Exercises 2

Flights at ABIA

Here we will be looking at flights into and out of Austin in the year 2008. I will present some interesting findings about flight delays out of Austin.

First, read in the data and examine some of the features.

```
atxflights = read.csv('ABIA.csv')
head(atxflights)
```

```
##   Year Month DayofMonth DayOfWeek DepTime CRSDepTime ArrTime CRSArrTime
## 1 2008     1          1         2     120       1935     309       2130
## 2 2008     1          1         2     555        600     826        835
## 3 2008     1          1         2     600        600     728        729
## 4 2008     1          1         2     601        605     727        750
## 5 2008     1          1         2     601        600     654        700
## 6 2008     1          1         2     636        645     934        932
##   UniqueCarrier FlightNum TailNum ActualElapsedTime CRSElapsedTime AirTime
## 1            9E      5746  84129E               109            115      88
## 2            AA      1614  N438AA               151            155     133
## 3            YV      2883  N922FJ               148            149     125
## 4            9E      5743  89189E                86            105      70
## 5            AA      1157  N4XAAA                53             60      38
## 6            NW      1674   N967N               178            167     145
##   ArrDelay DepDelay Origin Dest Distance TaxiIn TaxiOut Cancelled
## 1      339      345    MEM  AUS      559      3      18         0
## 2       -9       -5    AUS  ORD      978      7      11         0
## 3       -1        0    AUS  PHX      872      7      16         0
## 4      -23       -4    AUS  MEM      559      4      12         0
## 5       -6        1    AUS  DFW      190      5      10         0
## 6        2       -9    AUS  MSP     1042     11      22         0
##   CancellationCode Diverted CarrierDelay WeatherDelay NASDelay
## 1                         0          339            0        0
## 2                         0           NA           NA       NA
## 3                         0           NA           NA       NA
## 4                         0           NA           NA       NA
## 5                         0           NA           NA       NA
## 6                         0           NA           NA       NA
##   SecurityDelay LateAircraftDelay
## 1             0                 0
## 2            NA                NA
## 3            NA                NA
## 4            NA                NA
## 5            NA                NA
## 6            NA                NA
```

```
summary(atxflights)
```

```
##       Year          Month         DayofMonth       DayOfWeek
##  Min.   :2008   Min.   : 1.00   Min.   : 1.00   Min.   :1.000
##  1st Qu.:2008   1st Qu.: 3.00   1st Qu.: 8.00   1st Qu.:2.000
```

```
##  Median :2008   Median : 6.00   Median :16.00   Median :4.000
##  Mean   :2008   Mean   : 6.29   Mean   :15.73   Mean   :3.902
##  3rd Qu.:2008   3rd Qu.: 9.00   3rd Qu.:23.00   3rd Qu.:6.000
##  Max.   :2008   Max.   :12.00   Max.   :31.00   Max.   :7.000
##
##     DepTime       CRSDepTime      ArrTime       CRSArrTime
##  Min.   :   1   Min.   :  55   Min.   :   1   Min.   :   5
##  1st Qu.: 917   1st Qu.: 915   1st Qu.:1107   1st Qu.:1115
##  Median :1329   Median :1320   Median :1531   Median :1535
##  Mean   :1329   Mean   :1320   Mean   :1487   Mean   :1505
##  3rd Qu.:1728   3rd Qu.:1720   3rd Qu.:1903   3rd Qu.:1902
##  Max.   :2400   Max.   :2346   Max.   :2400   Max.   :2400
##  NA's   :1413                  NA's   :1567
##  UniqueCarrier     FlightNum       TailNum      ActualElapsedTime
##  WN     :34876   Min.   :   1          : 1104   Min.   : 22.0
##  AA     :19995   1st Qu.: 640   N678CA :  195   1st Qu.: 57.0
##  CO     : 9230   Median :1465   N511SW :  180   Median :125.0
##  YV     : 4994   Mean   :1917   N526SW :  176   Mean   :120.2
##  B6     : 4798   3rd Qu.:2653   N528SW :  172   3rd Qu.:164.0
##  XE     : 4618   Max.   :9741   N520SW :  168   Max.   :506.0
##  (Other):20749                  (Other):97265   NA's   :1601
##  CRSElapsedTime     AirTime         ArrDelay          DepDelay
##  Min.   : 17.0   Min.   :  3.00   Min.   :-129.000   Min.   :-42.000
##  1st Qu.: 58.0   1st Qu.: 38.00   1st Qu.:  -9.000   1st Qu.: -4.000
##  Median :130.0   Median :105.00   Median :  -2.000   Median :  0.000
##  Mean   :122.1   Mean   : 99.81   Mean   :   7.065   Mean   :  9.171
##  3rd Qu.:165.0   3rd Qu.:142.00   3rd Qu.:  10.000   3rd Qu.:  8.000
##  Max.   :320.0   Max.   :402.00   Max.   : 948.000   Max.   :875.000
##  NA's   :11      NA's   :1601     NA's   :1601       NA's   :1413
##      Origin          Dest          Distance        TaxiIn
##  AUS    :49623   AUS    :49637   Min.   :  66   Min.   :  0.000
##  DAL    : 5583   DAL    : 5573   1st Qu.: 190   1st Qu.:  4.000
##  DFW    : 5508   DFW    : 5506   Median : 775   Median :  5.000
##  IAH    : 3704   IAH    : 3691   Mean   : 705   Mean   :  6.413
##  PHX    : 2786   PHX    : 2783   3rd Qu.:1085   3rd Qu.:  7.000
##  DEN    : 2719   DEN    : 2673   Max.   :1770   Max.   :143.000
##  (Other):29337   (Other):29397                  NA's   :1567
##     TaxiOut         Cancelled       CancellationCode    Diverted
##  Min.   :  1.00   Min.   :0.00000    :97840          Min.   :0.000000
##  1st Qu.:  9.00   1st Qu.:0.00000   A:  719          1st Qu.:0.000000
##  Median : 12.00   Median :0.00000   B:  605          Median :0.000000
##  Mean   : 13.96   Mean   :0.01431   C:   96          Mean   :0.001824
##  3rd Qu.: 16.00   3rd Qu.:0.00000                    3rd Qu.:0.000000
##  Max.   :305.00   Max.   :1.00000                    Max.   :1.000000
##  NA's   :1419
##   CarrierDelay     WeatherDelay       NASDelay      SecurityDelay
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :  0.00   Median :  0.00   Median :  2.00   Median :  0.00
##  Mean   : 15.39   Mean   :  2.24   Mean   : 12.47   Mean   :  0.07
##  3rd Qu.: 16.00   3rd Qu.:  0.00   3rd Qu.: 16.00   3rd Qu.:  0.00
##  Max.   :875.00   Max.   :412.00   Max.   :367.00   Max.   :199.00
##  NA's   :79513    NA's   :79513    NA's   :79513    NA's   :79513
##  LateAircraftDelay
```

```
##   Min.   :  0.00
##   1st Qu.:  0.00
##   Median :  6.00
##   Mean   : 22.97
##   3rd Qu.: 30.00
##   Max.   :458.00
##   NA's   :79513
```

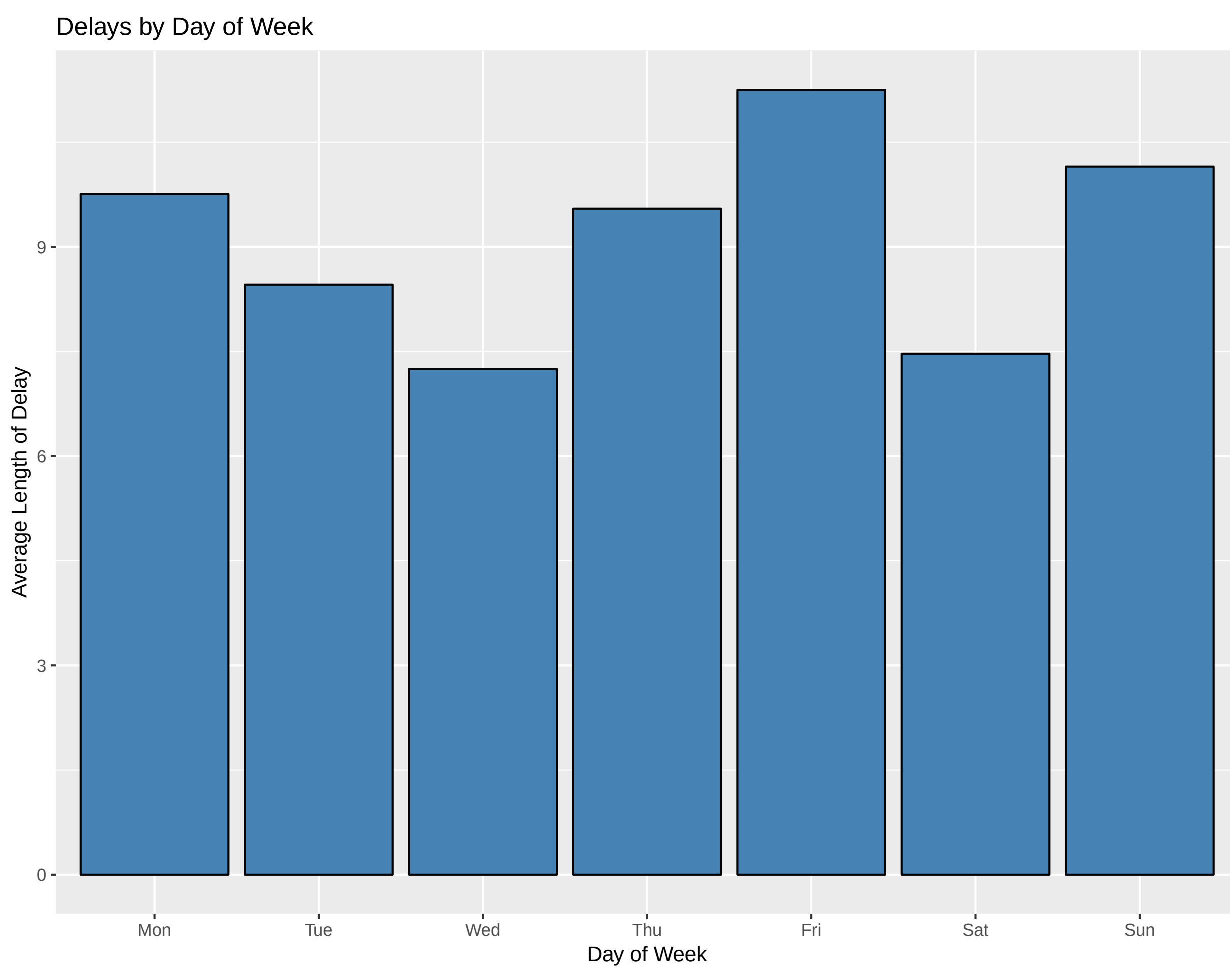
I want to take a look at flight delays by day of the week, and by month. However these variables are stored as numeric so we'll convert them to factors to make them better to deal with, as well as give them some labels.

```
# convert a couple variables to factors
atxflights$DayOfWeek = factor(atxflights$DayOfWeek, levels = c(1,2,3,4,5,6,7),
                              labels = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"))
atxflights$Month = factor(atxflights$Month, levels = c(1,2,3,4,5,6,7,8,9,10,11,12),
                          labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oc
```

I want to find the average flight delay length over each day of the week, so I'll store those in a new data frame.

```
# get the average length of delays by day of week
avgdelay = aggregate(atxflights$DepDelay, by = list(atxflights$DayOfWeek), FUN = mean, na.rm = TRUE)
avgdelay = as.data.frame(avgdelay)
```

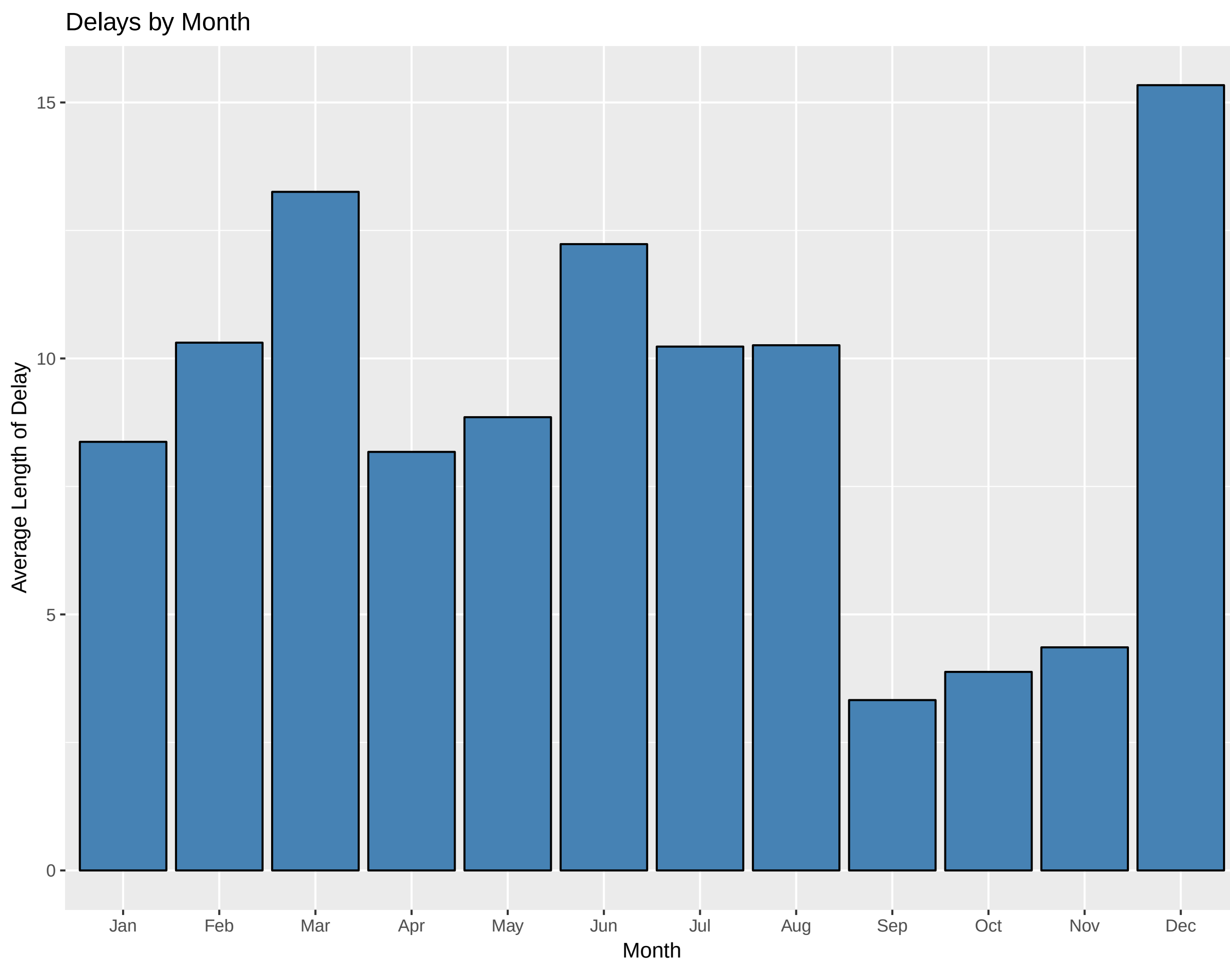
Now we plot flight delays by day of the week.



The worst day of the week for delays was Friday - this could be problematic as it is the end of the work week and many people would probably be traveling either back home or leaving on a trip. Wednesday and Saturday are pretty good days if you want to cut down on flight delay time.

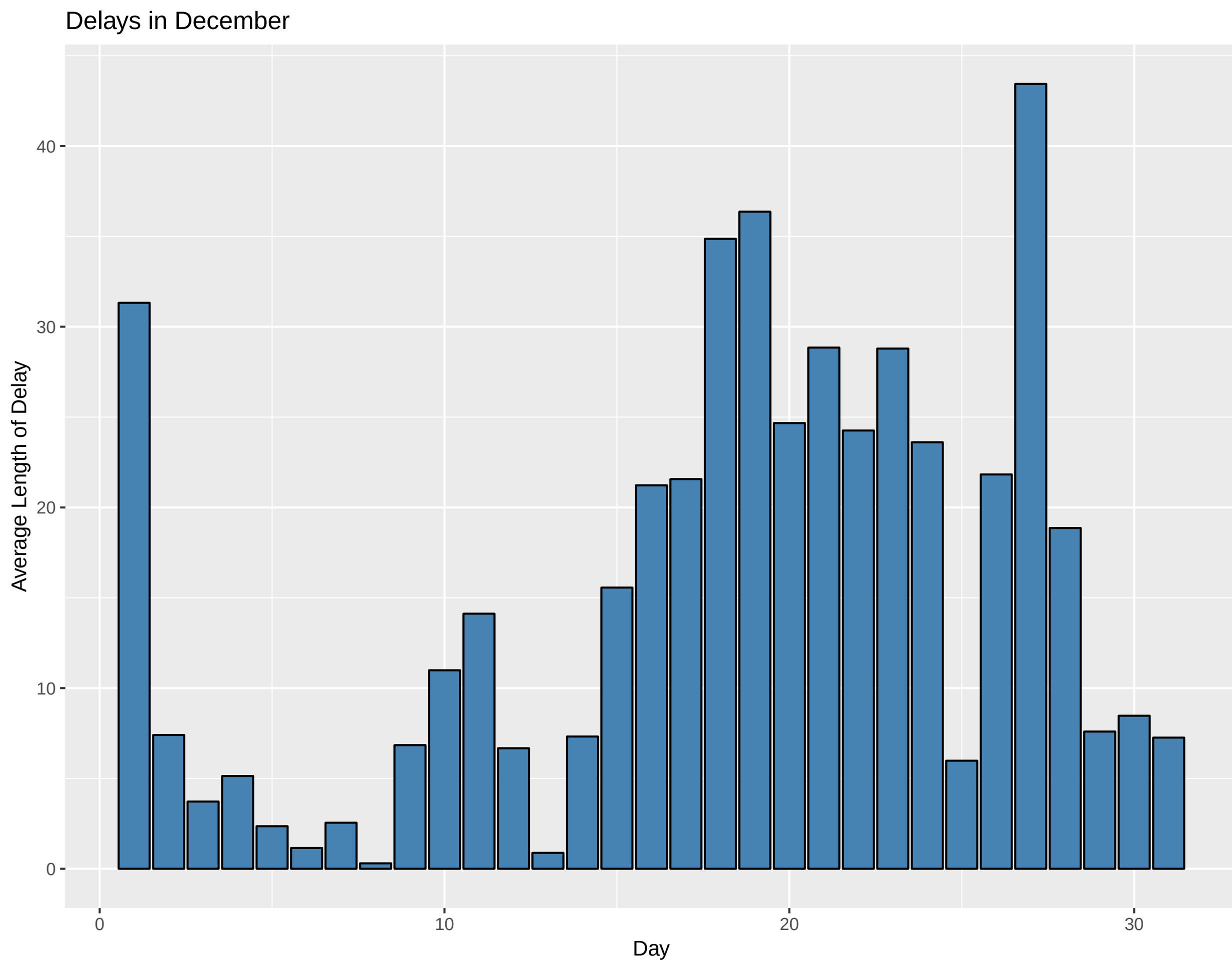
Next I want to take a look at the worst delay times by month. Again we'll compute the averages and plot it.

```
# get average delay by month
avgdelaymonth = aggregate(atxflights$DepDelay, by = list(atxflights$Month), FUN = mean, na.rm = TRUE)
avgdelaymonth = as.data.frame(avgdelaymonth)
```



December is the worst month, followed by March - not surprisingly, as we have the holiday season in December as well as SXSW in Austin during the month of March - expect longer delays at those times.

Let's take a look at the month of December - when do you experience the worst delays?



December 27th had the worst delay times - just a couple days after Christmas. Looks like flying on Christmas Day is the way to go! Or do your traveling around December 8th and you'll experience almost no delays. Unfortunately school is still in session!

Author Attribution

Now we'll look at the C50 corpus and build a couple models to try to predict the author, and see how we do.

The first step was pre-processing which involved reading in the documents, cleaning up file names, getting author names from the files, building the corpus, and removing unnecessary things like numbers, punctuation, stop words, etc.

Then we build the Document Term Matrix and check the summary.

```
## <<DocumentTermMatrix (documents: 5000, terms: 45522)>>
## Non-/sparse entries: 954272/226655728
## Sparsity           : 100%
## Maximal term length: 45
## Weighting          : term frequency (tf)
```

There are 45000+ terms here - we need to remove sparce terms to cut that number down. We'll remove terms that didn't come up in 97.5% of the documents.

```
## <<DocumentTermMatrix (documents: 5000, terms: 1407)>>
```

```
## Non-/sparse entries: 581774/6453226
## Sparsity           : 92%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

Much better - about 1400 terms now.

Next we set up the training and test sets from the Document Term Matrix, and we will simply ignore words in the test set that didn't appear in the training set in order to avoid getting zero probabilities.

We will try a random forest using PCA to get the first 100 principal components to put in the random forest model.

The accuracy of the random forest turns out to be:

```
accuracy_rf
```

```
## [1] 0.51
```

Not so great as we get an accuracy of 51%. Let's try another model.

Next we'll try a Naive Bayes model. First we use the training set and apply a smoothing function, then get the predictions and check them against the test set.

Then we get the accuracy for Naive Bayes:

```
accuracy_nb
```

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

It turns out Naive Bayes has an accuracy of about 60%, better than that of random forest. The table below shows the authors with how well the model did at predicting each author.

```
##               author nbcorrect accuracy
## 1      AaronPressman        42     0.84
## 2         AlanCrosby        25     0.50
## 3     AlexanderSmith        21     0.42
## 4    BenjaminKangLim        11     0.22
## 5      BernardHickey        27     0.54
## 6         BradDorfman        44     0.88
## 7   DarrenSchuettler        14     0.28
## 8         DavidLawder         7     0.14
## 9      EdnaFernandes        20     0.40
## 10       EricAuchard        25     0.50
## 11    FumikoFujisaki        49     0.98
## 12    GrahamEarnshaw        41     0.82
## 13  HeatherScoffield        17     0.34
## 14     JaneMacartney        23     0.46
## 15        JanLopatka        18     0.36
## 16      JimGilchrist        50     1.00
## 17          JoeOrtiz        34     0.68
## 18      JohnMastrini        40     0.80
## 19      JonathanBirt        32     0.64
## 20    JoWinterbottom        36     0.72
## 21       KarlPenhaul        47     0.94
## 22         KeithWeir        37     0.74
## 23    KevinDrawbaugh        27     0.54
## 24     KevinMorrison        27     0.54
## 25     KirstinRidley        34     0.68
## 26 KouroshKarimkhany        32     0.64
```

```
## 27         LydiaZajc        31      0.62
## 28    LynneO'Donnell        39      0.78
## 29   LynnleyBrowning        49      0.98
## 30   MarcelMichelson        30      0.60
## 31      MarkBendeich        20      0.40
## 32        MartinWolk        25      0.50
## 33      MatthewBunce        42      0.84
## 34     MichaelConnor        40      0.80
## 35        MureDickie        14      0.28
## 36         NickLouth        40      0.80
## 37   PatriciaCommins        31      0.62
## 38     PeterHumphrey        33      0.66
## 39        PierreTran        34      0.68
## 40        RobinSidel        40      0.80
## 41      RogerFillion        38      0.76
## 42       SamuelPerry        32      0.64
## 43      SarahDavison        27      0.54
## 44       ScottHillis        13      0.26
## 45       SimonCowell        27      0.54
## 46          TanEeLyn        21      0.42
## 47    TheresePoletti        25      0.50
## 48        TimFarrand        39      0.78
## 49        ToddNissen        19      0.38
## 50      WilliamKazer        17      0.34
```

The model did well at predicting Jim Gilchrist - 100% accuracy. However, it did very poorly at predicting David Lawder. Let's take a look.

```
##                author vector
## 1      AaronPressman      0
## 2         AlanCrosby      0
## 3     AlexanderSmith      0
## 4    BenjaminKangLim      0
## 5      BernardHickey      0
## 6        BradDorfman      8
## 7   DarrenSchuettler      0
## 8        DavidLawder      7
## 9      EdnaFernandes      0
## 10       EricAuchard      0
## 11    FumikoFujisaki      0
## 12    GrahamEarnshaw      0
## 13  HeatherScoffield      0
## 14     JaneMacartney      0
## 15        JanLopatka      1
## 16      JimGilchrist      0
## 17          JoeOrtiz      0
## 18      JohnMastrini      1
## 19      JonathanBirt      0
## 20    JoWinterbottom      0
## 21       KarlPenhaul      1
## 22         KeithWeir      0
## 23    KevinDrawbaugh      3
## 24     KevinMorrison      0
## 25     KirstinRidley      0
## 26 KouroshKarimkhany      0
```

```
## 27          LydiaZajc      0
## 28     LynneO'Donnell      0
## 29   LynnleyBrowning      0
## 30   MarcelMichelson      0
## 31       MarkBendeich      0
## 32         MartinWolk      2
## 33       MatthewBunce      0
## 34      MichaelConnor      0
## 35         MureDickie      0
## 36          NickLouth      0
## 37   PatriciaCommins      0
## 38      PeterHumphrey      0
## 39         PierreTran      0
## 40         RobinSidel      4
## 41       RogerFillion      0
## 42        SamuelPerry      0
## 43       SarahDavison      0
## 44        ScottHillis      0
## 45        SimonCowell      0
## 46           TanEeLyn      0
## 47    TheresePoletti      0
## 48         TimFarrand      0
## 49         ToddNissen     23
## 50       WilliamKazer      0
```

The model attributed many of David Lawder's documents to the author Todd Nissen, so perhaps these two authors are difficult to distinguish.

Overall, the Naive Bayes model performed the best, and even though it assumes independent features and we know that many words are correlated with each other, the Naive Bayes model had much better accuracy than random forest so we will choose Naive Bayes.

Association Rules Mining

Here we will examine some interesting association rules among shopping baskets from the data on grocery purchases. We will set a support threshold of 0.01 to get rules that occurred in 1% of the data, and confidence of 0.5 to get rules that were correct at least half the time. Setting the maximum size to 4 made no difference so we set it to 3. This gave 15 rules.

```r
inspect(groceryrules)
```

```
##      lhs                        rhs                   support confidence     lift count
## [1]  {curd,                                                                            
##       yogurt}               => {whole milk}       0.01006609  0.5823529 2.279125    99
## [2]  {butter,                                                                          
##       other vegetables}     => {whole milk}       0.01148958  0.5736041 2.244885   113
## [3]  {domestic eggs,                                                                   
##       other vegetables}     => {whole milk}       0.01230300  0.5525114 2.162336   121
## [4]  {whipped/sour cream,                                                              
##       yogurt}               => {whole milk}       0.01087951  0.5245098 2.052747   107
## [5]  {other vegetables,                                                                
##       whipped/sour cream} => {whole milk}       0.01464159  0.5070423 1.984385   144
## [6]  {other vegetables,                                                                
##       pip fruit}            => {whole milk}       0.01352313  0.5175097 2.025351   133
```

```
## [7]  {citrus fruit,
##       root vegetables}    => {other vegetables} 0.01037112  0.5862069 3.029608   102
## [8]  {root vegetables,
##       tropical fruit}     => {other vegetables} 0.01230300  0.5845411 3.020999   121
## [9]  {root vegetables,
##       tropical fruit}     => {whole milk}       0.01199797  0.5700483 2.230969   118
## [10] {tropical fruit,
##       yogurt}             => {whole milk}       0.01514997  0.5173611 2.024770   149
## [11] {root vegetables,
##       yogurt}             => {other vegetables} 0.01291307  0.5000000 2.584078   127
## [12] {root vegetables,
##       yogurt}             => {whole milk}       0.01453991  0.5629921 2.203354   143
## [13] {rolls/buns,
##       root vegetables}    => {other vegetables} 0.01220132  0.5020921 2.594890   120
## [14] {rolls/buns,
##       root vegetables}    => {whole milk}       0.01270971  0.5230126 2.046888   125
## [15] {other vegetables,
##       yogurt}             => {whole milk}       0.02226741  0.5128806 2.007235   219
```

Most of these rules predict whole milk, with a few predicting other vegetables. The rules predicting whole milk generally have other dairy items like yogurt, butter, or eggs, indicating people tend to buy milk when they buy other dairy items. Other vegetables were commonly associated with people buying root vegetables and some kind of fruit - these shoppers tend to buy fruits and vegetables together. Overall this gave a pretty small sample of rules, so we will change a couple parameters to try to look at other common rules.

We'll set the support to 0.001 to try and include more rules, and to compensate this we'll set confidence to 0.8 and only look at the rules with lift > 5. There was not much change increasing the maximum length above 5 so we set it to 5.

```
inspect(subset(groceryrules2, subset=lift > 5))
```

```
##      lhs                           rhs                    support confidence      lift count
## [1]  {liquor,
##       red/blush wine}           => {bottled beer}     0.001931876  0.9047619 11.235269    19
## [2]  {citrus fruit,
##       root vegetables,
##       soft cheese}              => {other vegetables} 0.001016777  1.0000000  5.168156    10
## [3]  {citrus fruit,
##       fruit/vegetable juice,
##       grapes}                   => {tropical fruit}   0.001118454  0.8461538  8.063879    11
## [4]  {butter milk,
##       other vegetables,
##       pastry}                   => {yogurt}           0.001220132  0.8000000  5.734694    12
## [5]  {pip fruit,
##       sausage,
##       sliced cheese}            => {yogurt}           0.001220132  0.8571429  6.144315    12
## [6]  {cream cheese,
##       margarine,
##       whipped/sour cream}       => {yogurt}           0.001016777  0.8333333  5.973639    10
## [7]  {butter,
##       cream cheese,
##       root vegetables}          => {yogurt}           0.001016777  0.9090909  6.516698    10
## [8]  {butter,
##       tropical fruit,
##       white bread}              => {yogurt}           0.001118454  0.8461538  6.065542    11
## [9]  {beef,
```

```
##       butter,
##       tropical fruit}        => {yogurt}           0.001016777  0.8333333  5.973639    10
## [10] {fruit/vegetable juice,
##       pork,
##       tropical fruit}        => {yogurt}           0.001016777  0.8333333  5.973639    10
## [11] {brown bread,
##       pip fruit,
##       whipped/sour cream}    => {other vegetables} 0.001118454  1.0000000  5.168156    11
## [12] {butter,
##       margarine,
##       tropical fruit}        => {yogurt}           0.001118454  0.8461538  6.065542    11
## [13] {fruit/vegetable juice,
##       pastry,
##       whipped/sour cream}    => {yogurt}           0.001220132  0.8000000  5.734694    12
## [14] {other vegetables,
##       rice,
##       whole milk,
##       yogurt}                => {root vegetables}  0.001321810  0.8666667  7.951182    13
## [15] {grapes,
##       tropical fruit,
##       whole milk,
##       yogurt}                => {other vegetables} 0.001016777  1.0000000  5.168156    10
## [16] {ham,
##       pip fruit,
##       tropical fruit,
##       yogurt}                => {other vegetables} 0.001016777  1.0000000  5.168156    10
## [17] {ham,
##       other vegetables,
##       pip fruit,
##       yogurt}                => {tropical fruit}   0.001016777  0.8333333  7.941699    10
## [18] {ham,
##       pip fruit,
##       tropical fruit,
##       whole milk}            => {other vegetables} 0.001118454  1.0000000  5.168156    11
## [19] {butter,
##       sliced cheese,
##       tropical fruit,
##       whole milk}            => {yogurt}           0.001016777  0.9090909  6.516698    10
## [20] {oil,
##       other vegetables,
##       tropical fruit,
##       whole milk}            => {root vegetables}  0.001321810  0.8666667  7.951182    13
## [21] {cream cheese,
##       curd,
##       other vegetables,
##       whipped/sour cream}    => {yogurt}           0.001016777  0.9090909  6.516698    10
## [22] {cream cheese,
##       curd,
##       whipped/sour cream,
##       whole milk}            => {yogurt}           0.001118454  0.8461538  6.065542    11
## [23] {butter,
##       other vegetables,
##       tropical fruit,
##       white bread}           => {yogurt}           0.001016777  0.9090909  6.516698    10
```

```
## [24] {beef,
##       citrus fruit,
##       other vegetables,
##       tropical fruit}       => {root vegetables}  0.001016777  0.8333333  7.645367    10
## [25] {butter,
##       curd,
##       other vegetables,
##       tropical fruit}       => {yogurt}           0.001016777  0.8333333  5.973639    10
## [26] {butter,
##       curd,
##       tropical fruit,
##       whole milk}           => {yogurt}           0.001220132  0.8571429  6.144315    12
## [27] {margarine,
##       root vegetables,
##       tropical fruit,
##       whole milk}           => {yogurt}           0.001016777  0.8333333  5.973639    10
## [28] {butter,
##       fruit/vegetable juice,
##       tropical fruit,
##       whipped/sour cream}   => {other vegetables} 0.001016777  1.0000000  5.168156    10
## [29] {newspapers,
##       rolls/buns,
##       soda,
##       whole milk}           => {other vegetables} 0.001016777  1.0000000  5.168156    10
## [30] {citrus fruit,
##       fruit/vegetable juice,
##       other vegetables,
##       soda}                 => {root vegetables}  0.001016777  0.9090909  8.340400    10
## [31] {citrus fruit,
##       root vegetables,
##       tropical fruit,
##       whipped/sour cream}   => {other vegetables} 0.001220132  1.0000000  5.168156    12
```

This gives us some more interesting rules to work with. There are quite a few associations of yogurt here with other dairy items so that reaffirms the previous finding about dairy shoppers, as well as different fruits showing up in many of the rules that predict yogurt - makes sense as people like to mix fruit and yogurt. One very interesting rule here is liquor and red wine predicting bottled beer, as it has the highest lift and a very high confidence of over 90%. Basically it's very significant and people will likely buy bottled beer when they have bought liquor and wine as well, so a grocery store should be sure to put beer close by the liquor and wine, and maybe include some promotional marketing or coupons to maximize profits.