	0	
##	arahDavison	1	0	0	
##	arcelMichelson	0	0	0	
##	arkBendeich	0	0	0	
##	arlPenhaul	0	0	0	
##	aronPressman	0	0	0	
##	arrenSchuettler	5	0	0	
##	artinWolk	0	0	0	
##	atriciaCommins	0	0	0	
##	atthewBunce	0	0	0	
##	avidLawder	0	0	0	
##	cottHillis	0	0	0	
##	dnaFernandes	0	0	0	
##	eatherScoffield	1	0	0	

##	eithWeir	0	0	0
##	enjaminKangLim	0	0	0
##	ernardHickey	0	0	0
##	eterHumphrey	0	0	0
##	evinDrawbaugh	2	0	0
##	evinMorrison	0	0	0
##	heresePoletti	0	0	0
##	ichaelConnor	1	0	0
##	ickLouth	0	0	0
##	ierreTran	0	0	0
##	illiamKazer	0	2	0
##	imFarrand	0	0	0
##	imGilchrist	0	0	0
##	imonCowell	0	0	0
##	irstinRidley	0	0	0
##	lanCrosby	0	0	0
##	lexanderSmith	0	0	0
##	obinSidel	0	0	0
##	oddNissen	0	1	0
##	oeOrtiz	0	0	0
##	ogerFillion	0	0	0
##	ohnMastrini	0	0	0
##	onathanBirt	0	0	0
##	ouroshKarimkhany	0	0	0
##	oWinterbottom	0	0	0
##	radDorfman	5	0	0
##	rahamEarnshaw	1	1	0
##	ricAuchard	1	0	0
##	umikoFujisaki	0	0	0
##	ureDickie	0	2	0
##	ydiaZajc	31	0	0
##	ynneO'Donnell	0	39	0
##	ynnleyBrowning	0	0	49

From the result from the confusion_matrix, we find that the articles from AlanCrosby are easily predicted as from JohnMastrini. AlexanderSmith's articles are predicted to be from JoeOrtiz. And DarrenSchuettler's work is more likely to be predicted as HeatherSchofield's. And the same situation happens to ScottHillis's articles as they are most predicted to be JaneMacartney. What is more, TanEeLyn's works are mostly equally predicted to from TanEeLyn and PeterHumphrey.

We now use a slightly more complex model (RandomForest) to predict authorship. The code can be reviewed in source file. Below is the confusion matrix output for this model.

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
## combine
```



```
## The following object is masked from 'package:ggplot2':
```

```
##
```

```
##     margin
```

```
## Loading required package: lattice
```

```
## Confusion Matrix and Statistics
```

```
##
```

```
##           Reference
```

```
## Prediction  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##           1 47  0  0  0  0  1  0  0  0  1  0  0  0  0  0  0  1  0  0  0
##           2  0 31  0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  3  0  0
##           3  0  0 15  0  0  0  0  2  0  0  0  0  0  0  0  0  1  0  1  0
##           4  0  0  0 22  0  0  0  0  0  0  1  0  0  0 22  0  0  0  0  0
##           5  0  0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##           6  0  0  0  0  0 33  0  6  0  1  0  0  0  0  0  0  0  0  0  0
##           7  0  0  0  0  0  0 14  0  0  0  0  0 25  0  0  0  0  0  0  0
##           8  0  0  0  0  0  0  0  7  1  0  0  0  0  0  0  0  0  0  0  0
##           9  0  0  1  0  0  0  0  0 18  0  0  0  0  0  0  0  4  0  2  0
##          10  0  0  0  0  0  0  0  0  0 15  0  0  0  0  0  0  0  0  0  0
##          11  0  0  0  0  0  1  0  0  0  1 49  0  0  0  0  0  0  0  0  0
##          12  0  0  0  2  0  0  0  0  0  0  0 48  0  0  3  0  0  0  0  0
##          13  0  0  1  0  1  0 34  0  0  0  0  0 19  0  0  0  0  0  0  0
##          14  0  2  0  0  0  0  0  0  0  0  0  0 28  0  0  0  0 13  0  0
##          15  0  0  0  5  0  0  0  0  0  0  0  0  0  8  0  0  0  0  0  0
##          16  0  0  0  0  0  0  0  0  2  0  0  1  0  0  0 50  0  0  0  0
##          17  0  0  1  0  0  0  0  0  5  0  0  0  0  0  0  0 42  0  0  1
##          18  0  0 28  0  0  0  0  0  0  0  0  0  0  0  0  0  2 32  0  0
##          19  0 16  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0 32  0  0
##          20  0  0  1  0  0  0  0  0  3  0  0  0  0  0  0  2  2 35  0  0
##          21  0  0  0  0  0  0  0  1  0  0  0  0  2  0  0  0  0  0  0 46
##          22  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0
##          23  0  0  0  0  0  8  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          24  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          25  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0
##          26  1  0  0  0  0  0  0  0  0  6  0  0  0  0  0  0  0  0  0  0
##          27  0  0  0  0  0  0  2  0  0  0  0  0  0  0  0  0  0  0  0  0
##          28  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          29  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  2  1  0
##          30  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0
##          31  0  0  0  0  9  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0
##          32  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          33  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4
##          34  1  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  0  0  0
##          35  0  0  0  1  0  0  0  0  0  0  0  0  0  0  3  0  0  0  0  0
##          36  0  0  0  0  0  0  0  0  0 12  0  0  0  0  0  0  0  0  0  0
##          37  0  0  0  0  0  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          38  0  0  0  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##          39  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0
##          40  0  0  0  0  0  2  0  1  3  0  1  0  0  0  0  0  0  0  0  0
##          41  0  0  0  0  0  0  0  0  1  4  0  0  0  0  0  0  0  0  0  0
##          42  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0  0  0  0  0  0
##          43  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  3  0  0
##          44  0  0  0  3  0  0  0  0  0  0  0  0  0  0 10  0  0  0  0  0
```

##	45	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	1	0	
##	46	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
##	47	1	0	1	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	
##	48	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	3	0	0	8	
##	49	0	0	0	0	0	2	0	34	8	0	0	0	0	0	0	0	0	0	0	0	
##	50	0	0	0	9	0	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	
##	Reference																					
##	Prediction	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
##	1	0	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	4	3	
##	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	4	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0	0	
##	5	0	0	12	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	
##	6	0	5	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	
##	7	0	1	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
##	10	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
##	11	0	1	1	1	0	0	1	0	0	0	2	0	0	1	0	0	0	0	0	0	
##	12	0	0	0	0	0	0	8	0	0	1	0	0	0	2	0	0	0	0	0	0	
##	13	0	0	0	1	0	0	0	0	0	3	0	1	0	1	0	0	0	0	0	0	
##	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	2	
##	17	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	18	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	20	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
##	21	1	2	0	0	0	0	0	1	0	0	1	2	4	0	0	0	0	0	0	1	
##	22	39	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	23	0	30	0	0	0	0	0	0	0	0	1	0	6	0	0	2	0	0	0	1	
##	24	0	0	28	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	
##	25	1	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	26	0	0	0	2	45	1	0	0	0	0	4	0	1	0	1	2	0	0	0	19	
##	27	0	0	0	0	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	28	0	0	0	0	0	0	39	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	29	0	0	0	0	0	0	0	49	0	0	0	0	0	0	0	0	0	0	0	0	
##	30	0	0	0	1	0	0	0	0	47	0	0	2	0	0	0	2	0	23	1	0	
##	31	3	0	4	0	0	1	0	0	0	40	0	0	0	0	1	1	0	0	0	0	
##	32	0	0	0	0	0	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	
##	33	0	0	0	0	0	0	0	0	0	0	0	45	1	0	0	0	0	0	0	0	
##	34	2	1	0	0	0	0	0	0	0	0	2	0	31	0	1	1	0	0	2	0	
##	35	0	0	0	0	0	0	1	0	0	1	0	0	0	24	0	0	0	0	0	3	
##	36	0	0	2	4	1	0	0	0	0	0	1	0	0	0	43	0	0	0	0	1	
##	37	0	3	0	0	0	0	0	0	0	0	1	0	3	0	0	38	0	0	0	0	
##	38	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	39	0	0	0	
##	39	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	25	0	0	
##	40	0	6	0	0	1	0	0	0	0	0	0	0	0	1	3	0	0	45	1	0	
##	41	0	0	1	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	39	0	
##	42	0	0	0	0	0	0	0	0	1	0	8	0	0	0	1	0	0	0	0	21	
##	43	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
##	44	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	
##	45	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
##	46	0	0	0	1	0	0	0	0	0	0	0	0	0	4	0	0	8	0	0	0	

```

##      47  0  0  0  0  2  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  5
##      48  1  1  0  1  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
##      49  0  0  0  0  0  0  1  0  0  0  1  0  2  0  0  0  0  0  0  0  0
##      50  0  0  0  0  0  0  0  0  0  0  0  0  0  8  0  0  1  0  0  0  0
##
##      Reference
## Prediction 43 44 45 46 47 48 49 50
##      1    0  0  1  0  4  0  1  0
##      2    0  0  0  0  0  0  0  0
##      3    0  0  0  0  0  0  0  0
##      4    1 22  0  0  0  0  0  9
##      5    0  0  0  0  0  0  0  0
##      6    0  0  0  0  0  0  1  0
##      7    0  0  0  0  0  0  0  0
##      8    0  0  0  0  0  0 10  0
##      9    0  0  0  0  0  7  0  0
##     10    0  0  0  0  5  0  0  0
##     11    0  0  0  0  0  0  0  0
##     12    0  0  0  0  0  0  0  3
##     13    0  0  0  0  0  0  0  0
##     14    0  0  0  0  0  0  0  0
##     15    0  2  0  0  0  0  0  3
##     16    1  0  0  0  0  0  2  1
##     17    0  0  0  0  0  0  0  0
##     18    0  0 12  0  0  1  0  0
##     19    0  0  0  0  0  0  0  0
##     20    0  0  0  0  0  1  0  0
##     21    0  1  0  0  0  0  3  0
##     22    0  0  0  0  0  1  0  0
##     23    0  0  0  0  2  0  0  0
##     24    0  0  0  0  0  0  0  0
##     25    0  0  0  0  0  0  0  0
##     26    0  0  0  0  8  1  0  1
##     27    0  0  0  0  0  0  0  0
##     28    0  1  0  0  0  0  0  1
##     29    0  0  0  0  0  0  0  0
##     30    0  0  0  0  0  0  0  0
##     31    0  0  0  0  0  1  0  0
##     32    0  0  0  0  0  0  0  0
##     33    0  0  0  0  0  0  0  0
##     34    0  0  0  0  1  1  3  0
##     35    0  6  0  0  0  0  0  4
##     36    0  0  0  0  4  0  0  0
##     37    0  0  0  0  3  0  1  0
##     38 12  4  0 17  0  0  0  2
##     39    0  0  0  0  0  0  0  0
##     40    0  0  0  0  1  0  0  0
##     41    0  0  0  0  0  0  0  0
##     42    0  0  0  0  1  0  0  0
##     43 22  0  0  9  0  0  0  0
##     44    0  9  0  0  0  0  0  5
##     45    0  0 36  0  0  3  0  0
##     46 14  1  0 24  0  0  0  4
##     47    0  0  0  0 21  0  0  0
##     48    0  0  1  0  0 34  0  0

```

```

##          49  0  0  0  0  0  0  29  0
##          50  0  4  0  0  0  0  0  17
##
## Overall Statistics
##
##          Accuracy : 0.6216
##          95% CI : (0.6023, 0.6407)
##          No Information Rate : 0.02
##          P-Value [Acc > NIR] : < 2.2e-16
##
##          Kappa : 0.6139
##          McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##          Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
## Sensitivity      0.9400    0.6200    0.3000    0.4400    0.5200    0.6600
## Specificity      0.9922    0.9971    0.9984    0.9763    0.9943    0.9935
## Pos Pred Value   0.7121    0.8158    0.7895    0.2750    0.6500    0.6735
## Neg Pred Value   0.9988    0.9923    0.9859    0.9884    0.9902    0.9931
## Prevalence       0.0200    0.0200    0.0200    0.0200    0.0200    0.0200
## Detection Rate   0.0188    0.0124    0.0060    0.0088    0.0104    0.0132
## Detection Prevalence 0.0264    0.0152    0.0076    0.0320    0.0160    0.0196
## Balanced Accuracy 0.9661    0.8086    0.6492    0.7082    0.7571    0.8267
##
##          Class: 7 Class: 8 Class: 9 Class: 10 Class: 11
## Sensitivity      0.2800    0.1400    0.3600    0.3000    0.9800
## Specificity      0.9824    0.9955    0.9939    0.9971    0.9963
## Pos Pred Value   0.2456    0.3889    0.5455    0.6818    0.8448
## Neg Pred Value   0.9853    0.9827    0.9870    0.9859    0.9996
## Prevalence       0.0200    0.0200    0.0200    0.0200    0.0200
## Detection Rate   0.0056    0.0028    0.0072    0.0060    0.0196
## Detection Prevalence 0.0228    0.0072    0.0132    0.0088    0.0232
## Balanced Accuracy 0.6312    0.5678    0.6769    0.6486    0.9882
##
##          Class: 12 Class: 13 Class: 14 Class: 15 Class: 16
## Sensitivity      0.9600    0.3800    0.5600    0.1600    1.0000
## Specificity      0.9922    0.9829    0.9939    0.9959    0.9955
## Pos Pred Value   0.7164    0.3115    0.6512    0.4444    0.8197
## Neg Pred Value   0.9992    0.9873    0.9910    0.9831    1.0000
## Prevalence       0.0200    0.0200    0.0200    0.0200    0.0200
## Detection Rate   0.0192    0.0076    0.0112    0.0032    0.0200
## Detection Prevalence 0.0268    0.0244    0.0172    0.0072    0.0244
## Balanced Accuracy 0.9761    0.6814    0.7769    0.5780    0.9978
##
##          Class: 17 Class: 18 Class: 19 Class: 20 Class: 21
## Sensitivity      0.8400    0.6400    0.6400    0.7000    0.9200
## Specificity      0.9967    0.9820    0.9865    0.9943    0.9922
## Pos Pred Value   0.8400    0.4211    0.4923    0.7143    0.7077
## Neg Pred Value   0.9967    0.9926    0.9926    0.9939    0.9984
## Prevalence       0.0200    0.0200    0.0200    0.0200    0.0200
## Detection Rate   0.0168    0.0128    0.0128    0.0140    0.0184
## Detection Prevalence 0.0200    0.0304    0.0260    0.0196    0.0260
## Balanced Accuracy 0.9184    0.8110    0.8133    0.8471    0.9561
##
##          Class: 22 Class: 23 Class: 24 Class: 25 Class: 26
## Sensitivity      0.7800    0.6000    0.5600    0.5000    0.9000
## Specificity      0.9955    0.9918    0.9927    0.9992    0.9808

```

## Pos Pred Value	0.7800	0.6000	0.6087	0.9259	0.4891
## Neg Pred Value	0.9955	0.9918	0.9910	0.9899	0.9979
## Prevalence	0.0200	0.0200	0.0200	0.0200	0.0200
## Detection Rate	0.0156	0.0120	0.0112	0.0100	0.0180
## Detection Prevalence	0.0200	0.0200	0.0184	0.0108	0.0368
## Balanced Accuracy	0.8878	0.7959	0.7763	0.7496	0.9404
##	Class: 27	Class: 28	Class: 29	Class: 30	Class: 31
## Sensitivity	0.6200	0.7800	0.9800	0.9400	0.8000
## Specificity	0.9992	0.9988	0.9984	0.9878	0.9914
## Pos Pred Value	0.9394	0.9286	0.9245	0.6104	0.6557
## Neg Pred Value	0.9923	0.9955	0.9996	0.9988	0.9959
## Prevalence	0.0200	0.0200	0.0200	0.0200	0.0200
## Detection Rate	0.0124	0.0156	0.0196	0.0188	0.0160
## Detection Prevalence	0.0132	0.0168	0.0212	0.0308	0.0244
## Balanced Accuracy	0.8096	0.8894	0.9892	0.9639	0.8957
##	Class: 32	Class: 33	Class: 34	Class: 35	Class: 36
## Sensitivity	0.4400	0.9000	0.6200	0.4800	0.8600
## Specificity	1.0000	0.9976	0.9931	0.9922	0.9898
## Pos Pred Value	1.0000	0.8824	0.6458	0.5581	0.6324
## Neg Pred Value	0.9887	0.9980	0.9923	0.9894	0.9971
## Prevalence	0.0200	0.0200	0.0200	0.0200	0.0200
## Detection Rate	0.0088	0.0180	0.0124	0.0096	0.0172
## Detection Prevalence	0.0088	0.0204	0.0192	0.0172	0.0272
## Balanced Accuracy	0.7200	0.9488	0.8065	0.7361	0.9249
##	Class: 37	Class: 38	Class: 39	Class: 40	Class: 41
## Sensitivity	0.7600	0.7800	0.5000	0.9000	0.7800
## Specificity	0.9943	0.9833	0.9984	0.9918	0.9959
## Pos Pred Value	0.7308	0.4875	0.8621	0.6923	0.7959
## Neg Pred Value	0.9951	0.9955	0.9899	0.9979	0.9955
## Prevalence	0.0200	0.0200	0.0200	0.0200	0.0200
## Detection Rate	0.0152	0.0156	0.0100	0.0180	0.0156
## Detection Prevalence	0.0208	0.0320	0.0116	0.0260	0.0196
## Balanced Accuracy	0.8771	0.8816	0.7492	0.9459	0.8880
##	Class: 42	Class: 43	Class: 44	Class: 45	Class: 46
## Sensitivity	0.4200	0.4400	0.1800	0.7200	0.4800
## Specificity	0.9939	0.9939	0.9906	0.9939	0.9853
## Pos Pred Value	0.5833	0.5946	0.2812	0.7059	0.4000
## Neg Pred Value	0.9882	0.9886	0.9834	0.9943	0.9893
## Prevalence	0.0200	0.0200	0.0200	0.0200	0.0200
## Detection Rate	0.0084	0.0088	0.0036	0.0144	0.0096
## Detection Prevalence	0.0144	0.0148	0.0128	0.0204	0.0240
## Balanced Accuracy	0.7069	0.7169	0.5853	0.8569	0.7327
##	Class: 47	Class: 48	Class: 49	Class: 50	
## Sensitivity	0.4200	0.6800	0.5800	0.3400	
## Specificity	0.9939	0.9914	0.9804	0.9894	
## Pos Pred Value	0.5833	0.6182	0.3766	0.3953	
## Neg Pred Value	0.9882	0.9935	0.9913	0.9866	
## Prevalence	0.0200	0.0200	0.0200	0.0200	
## Detection Rate	0.0084	0.0136	0.0116	0.0068	
## Detection Prevalence	0.0144	0.0220	0.0308	0.0172	
## Balanced Accuracy	0.7069	0.8357	0.7802	0.6647	

This Random Forest model gives us an improved accuracy of 62.04% (compared to Naive Bayes model's 60.24%). Although it yielded a better performance, we might stick with Naive Bayes in practice since it's 1) more computationally efficient and 2) more interpretable and intuitive.

Practice with association rule mining

In this problem we used data on grocery purchases to find some interesting association rules for these shopping baskets. First we tried the Apriori algorithm with the following parameters: support=.01, confidence=.55, maxlen=4, and lift>=2. Below are the items from these baskets. The ubiquitous terms appear to be whole milk, vegetables, yogurt, and fruit.

```
## Apriori
##
## Parameter specification:
## confidence minval smax arem aval originalSupport support minlen maxlen
##      0.55    0.1    1 none FALSE          TRUE    0.01      1      4
## target  ext
## rules FALSE
##
## Algorithmic control:
## filter tree heap memopt load sort verbose
##      0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [7 rule(s)] done [0.00s].
## creating S4 object ... done [0.00s].

##      lhs                      rhs          support confidence    lift
## 1 {curd,
##   yogurt}                    => {whole milk}  0.01006609  0.5823529  2.279125
## 2 {butter,
##   other vegetables} => {whole milk}  0.01148958  0.5736041  2.244885
## 3 {domestic eggs,
##   other vegetables} => {whole milk}  0.01230300  0.5525114  2.162336
## 4 {citrus fruit,
##   root vegetables} => {other vegetables} 0.01037112  0.5862069  3.029608
## 5 {root vegetables,
##   tropical fruit}  => {other vegetables} 0.01230300  0.5845411  3.020999
## 6 {root vegetables,
##   tropical fruit}  => {whole milk}    0.01199797  0.5700483  2.230969
## 7 {root vegetables,
##   yogurt}          => {whole milk}    0.01453991  0.5629921  2.203354
```

To find more interesting rules, we ran the algorithm again with support=.001, confidence=.55, maxlen=4, and lift>=10. This gave a list of 7 rules. By lowering the support, we were able to find items that appeared less often, and by raising the lift, were able to find more significant associations. We left confidence the same; changing it higher would reduce the number of rules too much, and changing it lower resulted in more similar, less-varied rules, like {baking powder,flour} => {sugar} and {baking powder,margarine} => {sugar}.

```
## Apriori
##
```

```

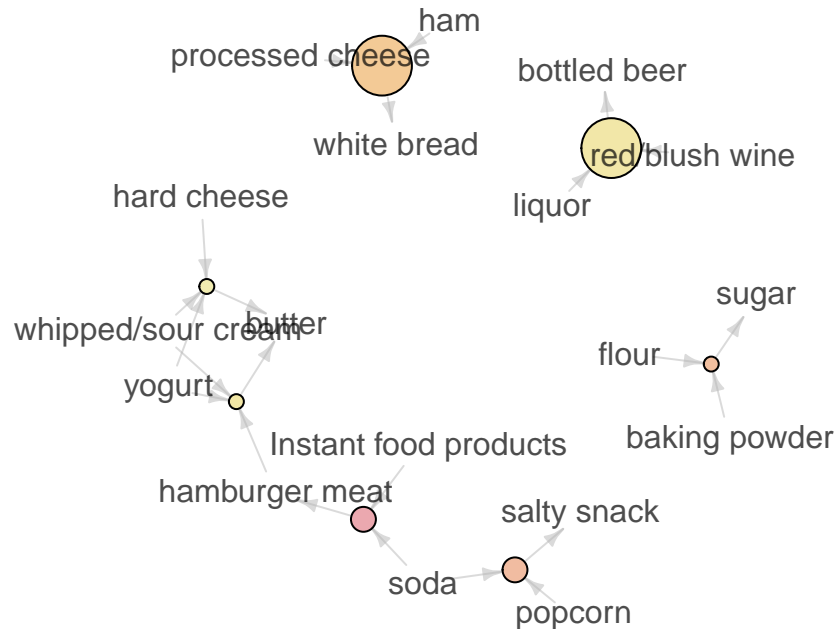
## Parameter specification:
## confidence minval smax arem aval originalSupport support minlen maxlen
##      0.55    0.1    1 none FALSE          TRUE    0.001      1      4
## target  ext
## rules FALSE
##
## Algorithmic control:
## filter tree heap memopt load sort verbose
##      0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.01s].
## writing ... [3314 rule(s)] done [0.00s].
## creating S4 object ... done [0.00s].

##      lhs                      rhs          support confidence    lift
## 1 {liquor,
##   red/blush wine}          => {bottled beer}    0.001931876  0.9047619 11.23527
## 2 {popcorn,
##   soda}                    => {salty snack}    0.001220132  0.6315789 16.69779
## 3 {Instant food products,
##   soda}                    => {hamburger meat} 0.001220132  0.6315789 18.99565
## 4 {ham,
##   processed cheese}        => {white bread}    0.001931876  0.6333333 15.04549
## 5 {baking powder,
##   flour}                   => {sugar}          0.001016777  0.5555556 16.40807
## 6 {hard cheese,
##   whipped/sour cream,
##   yogurt}                  => {butter}          0.001016777  0.5882353 10.61522
## 7 {hamburger meat,
##   whipped/sour cream,
##   yogurt}                  => {butter}          0.001016777  0.6250000 11.27867

```

Graph for 7 rules

size: support (0.001 – 0.002)
color: lift (10.615 – 18.996)



From the output, the item sets make a lot of sense. Some common sense relationships are purchasing liquor and wine leading to beer (90% of people who buy liquor and wine will also buy beer), purchasing popcorn and soda leading to salty snacks, and purchasing baking powder and flour leading to sugar (compared to a random person, people who buy baking powder and flour are 16 times as likely to buy sugar). The explanations are quite intuitive. Popcorn, soda, and salty snacks are complementary goods, while baking powder, flour, and sugar are highly related due to the organizational structure of supermarkets. There are also combo deals that might sell certain popular goods together, like ham and cheese for making sandwiches.