Homework2- STA380: Agarwal, Sneha

Question 1

Loading the libraries and reading file ABIA.csv

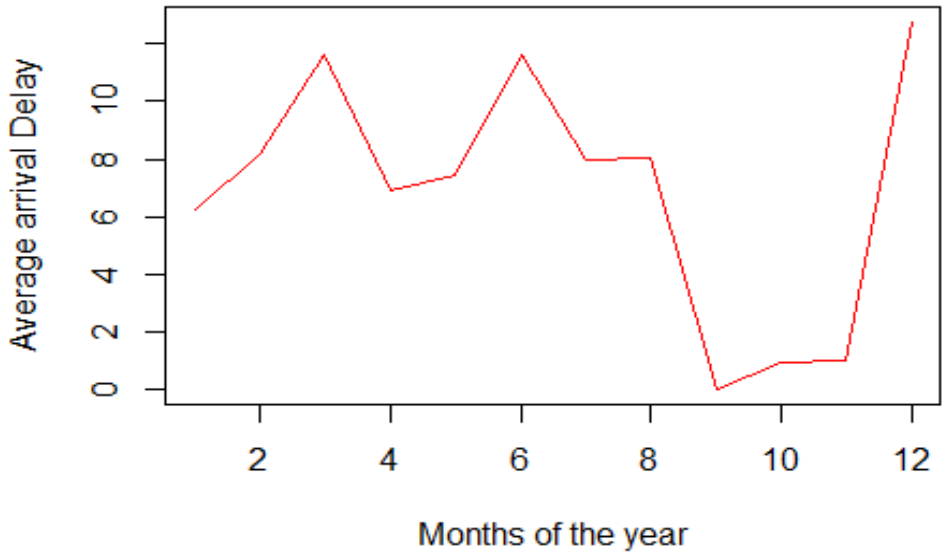
```
library(ggplot2)
library(XML)
airport = read.csv("https://raw.githubusercontent.com/jgscott/STA380/master/data/ABIA.csv")
attach(airport)
```

There could be a number of exploratory analysis on this question. But, I choose to look at the delays and find its relationship with the Month and DayofWeek. What are the times, passengers should avoid booking tickets as there might be delay in flight schedules?

Findings:

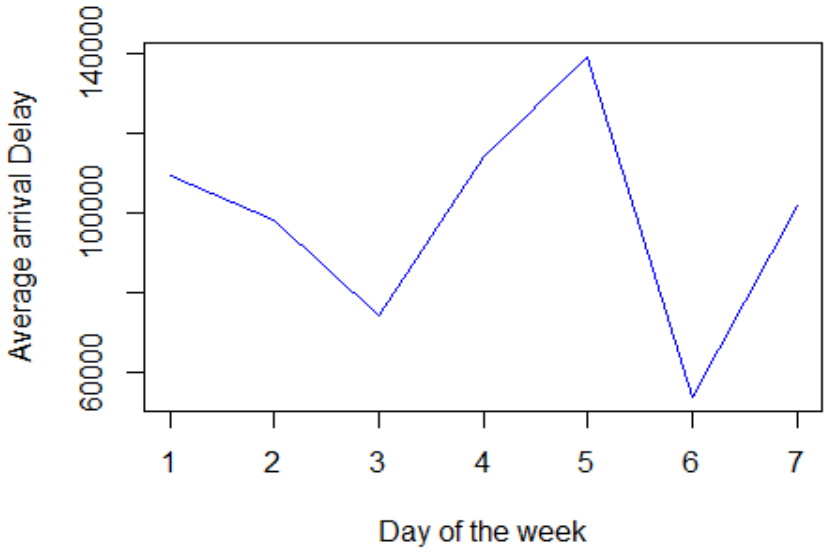
- For the months of September, October, November delay is least

```
delay =aggregate(airport$ArrDelay ,by = list(Month), FUN = mean, na.rm = TRUE)
plot(delay$Group.1, delay$x, type = "l", col = 'red', xlab = 'Months of the year', ylab = 'Average arrival Delay')
```



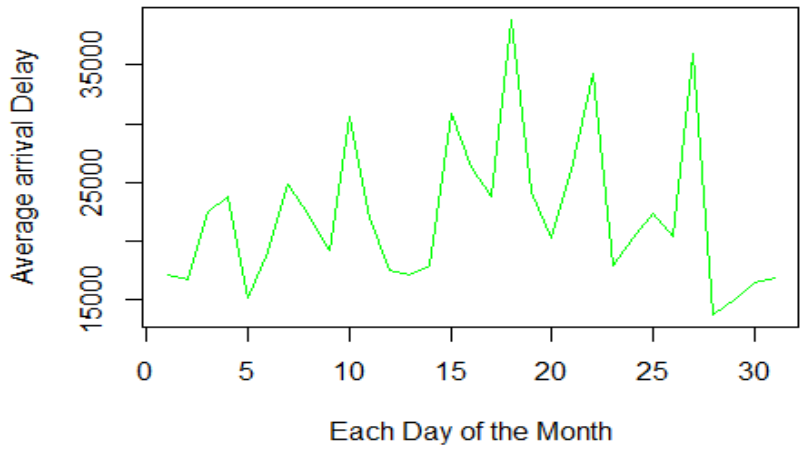
- For the Fridays we have maximum delay and least for Saturdays.

```
delay1 =aggregate(airport$ArrDelay ,by = list(DayOfWeek), FUN = sum, na.rm =
TRUE)
plot(delay1$Group.1, delay1$x, type = "l", col = 'blue', xlab = 'Day of the w
eek', ylab = 'Average arrival Delay')
```



- Towards the beginning of the month the delays are less and increases towards the second half.

```
delay2 =aggregate(airport$ArrDelay ,by = list(DayofMonth), FUN = sum, na.rm = TRUE)
plot(delay2$Group.1, delay2$x, type = "l", col = 'green', xlab = 'Each Day of the Month', ylab = 'Average arrival Delay')
```

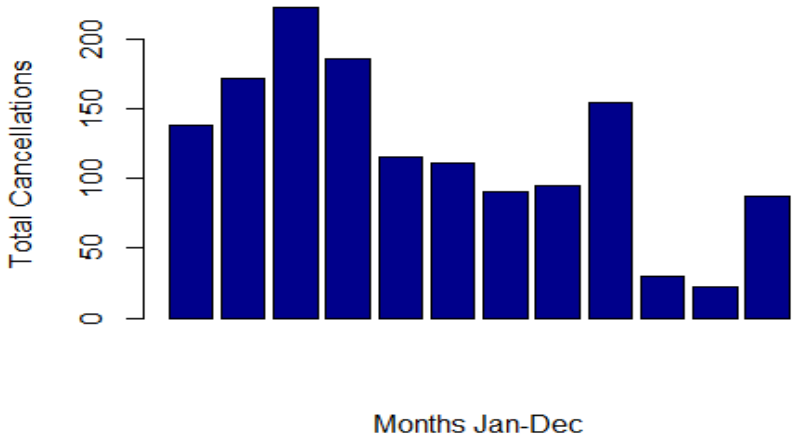


Another thing I checked was the number of cancellations over the year and week. So basically, what are the times when there is more possibility of flights getting cancelled?

Findings

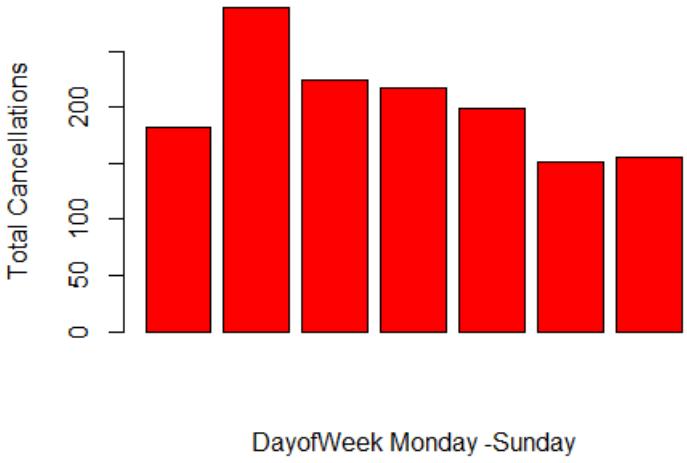
- In the month of March, we have the highest number of cancellations.

```
#Finding the relationship between the number of cancellations for each month
sum(airport$Cancelled)

## [1] 1420

monthbycancellation=aggregate(Cancelled,by = list(Month), sum)
Cancellations <- as.matrix(monthbycancellation[2])
barplot(t(Cancellations),xlab="Months Jan-Dec",ylab="Total Cancellations", col = 'darkblue')
```



- On Tuesdays we have the highest number of cancellations.

```
#Finding the relationship between the number of cancellations for each day
weekdaybycancellation=aggregate(Cancelled,by = list(DayOfWeek), sum)
Cancellations <- as.matrix(weekdaybycancellation[2])
barplot(t(Cancellations),xlab="DayofWeek Monday -Sunday",ylab="Total Cancella
tions", col = 'red')
```



Question2

Loading the libraries required.

```
library(tm)
library(randomForest)
library(e1071)
library(rpart)
library(ggplot2)
library(caret)
```

Looking at the CORPUS data and training the data.

```
#initialising reader function
readerPlain = function(fname){readPlain(elem=list(content=readLines(fname)),
id=fname, language='en') }

#Training data set -CORPUS

author_dirs = Sys.glob('C:/Users/Sneha/Documents/Education/STA380/data/Reuter
sC50/C50train/*')
file_list = NULL
train_labels = NULL
for(author in author_dirs) {
  author_name = substring(author, first=68)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  train_labels = append(train_labels, rep(author_name, length(files_to_add)))
}

# Named conversion & cleanup
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

#Initialize Training Corpus
train_corpus = Corpus(VectorSource(all_docs))
names(train_corpus) = file_list

#Pre-Processing
train_corpus = tm_map(train_corpus, content_transformer(tolower))
train_corpus = tm_map(train_corpus, content_transformer(removeNumbers))
train_corpus = tm_map(train_corpus, content_transformer(removePunctuation))
train_corpus = tm_map(train_corpus, content_transformer(stripWhitespace))
train_corpus = tm_map(train_corpus, content_transformer(removeWords), stopwor
ds("SMART"))
```

Create training DTM & dense matrix

```
DTM_train = DocumentTermMatrix(train_corpus)
DTM_train = removeSparseTerms(DTM_train, 0.975)
#DTM_train = as.matrix(DTM_train)
```

Now let us test the CORPUS data on training dataset

```
#Looking at the c50test folder

author_dirs = Sys.glob('C:/Users/Sneha/Documents/Education/STA380/data/ReutersC50/C50test/*')
file_list = NULL
test_labels = NULL
for(author in author_dirs) {
  author_name = substring(author, first=67)
  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)))
}

# Named conversion & cleanup
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

#Initialize Testing Corpus
test_corpus = Corpus(VectorSource(all_docs))
names(test_corpus) = file_list

#pre-processing

test_corpus = tm_map(test_corpus, content_transformer(tolower))
test_corpus = tm_map(test_corpus, content_transformer(removeNumbers))
test_corpus = tm_map(test_corpus, content_transformer(removePunctuation))
test_corpus = tm_map(test_corpus, content_transformer(stripWhitespace))
test_corpus = tm_map(test_corpus, content_transformer(removeWords), stopwords("SMART"))

#creating dictionary
reuters_dict = NULL
reuters_dict = dimnames(DTM_train)[[2]]

#Create testing DTM & matrix using dictionary words only
DTM_test = DocumentTermMatrix(test_corpus, list(dictionary=reuters_dict))
```

```
DTM_test = removeSparseTerms(DTM_test, 0.93)
#DTM_test = as.matrix(DTM_test)
```

Converting DTMs into the data frames

```
DTM_train_df = as.data.frame(inspect(DTM_train))
DTM_test_df = as.data.frame(inspect(DTM_test))
```

Creating a Naive Bayes Model - 1st model

The features are the class of each author.

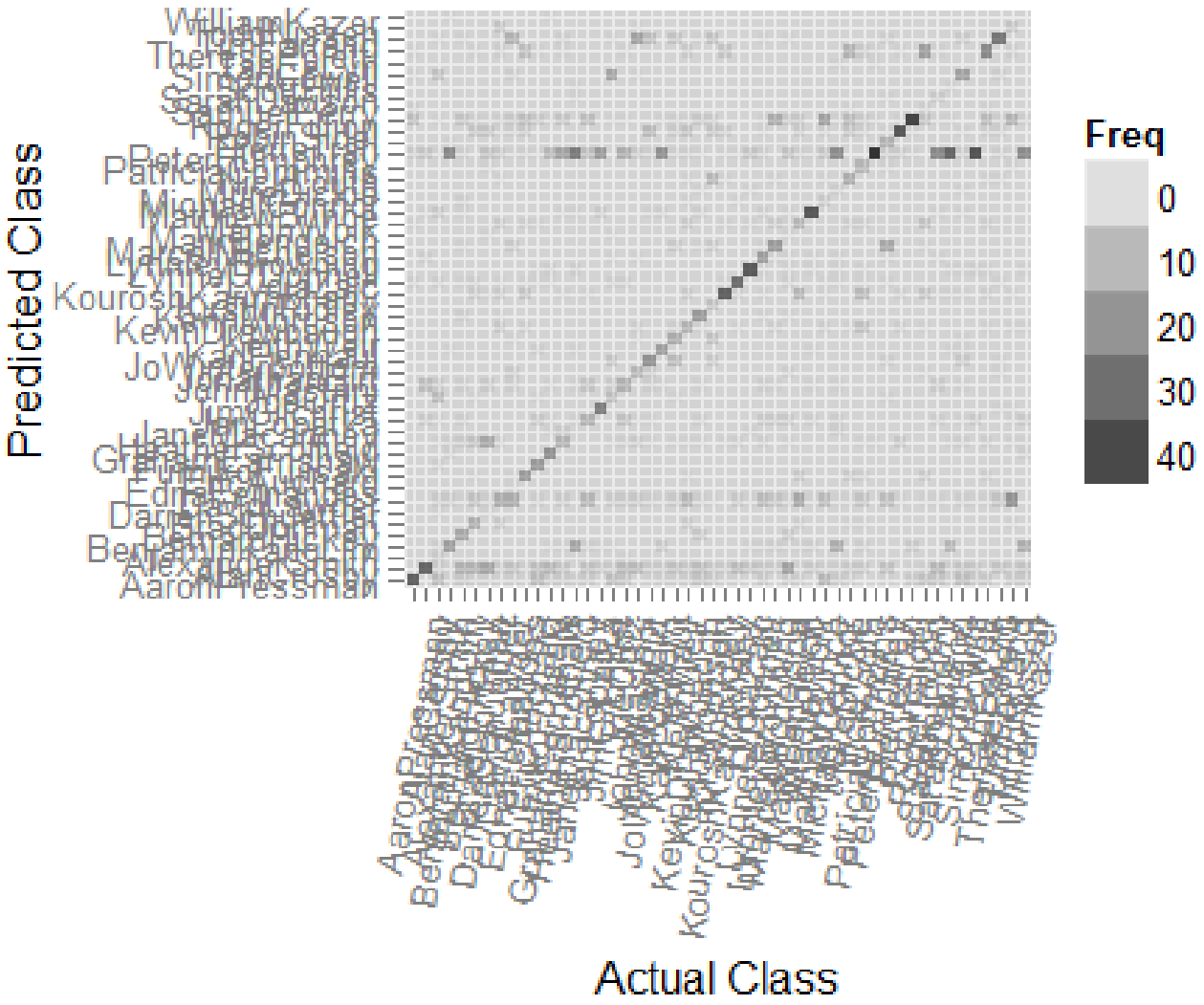
```
######### Naive Bayes Model #########

#Creating the model with the training dataframe and labels are the authors.
NBModel = naiveBayes(x = DTM_train_df, y = as.factor(train_labels), laplace=1
)

#Predicting the model on the test dataframe
NBPredict = predict(NBModel, DTM_test_df)
```

Let us now look into the details of this model. The table NBtable has the predicted values, and the label for each value. We get the frequency of each matched and unmatched pair, for the correctly predicted values

```
NBtable = as.data.frame(table(NBPredict,test_labels))
#head(NBtable)
```

Let's now plot the above table and see that how does it look visually. We are plotting the predicted value and the actual value and the frequency.

```
plot = ggplot(NBtable)
plot + geom_tile(aes(x=test_labels, y=NBPredict, alpha=Freq)) +
    scale_x_discrete(name="Actual Class") +
    scale_y_discrete(name="Predicted Class") +
    theme(axis.text.x = element_text(angle = 80, hjust = 1))
```



Let us construct a confusion matrix. And check how well the model could perform. In the Test data set, when we removed the sparse terms, I took the percentage as 93% to get a better prediction.

```
NBconfusion = confusionMatrix(table(NBPredict,test_labels))
NBconfusion$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.2956000      0.2812245      0.2777608      0.3139191      0.0200000
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

The accuracy of this model is 29.56%. For this bag of words, the prediction is pretty good. As, there the dataset is huge. We have 50 author data and even more number of .txt files.

But, let us try another model.

Model 2 - Random Forest Model

To execute this model, we need to add another column in the training and test dataset. We first convert the dataset into matrix and then add the column.

```
######### Random Forest Model #########
library(plyr)

DTM_test = as.matrix(DTM_test)
DTM_train = as.matrix(DTM_train)

words <- data.frame(DTM_test[,intersect(colnames(DTM_test), colnames(DTM_trai
n))])
readcol <- read.table(textConnection(""), col.names = colnames(DTM_train), co
lClasses = "integer")

DTM_test_final = rbind.fill(words, readcol)
DTM_test_df = as.data.frame(DTM_test_final)
```

Now let us do the modelling part.

```
modelRF = randomForest(x=DTM_train_df, y=as.factor(train_labels), mtry= 3, nt
ree=200)
```

Predicting the test data

```
predictRF = predict(modelRF, data=DTM_test_final)
```

Looking at the table having the predicted values along with different class of authors.

```
tableRF = as.data.frame(table(predictRF,test_labels))
```

Let's construct a confusion matrix and check the accuracy.

```
confusionRF = confusionMatrix(table(predictRF,test_labels))
confusionRF$overall

##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.6944000      0.6881633      0.6759223      0.7124212      0.0200000
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

The accuracy with mtry = 3, is 69.84%. I also inspected the accuracy by increasing th number of mtry and noticed that the accuracy increases with mtry of 4 and 5(To around 74%) but, I did not choose that as that might be overfitting.

Conclusion

The random forest model works better than the Naïve Bayes model.

Question3

Loading the required library.

```
library(arules)
```

```
## Warning: package 'arules' was built under R version 3.2.2
```

```
## Loading required package: Matrix
##
## Attaching package: 'arules'
##
## The following objects are masked from 'package:base':
##
##     %in%, write
```

```
#detach(package:tm, unload=TRUE)
```

Loading the groceries data, and as mentioned in the question took separator as comma and format as basket. And trying to look at the different association rules.

Now we create some association rules, by randomly selecting the support value, confidence and the maxlen. I wanted to look at a lot of association rules in the first place, so that I can find smaller subsets and try to see some interesting associations. I could have chosen values of support and confidence to be higher and get lesser rules, but I wanted to look at the lot of associations to find good associations in the subset.

```
groceriesrule <- apriori(groceries,parameter=list(support=.001, confidence=.1
5, maxlen=4))
```

```
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##        0.15    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
##
## apriori - find association rules with the apriori algorithm
## version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
## 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 [0.02s].
## writing ... [24907 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

groceriesrule has a set of 24907 associations. Now, I would like to see some patterns in the subsets.

```
#inspect
# Look at the output
inspect(groceriesrule)
```

First subset: I wanted to see the associations when lift is high, that is the dependent association by X and Y to be higher.

```
## Choose a subset
inspect(subset(groceriesrule, subset=lift > 20))
```

```
##   lhs                 rhs                          support confidence     l
ift
## 1 {bottled beer,
##    liquor}          => {red/blush wine}        0.001931876  0.4130435 21.49
356
## 2 {bottled beer,
##    red/blush wine} => {liquor}                0.001931876  0.3958333 35.71
579
## 3 {hamburger meat,
##    soda}           => {Instant food products} 0.001220132  0.2105263 26.20
919
## 4 {ham,
##    white bread}    => {processed cheese}      0.001931876  0.3800000 22.92
822
```

For the above result, I see that Instant food products are highly likely to be purchased if hamburger met and soda is purchased, and not standalone. Similarly we can see three other associations. I also noticed that these associations have RELATIVELY high confidence from the complete set of rules.

Now I want to inspect the rules more, to see interesting associations. What I realize is, getting a lot of associations in a subset is not going to give us any important interpretation, so I choose to create subset with lesser rules.

I am now trying to look for rules having confidence interval of 95%, in order to check the highest probability of buying Y when X is bought.

```
inspect(subset(groceriesrule, subset=confidence > 0.95))
```

```
##    lhs                     rhs                    support confidence     l
ift
## 1  {rice,
##     sugar}              => {whole milk}       0.001220132          1 3.913
649
## 2  {canned fish,
##     hygiene articles}   => {whole milk}       0.001118454          1 3.913
649
## 3  {butter,
##     rice,
##     root vegetables}    => {whole milk}       0.001016777          1 3.913
649
## 4  {flour,
##     root vegetables,
##     whipped/sour cream} => {whole milk}       0.001728521          1 3.913
649
## 5  {butter,
##     domestic eggs,
##     soft cheese}        => {whole milk}       0.001016777          1 3.913
649
## 6  {citrus fruit,
##     root vegetables,
##     soft cheese}        => {other vegetables} 0.001016777          1 5.168
156
## 7  {butter,
##     hygiene articles,
##     pip fruit}          => {whole milk}       0.001016777          1 3.913
649
## 8  {hygiene articles,
##     root vegetables,
##     whipped/sour cream} => {whole milk}       0.001016777          1 3.913
649
## 9  {hygiene articles,
##     pip fruit,
```

```
##     root vegetables}    => {whole milk}       0.001016777          1 3.913
649
## 10 {cream cheese ,
##     domestic eggs,
##     sugar}              => {whole milk}       0.001118454          1 3.913
649
## 11 {curd,
##     domestic eggs,
##     sugar}              => {whole milk}       0.001016777          1 3.913
649
## 12 {cream cheese ,
##     domestic eggs,
##     napkins}            => {whole milk}       0.001118454          1 3.913
649
## 13 {brown bread,
##     pip fruit,
##     whipped/sour cream} => {other vegetables} 0.001118454          1 5.168
156
```

Looking at this result, it shows whole milk is purchased with a lot of combination of products. Which totally makes sense, like people do buy milk whenever they go shopping, like even if they buy anything.

Another thing I wanted to look at, what happens when we keep a higher support value. That is of all the transactions, when I take those Xs, which are highly likely to be purchases. Also, adding to that, I took a confidence of 50%, to see the association rules, of Y with these commonly purchased Xs.

```
inspect(subset(groceriesrule, subset=support > .01 & confidence > 0.5))
```

```
##    lhs                     rhs                   support confidence     li
ft
## 1  {curd,
##     yogurt}             => {whole milk}       0.01006609  0.5823529 2.2791
25
## 2  {butter,
##     other vegetables}   => {whole milk}       0.01148958  0.5736041 2.2448
85
## 3  {domestic eggs,
##     other vegetables}   => {whole milk}       0.01230300  0.5525114 2.1623
36
## 4  {whipped/sour cream,
##     yogurt}             => {whole milk}       0.01087951  0.5245098 2.0527
47
## 5  {other vegetables,
##     whipped/sour cream} => {whole milk}       0.01464159  0.5070423 1.9843
85
## 6  {other vegetables,
##     pip fruit}          => {whole milk}       0.01352313  0.5175097 2.0253
```

```
51
## 7  {citrus fruit,
##     root vegetables}    => {other vegetables} 0.01037112  0.5862069 3.0296
08
## 8  {root vegetables,
##     tropical fruit}     => {other vegetables} 0.01230300  0.5845411 3.0209
99
## 9  {root vegetables,
##     tropical fruit}     => {whole milk}       0.01199797  0.5700483 2.2309
69
## 10 {tropical fruit,
##     yogurt}             => {whole milk}       0.01514997  0.5173611 2.0247
70
## 11 {root vegetables,
##     yogurt}             => {whole milk}       0.01453991  0.5629921 2.2033
54
## 12 {rolls/buns,
##     root vegetables}    => {other vegetables} 0.01220132  0.5020921 2.5948
90

## 13 {rolls/buns,
##     root vegetables}    => {whole milk}       0.01270971  0.5230126 2.0468
88
## 14 {other vegetables,
##     yogurt}             => {whole milk}       0.02226741  0.5128806 2.0072
35
```

This result is pretty much similar to the above one. One thing I noticed here is 'Other vegetables' are mostly purchased whenever 'root vegetables' are purchased.

Then I looked at the associations with same support , but, confidence as very low. So, I am looking at the least likely good association.

```
inspect(subset(groceriesrule, subset=support > .01 & confidence < 0.16))
```

```
##    lhs                     rhs                     support confidence
lift
## 1  {root vegetables}    => {beef}                0.01738688  0.1595149 3.04
03668
## 2  {brown bread}        => {root vegetables}     0.01016777  0.1567398 1.43
80000
## 3  {whipped/sour cream} => {citrus fruit}        0.01087951  0.1517730 1.83
37690
## 4  {root vegetables}    => {whipped/sour cream} 0.01708185  0.1567164 2.18
62496
## 5  {citrus fruit}       => {soda}                0.01281139  0.1547912 0.88
76799
## 6  {yogurt}             => {citrus fruit}        0.02165735  0.1552478 1.87
57521
```

```
## 7  {shopping bags}      => {sausage}            0.01565836  0.1589267 1.69
16065
## 8  {shopping bags}      => {yogurt}             0.01525165  0.1547988 1.10
96544
## 9  {sausage}            => {root vegetables}    0.01494662  0.1590909 1.45
95700
## 10 {soda}               => {yogurt}             0.02735130  0.1568513 1.12
43678
## 11 {whole milk}         => {soda}               0.04006101  0.1567847 0.89
91124
## 12 {other vegetables,
##     whole milk}         => {butter}             0.01148958  0.1535326 2.77
06297
```

I get a set of 12 rules, with bizzare combinations of things. And it totally makes sense that, these combinations have low confidence. Like for example, from the above result, whipped/sour cream and root vegetables are purchased together with the confidence of around 15% only.

Conclusion

There could be a lot of playing around this dataset to find interesting rules. I have presented 4 of my findings but, had looked on several others.