Exercise_1

Lucas Fernandez

8/9/2021

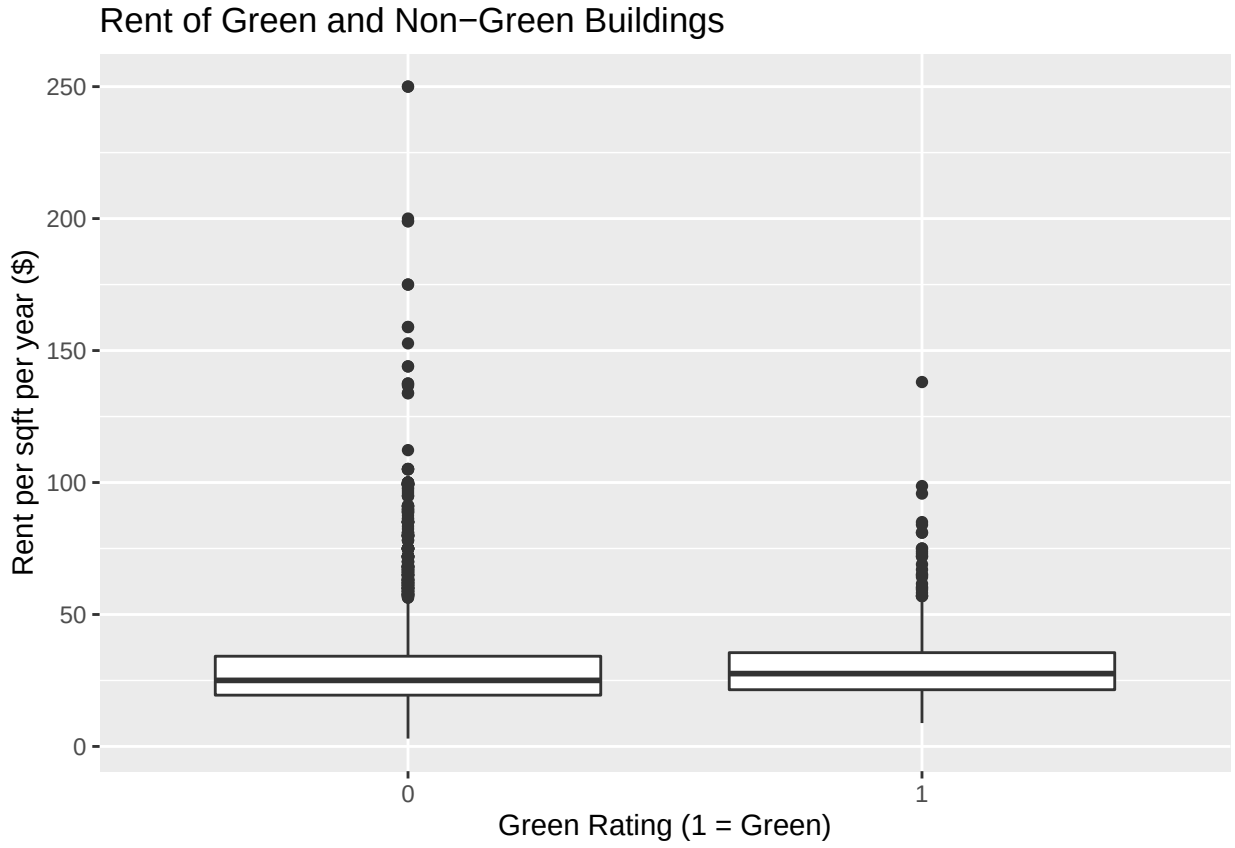
GITHUB link below

https://github.com/LucasFernandez124/STAT_380_Part_2_Exercises

Problem 1

Part a

I am going to start with following the Excel guru's path to see the process they used and then look for confounding variables and bad assumptions

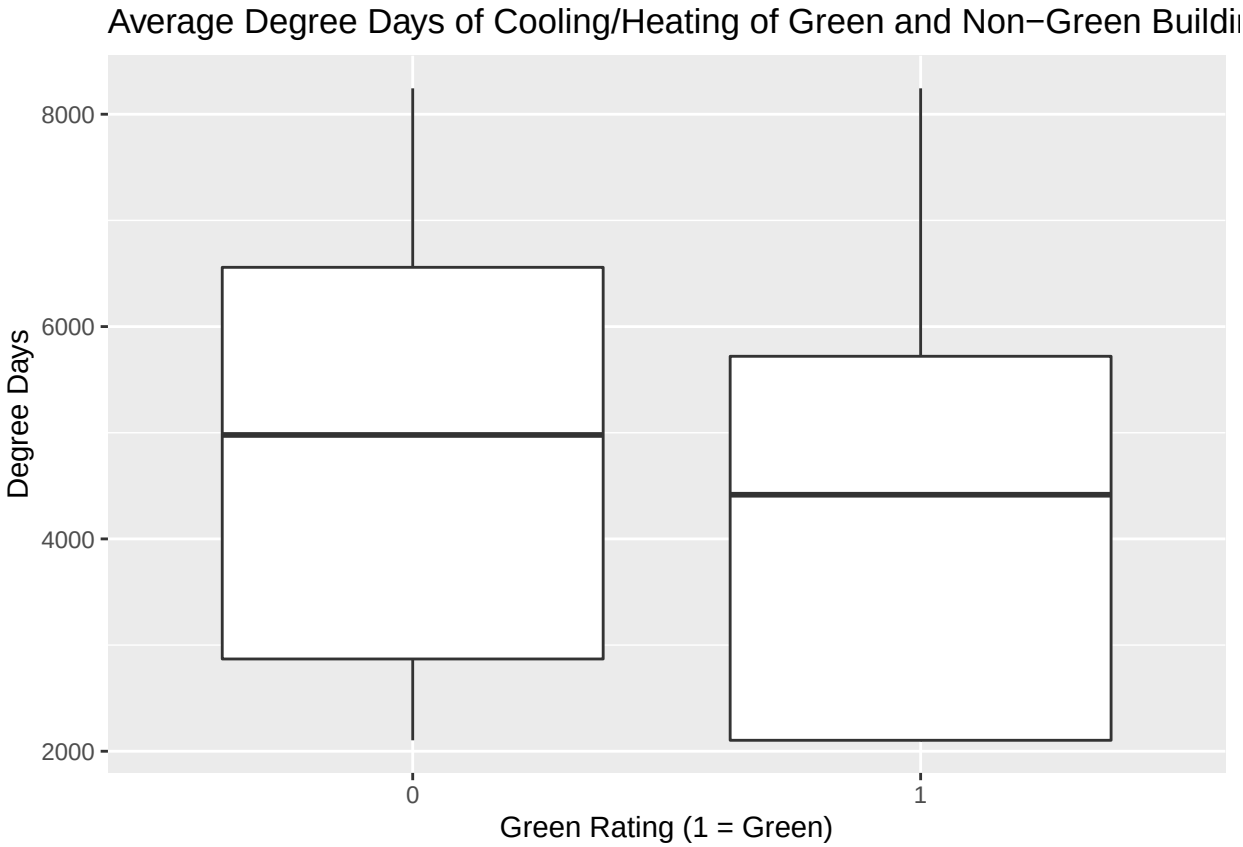


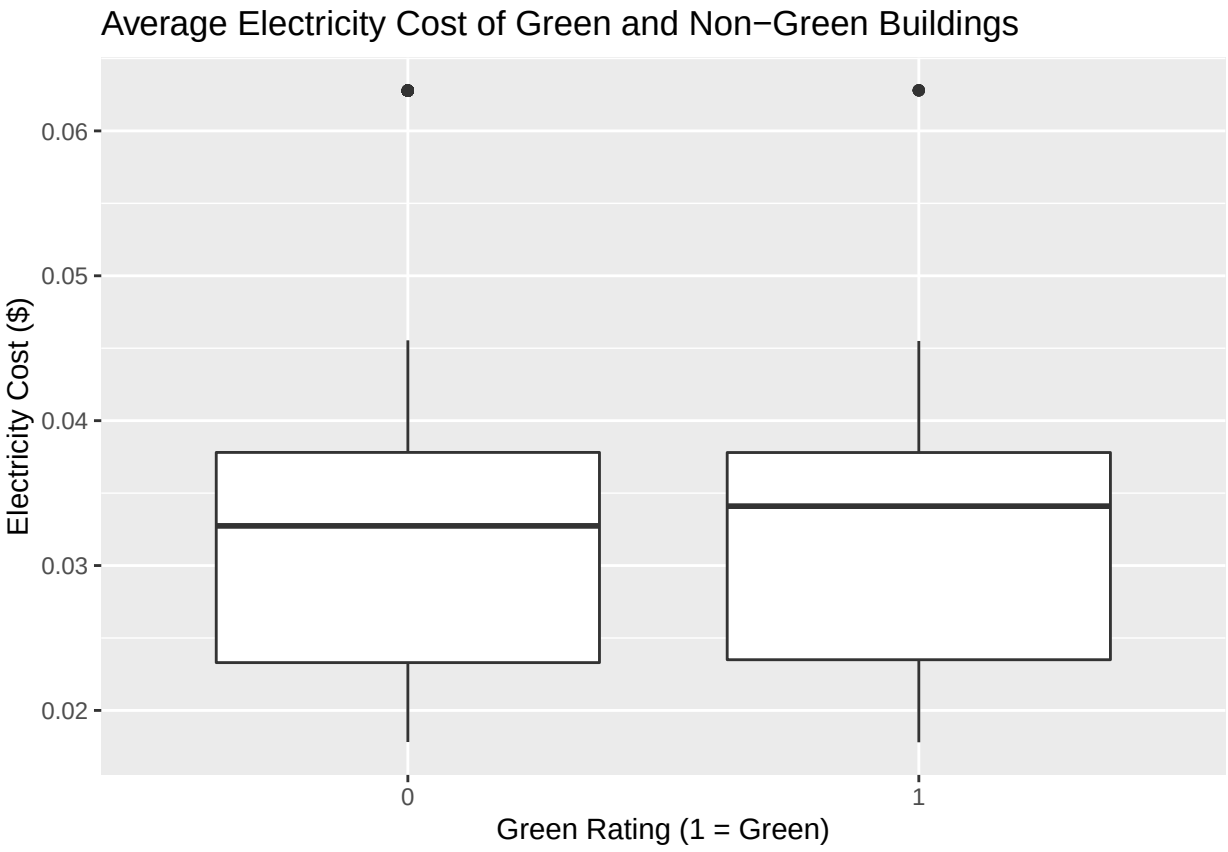
Part b

While there was nothing "wrong" with the analysis that the Excel guru did, he failed to truly address whether green building command higher rents holding all things equal. As seen in the first bar graph, the reason why rent of green buildings is higher is that rents with utilities included are much higher than their non-green building counterparts. However, rent when utilities are not included are lower for green buildings than for non-green buildings. This indicates the value of green buildings may not be naturally higher but rather than green buildings have higher utility bills (despite being green) raising rent prices when the bill is included in rent.

I looked at heating/cooling degree days and the cost of electricity to better understand the disparity in utility costs between the two sets of buildings. It seems that non-green buildings actually needed more electricity for temperature control than green buildings but the electricity costs for green buildings are higher. This higher cost for electricity likely explains why rents for green buildings, where utilities were included in rent, were so much higher than rent for non-green buildings. In conclusion, I would advise the developer that rent prices of green buildings may be higher than non-green due to the buildings having higher electricity costs. The true cost difference between green and non-green buildings is likely lower than the developer should expect and a green building may need longer to "truly" recuperate costs spent on building green.


Average Rent of Green and Non−Green Buildings split on Inclusion of Utilit
Rent ($)
250
200
150
100
50
0
0
1
Green Rating (1 = Green)
Utilities Inclusion (1 = Utils not included)
0
1


Average Degree Days of Cooling/Heating of Green and Non−Green Buildi
8000
6000
4000
2000
Degree Days
0
1
Green Rating (1 = Green)



Problem 2

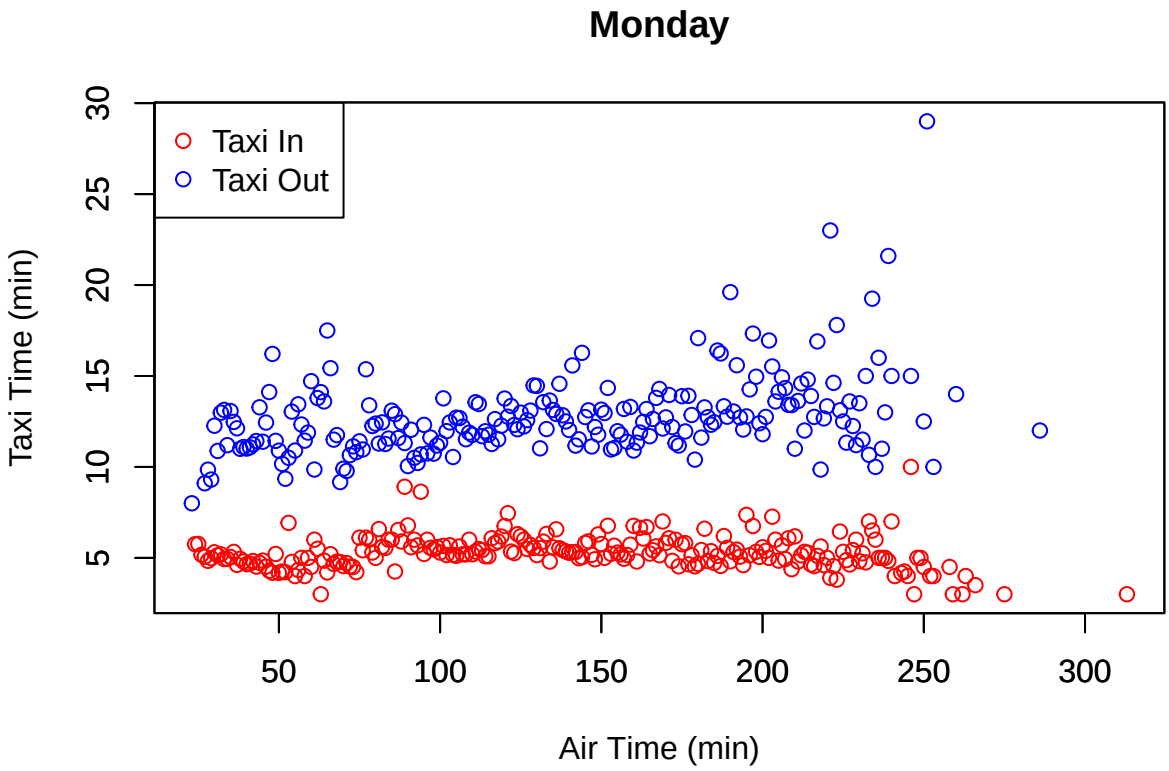
Have you ever been stuck taxi-ing at the airport after a long flight and you are wondering when you can finally stand up? Or maybe you are waiting to get up in the air before a trip thinking that you just want to get up in the air so you can make some progress to your destination.

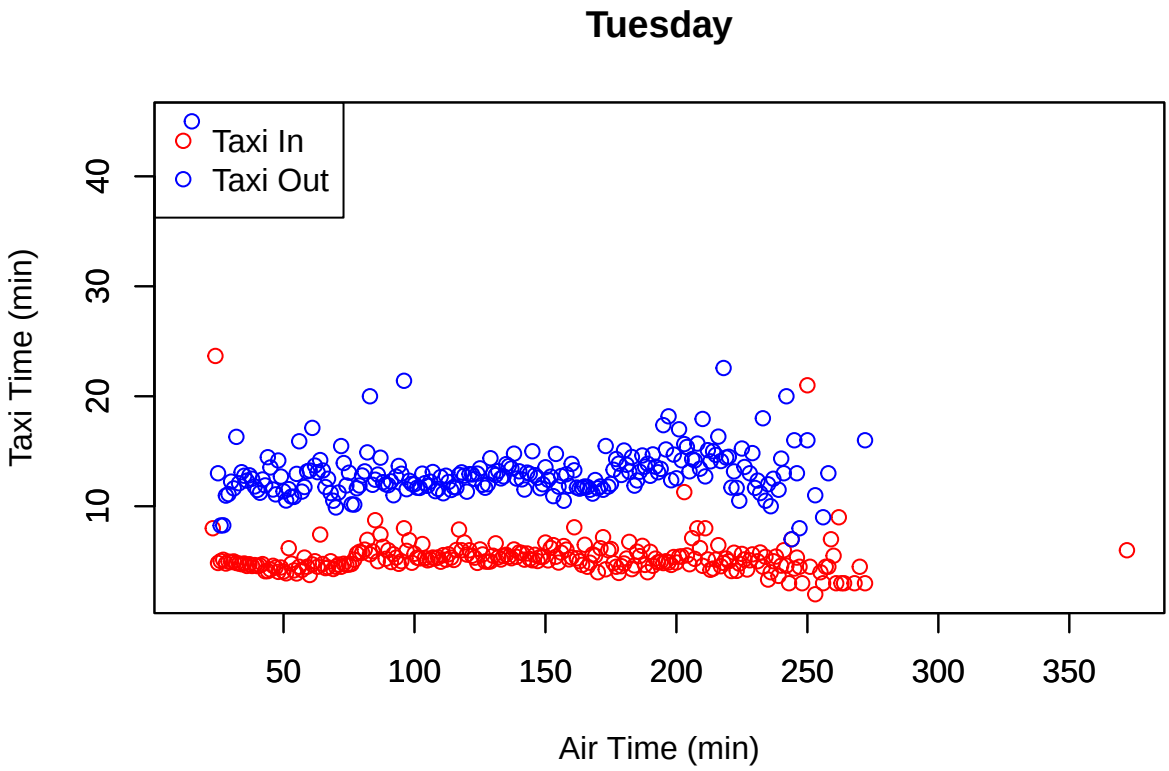
My graphs will look at a combination of variables that affect the time it takes to taxi in after a flight and the amount of time it takes to taxi out before a flight. The variables I will be looking at will be taxi time in and out versus air time while controlling for day of the week. Intuitively, different days of the week may have different average taxi times due to how busy the weekends are. I am basically trying to see if that feeling of a long taxi time after/before a long flight is true or just our imaginations.

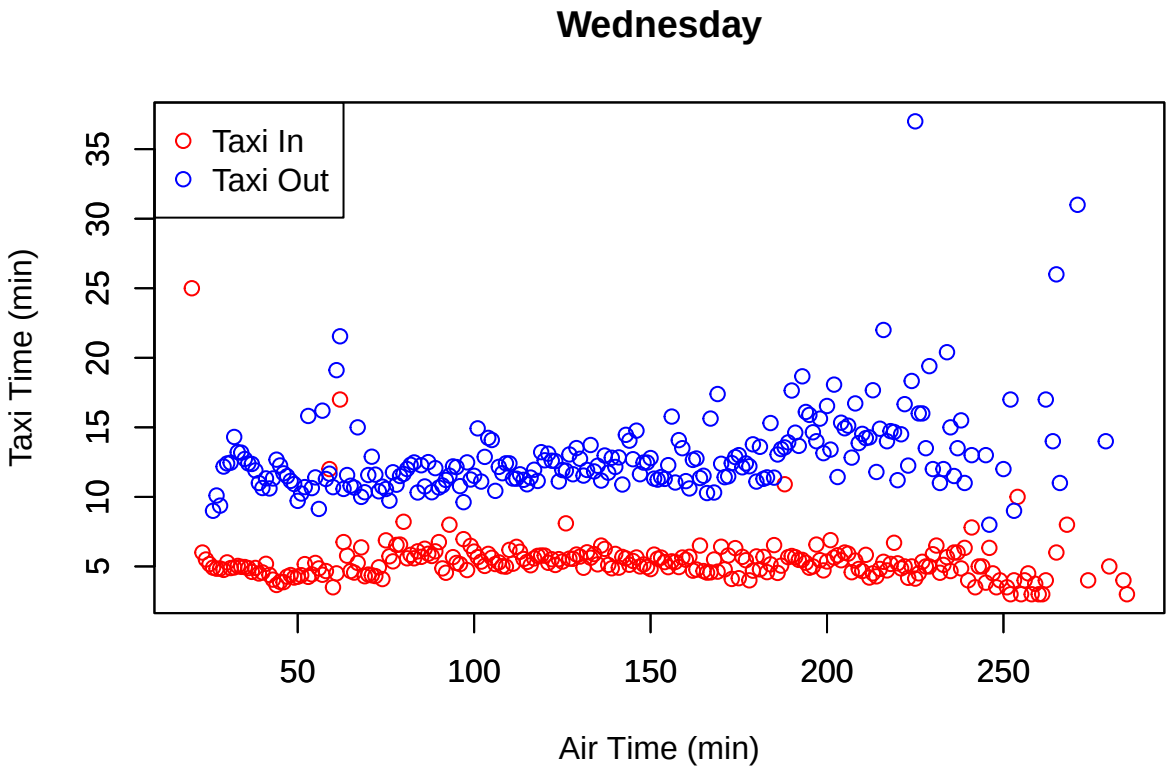
Part a

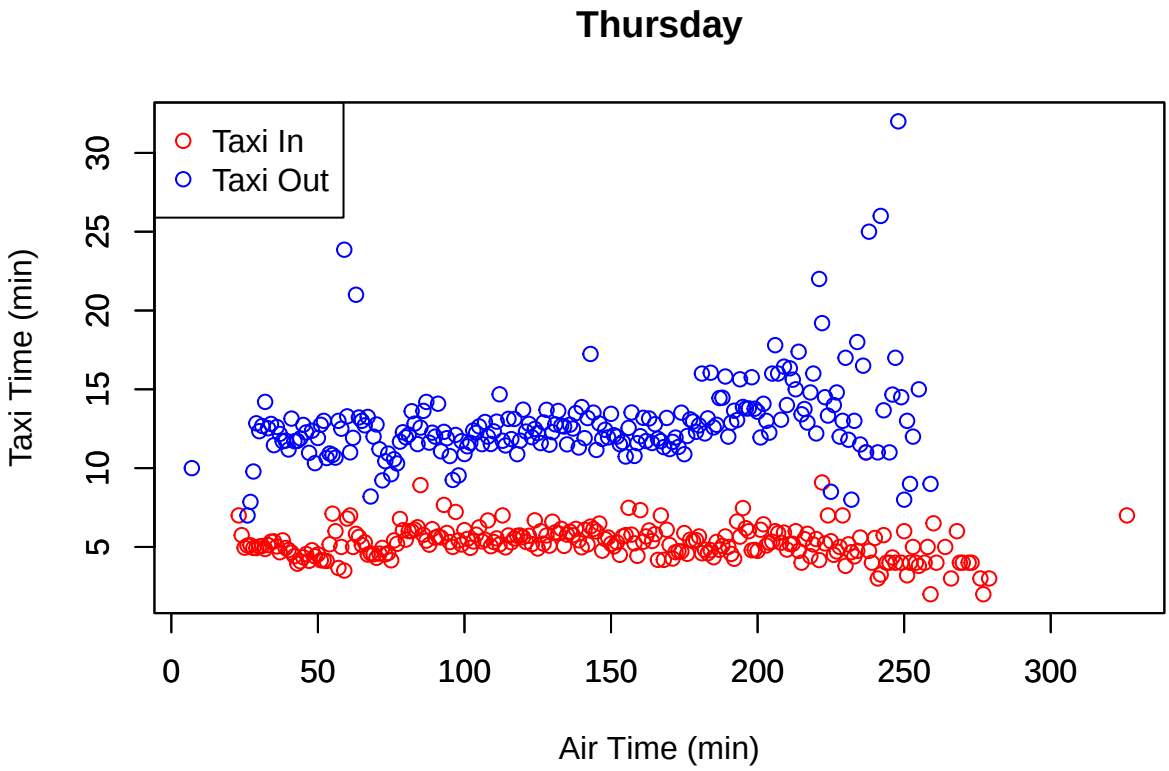
Part b

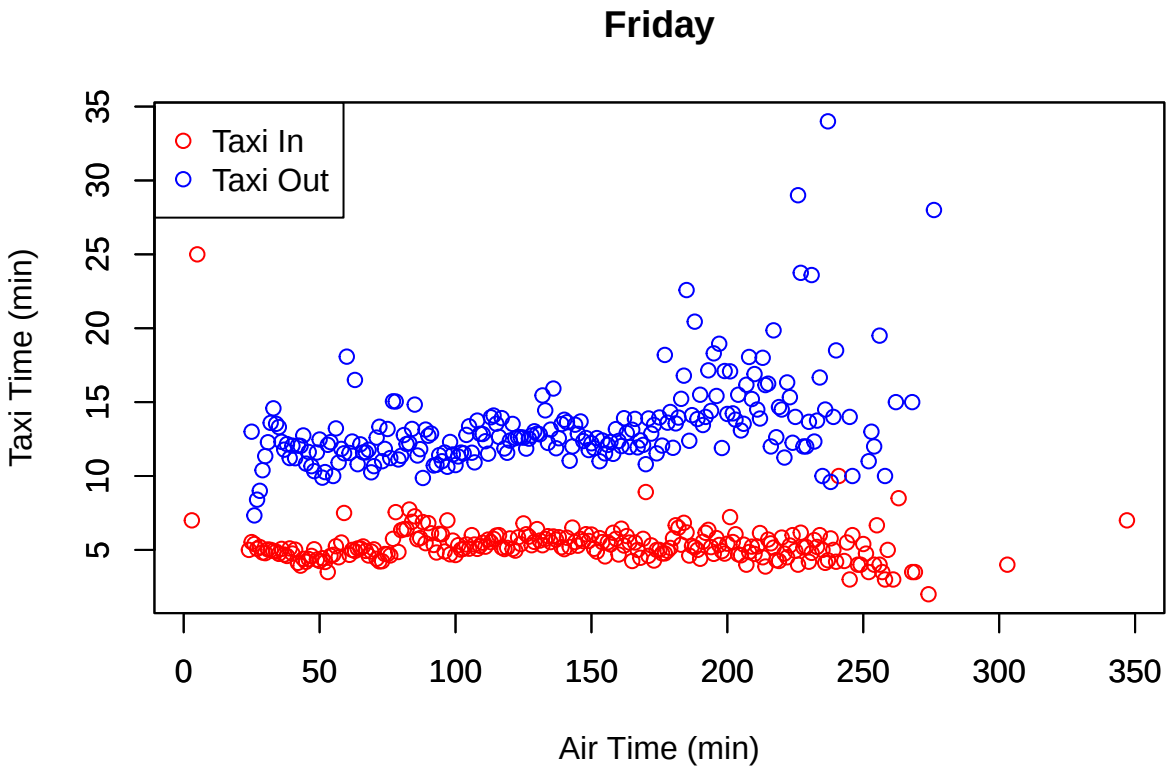
I'll let the graphs speak for themselves for the most part. I think the most interesting part is that the taxi out time is actually greater for longer flights. If you are sitting there thinking I just want to get my long flight out of the way, then you are not imagining that the wait is longer than usual. However, the opposite is true for taxing in as the taxi time is lesser for longer flights. I guess air traffic control knows people want to get off the plane ASAP.

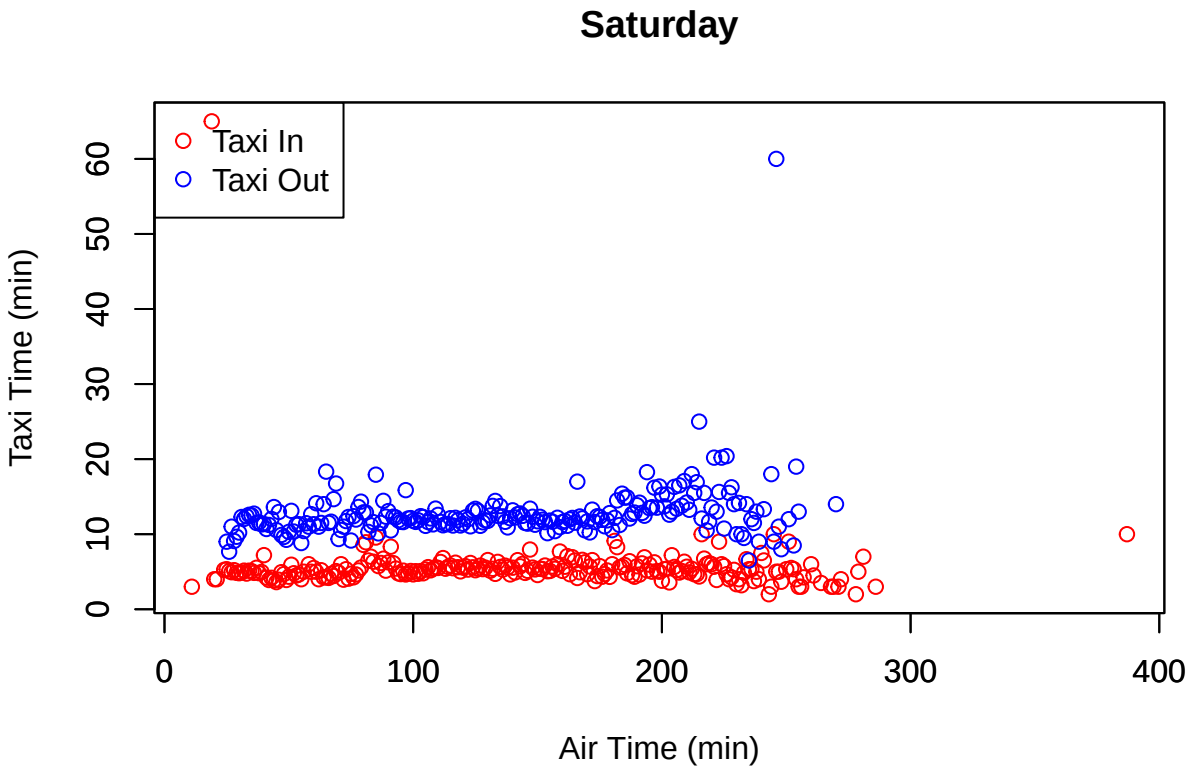

Monday
Taxi In
Taxi Out
Taxi Time (min)
5
10
15
20
25
30
50
100
150
200
250
300
Air Time (min)

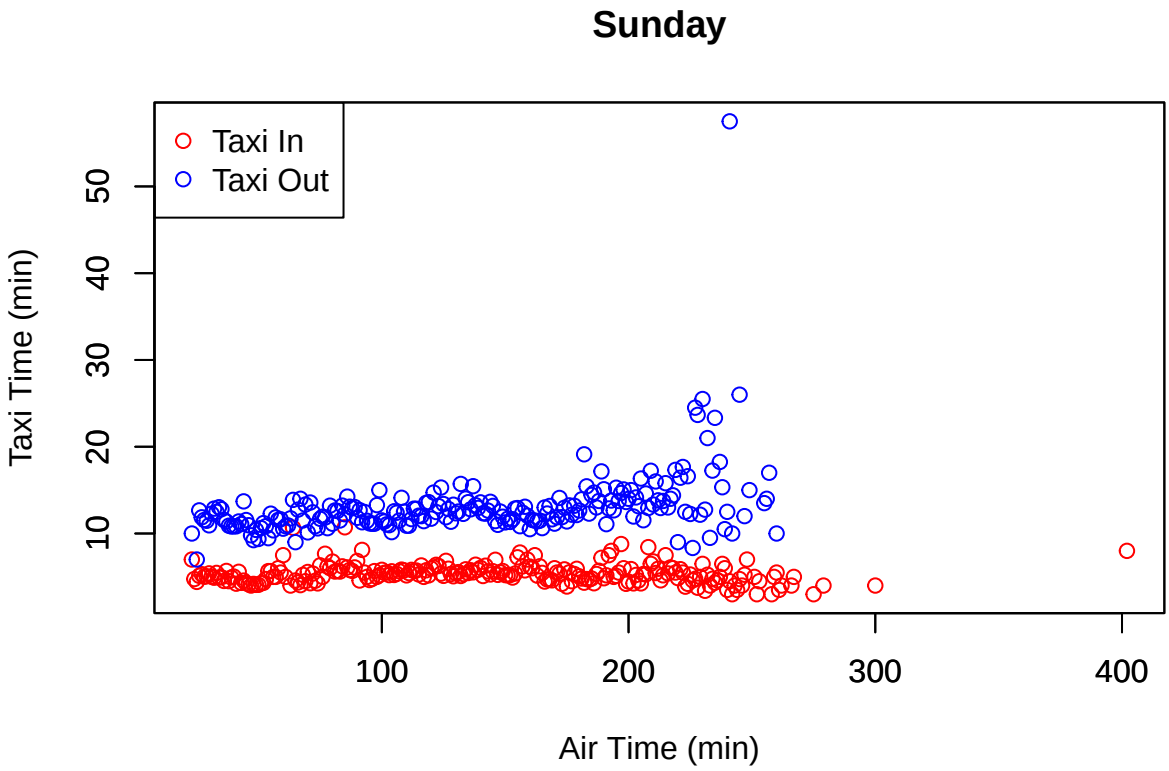

Tuesday
Taxi In
Taxi Out
Taxi Time (min)
10
20
30
40
50
100
150
200
250
300
350
Air Time (min)


Wednesday
Taxi In
Taxi Out
Taxi Time (min)
5
10
15
20
25
30
35
50
100
150
200
250
Air Time (min)


Thursday
Taxi In
Taxi Out
Taxi Time (min)
5
10
15
20
25
30
0
50
100
150
200
250
300
Air Time (min)


Friday
Taxi In
Taxi Out
Taxi Time (min)
5
10
15
20
25
30
35
0
50
100
150
200
250
300
350
Air Time (min)


Saturday
Taxi In
Taxi Out
Taxi Time (min)
0
10
20
30
40
50
60
0
100
200
300
400
Air Time (min)



Problem 3

Part a

I am going to begin by downloading the daily data for the last five years of my data. I will be making three portfolios for the portfolio modeling based on three different areas of the world. The three areas of the world are Asia Pacific, Europe, and Latin America. I will be using the five ETF's with the highest total assets (as long as they have five years of data). I think it will be interesting using the different geographic areas. Europe represents a developed market, Asia Pacific represents a market with some developed countries and some developing, and Latin America represents developing markets.

```
## Registered S3 method overwritten by 'mosaic':
##   method                           from
##   fortify.SpatialPolygonsDataFrame ggplot2
```

```
##
## The 'mosaic' package masks several functions from core packages in order to add
## additional features.  The original behavior of these functions should not be affected by this.
```

```
##
## Attaching package: 'mosaic'
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
##     count, do, tally

## The following object is masked from 'package:Matrix':
##
##     mean

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

## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
## [1] "EWT"  "EWY"  "INDA" "AAXJ" "VPL"
```

```
## [1] "VGK"  "EZU"  "IEUR" "EWU"  "FEZ"
```

```
## [1] "EWZ"  "ILF"  "EWW"  "EWZS" "BRF"
```

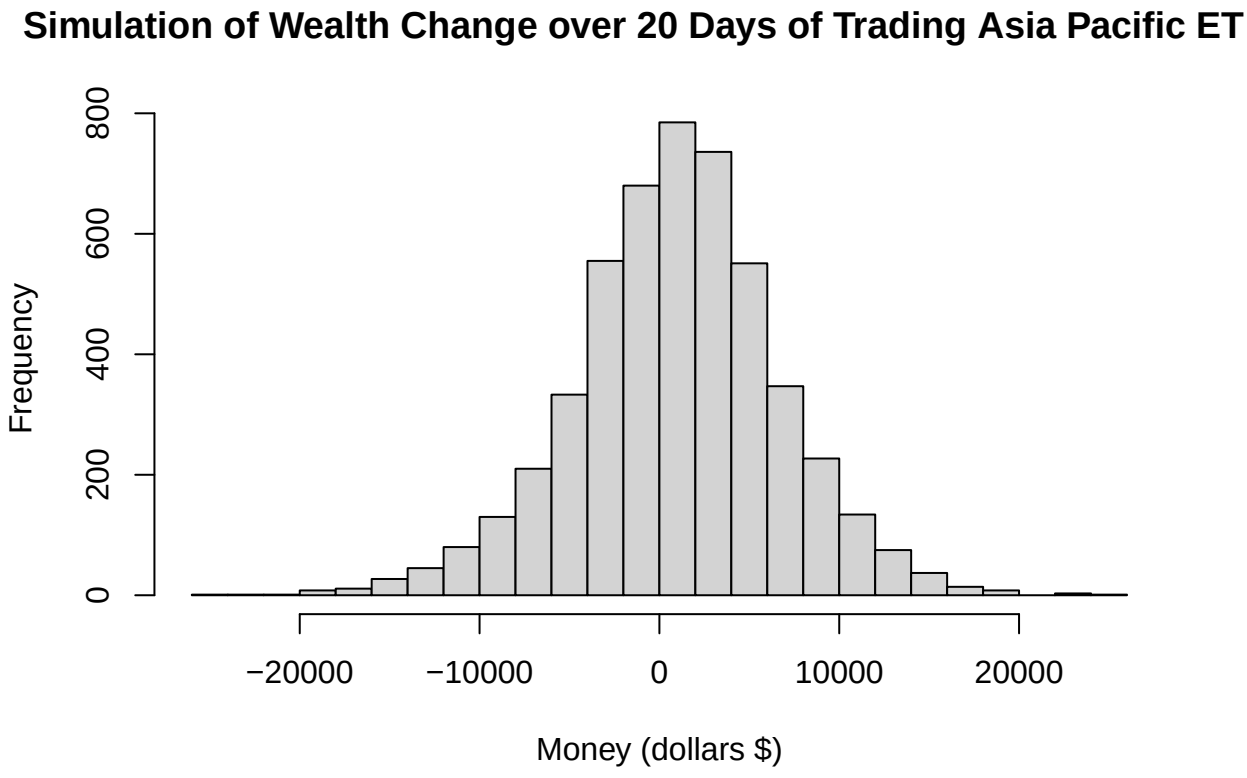
Part b

The simulated portfolios of the geographic ETF's more or less returned the results expected of them based on the level of development of countries included in the ETF's. In terms of the value added to initial wealth, the developed market of Europe had the lowest growth while the developing market of Latin America had the highest growth. This makes sense with the traditional understanding that developed economies have slower growth and thus their stock markets grow at a slower rate. The value added for Asia Pacific, Europe, and Latin America are $840, $791, and $966, respectively.

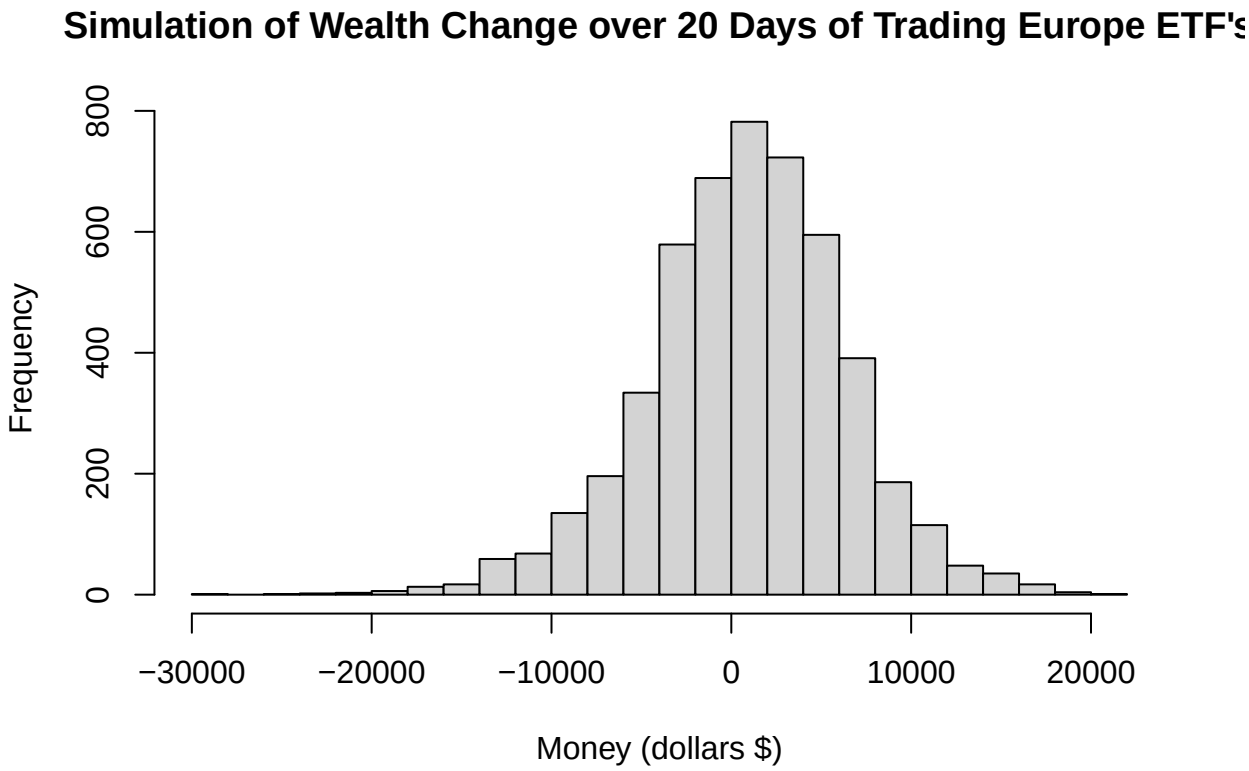
Surprisingly for value at risk (VaR), the results are different than expected. The VaR of European ETF's was higher than that of Asia Pacific. This indicates that, despite being a developed market, Europe may be a bad place to put investments in ETF's compared to even somewhat developed markets like Asia Pacific. Unsurprisingly, the VaR of Latin America was the highest. This indicates that high growth markets, such as Latin America, are subject to more risk. The VaR for Asia Pacific, Europe, and Latin America are $8456, $8720, and $14577, respectively.

```
## The mean of the simulation of Asia Pacific ETFs is 100970.5
```

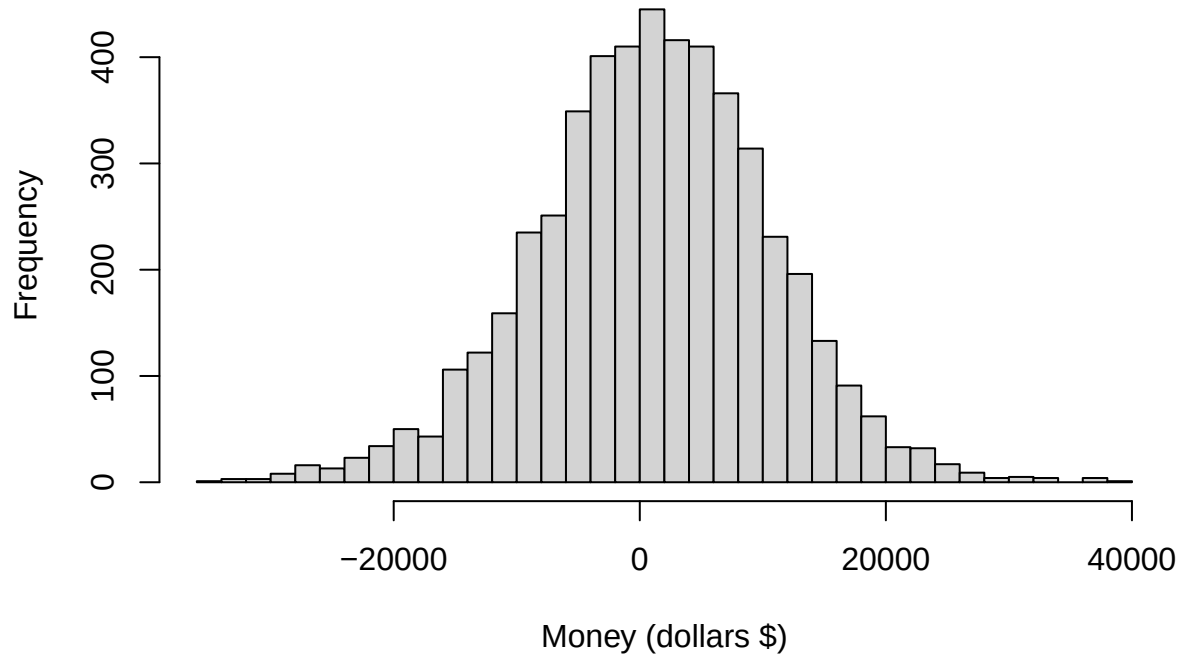
```
## The mean difference from initial wealth of the simulation of Asia Pacific ETFs is 970.5244
```



```
## The value at risk of the Asia Pacific ETFs is 8616.097

## The mean of the simulation of Europe ETFs is 100838.1

## The mean difference from initial wealth of the simulation of Europe ETFs is 838.12
```




```
## The value at risk of the Europe ETFs is 8796.247

## The mean of the simulation of Latin American ETFs is 101070.8

## The mean difference from initial wealth of the simulation of Latin American ETFs is 1070.789
```




```
## The value at risk of the Latin American ETFs is 14903.62
```

Problem 4

Part a

I am going to use Kmeans++ clustering to get "clusters" of NutrientH2O's market segments. Hopefully, these clusters give us the different "types" of people interested in their products. I am going to begin by processing the data. Then I will run Kmeans++ with different K's and visualize the data. Finally, I will use the gap statistic to get my "best" K and then compare with previous visualizations. I will use my best K to analyze what clusters or market segments are buying Nutrient H2O's products.

Part b

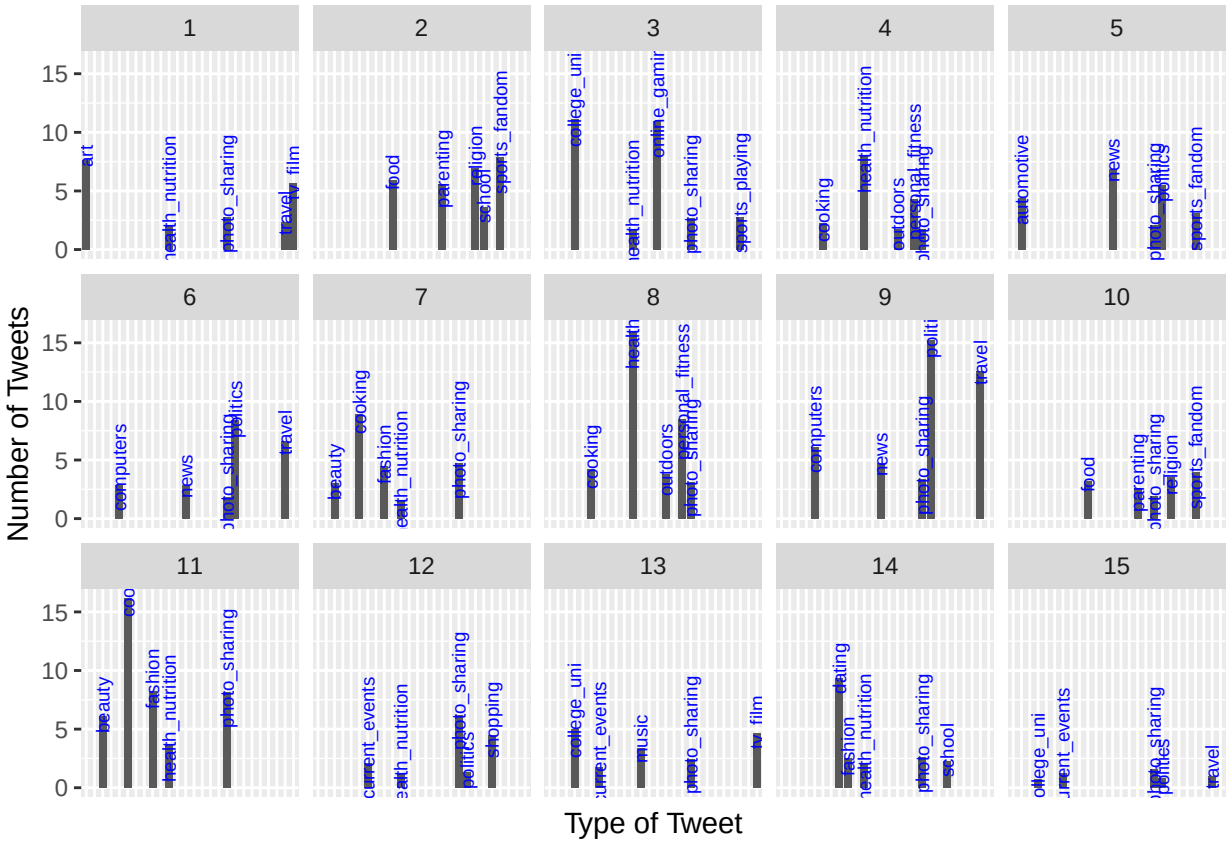
I am just going to quickly analyze the first Kmeans++ of K=5 because this model is not our final model. The first cluster appears to be users who may not tweet much because they are in college and in do gaming/shopping. This first cluster is likely a market segment of typical college students. The second cluster and third clusters are similar but differ in that the second cluster has outdoors and personal fitness and the third cluster has fashion and beauty. The second cluster is likely a market segment of fitness buffs, people who like to go out and hike/jog outside. The third cluster is likely people who spend too much time on twitter and instagram posting pictures of themselves, the Kim Kardashian type of person.

```
## [1] "K=5"
```

```
## [1] "cluster 1 max values"

##  online_gaming       shopping current_events    college_uni  photo_sharing
##       1.168458       1.269740       1.449153       1.511782       2.311906

## [1] "cluster 2 max values"

##        school     parenting          food      religion sports_fandom
##      2.687817      4.016497      4.535533      5.241117      5.864213

## [1] "cluster 3 max values"

##     computers photo_sharing          news        travel      politics
##      2.467133      2.514685      5.195804      5.566434      8.823776

## [1] "K=3"

## [1] "cluster 1 max values"

##           travel         politics   current_events health_nutrition
##         1.240048         1.360467         1.384433         1.435334
##    photo_sharing
##         2.052882

## [1] "cluster 2 max values"

##        school     parenting          food      religion sports_fandom
##      2.669502      4.001215      4.534629      5.247874      5.844471

## [1] "cluster 3 max values"

##         politics personal_fitness    photo_sharing          cooking
##         2.995522         3.115423         4.355224         4.774129
## health_nutrition
##         5.611940

## [1] "K=10"

## [1] "cluster 1 max values"

##        school     parenting          food      religion sports_fandom
##      2.764350      4.249245      4.712991      5.543807      6.178248

## [1] "cluster 2 max values"

##        travel photo_sharing   college_uni           art       tv_film
##      2.085366      2.519512      2.614634      5.039024      5.646341

## [1] "cluster 3 max values"

## health_nutrition    photo_sharing   sports_playing    online_gaming
##         1.776504         2.684814         2.750716        10.925501
##      college_uni
##        11.154728
```

Part c

The market segmentation of twitter followers is shown in the facet plot below. The facet plot contains a bar chart for each cluster that shows the five largest tweet types for that cluster. Just to look at one of the clusters as an example, the cluster with high college and online gaming tweets is probably a college student who plays video games in his free time. Another cluster, would be the one high in food, school, religion, and parenting which is likely WASP (White Anglo-Saxon Protestant) type parents. Some of the clusters have very low values for all their tweet types and these are likely the "in-between" clusters that group values between the clusters that are more cohesive. (*as a note I think Rmarkdown is messing with the graph labels a bit since the aspect ratio is smaller than normal)



Problem 5

Part a

I will start by getting the texts for documents in the training set. I will then do preprocessing including applying PCA on the TF-IDF matrix in the next chunk. After preprocessing, I will move on to applying the same preprocessing to the testing set. From there, I will apply the KNN model to try to predict the author of an article from it's nearest neighbors.

For the chunk below, I was focused on creating my corpus from the documents. Due to where the files are I had to do some funky working directory calls but I believe that it all ends up being good in the end.

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

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

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

##
## Attaching package: 'tm'

## The following object is masked from 'package:mosaic':
##
##     inspect

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.2     v purrr   0.3.4
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x purrr::accumulate()        masks foreach::accumulate()
## x NLP::annotate()            masks ggplot2::annotate()
## x mosaic::count()            masks dplyr::count()
## x purrr::cross()             masks mosaic::cross()
## x mosaic::do()               masks dplyr::do()
## x tidyr::expand()            masks Matrix::expand()
## x dplyr::filter()            masks stats::filter()
## x xts::first()               masks dplyr::first()
## x ggstance::geom_errorbarh() masks ggplot2::geom_errorbarh()
## x dplyr::lag()               masks stats::lag()
## x xts::last()                masks dplyr::last()
## x tidyr::pack()              masks Matrix::pack()
## x mosaic::stat()             masks ggplot2::stat()
## x mosaic::tally()            masks dplyr::tally()
## x tidyr::unpack()            masks Matrix::unpack()
## x purrr::when()              masks foreach::when()

##
## Attaching package: 'proxy'

## The following object is masked from 'package:Matrix':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix

## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
## Content:  documents: 2500
```

Part b

The notable take away from the chunk below is the factors I choose for the sparcity of my matrix and the number of dimensions I choose to reduce to. I choose a sparcity of .975 meaning any words that do not appear in 63 documents will be removed from the matrix. I think there is a limitation in that a word could be used a lot by one author but not a lot by the other authors. 63 documents is more than any one author has written. The number of dimensions chosen for the PCA is 80 dimensions which explains about 40% of the variance. The reason and limitations behind these choices will be explained a bit later.

```
##
## Docs           PC1           PC2           PC3           PC4         PC5
##    1  0.0002577571  0.0006633441 -0.0015263021 -0.0011366902 0.003465915
##    2 -0.0001771529  0.0008144799 -0.0029025628 -0.0006195011 0.004090249
##    3  0.0005955053 -0.0002204849  0.0004264007 -0.0013001268 0.001365680
##    4  0.0008436952 -0.0003899458 -0.0026435509 -0.0003595956 0.001991451
##    5  0.0008031792 -0.0003842024 -0.0017917593 -0.0004695679 0.001862147
##
## Docs           PC6           PC7          PC8          PC9         PC10
##    1 -1.021093e-03 0.0003226772  0.001749198 -0.002472621 0.0008604453
##    2  1.415856e-05 0.0003034156  0.003261147 -0.002493020 0.0010640611
##    3 -8.117216e-04 0.0019239380 -0.001471050 -0.001823169 0.0005059647
##    4  2.075056e-03 0.0013781305  0.003319290 -0.002904969 0.0036811328
##    5  1.252307e-03 0.0016385622  0.001816363 -0.003006217 0.0032603480
##
## Docs          PC11          PC12          PC13          PC14          PC15
##    1 -1.275090e-03 3.635381e-04 -4.151017e-04 -0.0001099100  0.0008979087
##    2 -4.784877e-05 1.744738e-05  5.442719e-05  0.0007500033  0.0004917589
##    3 -1.332058e-03 1.267452e-04 -9.727775e-04 -0.0011155890  0.0006524108
##    4 -2.691776e-03 1.366553e-03 -1.577156e-03 -0.0010304647 -0.0013769395
##    5 -2.376489e-03 7.113242e-04 -1.281952e-03 -0.0006570218 -0.0004141805
##
## Docs          PC16          PC17          PC18          PC19         PC20
##    1  0.0001532739  0.0000301379  0.0013366063 -0.0012048720 0.0027717352
##    2  0.0002783138  0.0000504977  0.0002565577 -0.0011728421 0.0016752855
##    3 -0.0005877939 -0.0005347289  0.0003238362 -0.0010614444 0.0001362835
##    4  0.0004089983 -0.0004951764 -0.0025006105  0.0005374984 0.0026739887
##    5  0.0002912566 -0.0001387252 -0.0017202761 -0.0003748316 0.0026444813
##
## Docs          PC21          PC22          PC23          PC24         PC25
##    1  0.0012112690 -0.0002014216 -0.0015814563  3.647213e-04 7.086159e-04
##    2  0.0004950811  0.0005354989 -0.0017276111  2.471994e-05 1.600307e-03
##    3 -0.0007185361  0.0002498793 -0.0005830116  3.017786e-04 3.092295e-04
##    4 -0.0015279054 -0.0022582519 -0.0023863842 -2.463653e-03 6.205158e-04
##    5 -0.0011212464 -0.0012397592 -0.0017270022 -1.970424e-03 6.753497e-05
##
## Docs          PC26          PC27          PC28          PC29          PC30
##    1  0.0008658066 0.0024000276 -1.452467e-05 -0.0016384646 -1.771795e-03
##    2  0.0013189670 0.0013465693 -5.881295e-04 -0.0021258892 -1.922372e-03
##    3 -0.0004177983 0.0004827385  2.820927e-04 -0.0001485861 -3.265248e-04
##    4  0.0023749049 0.0051091335 -5.680310e-04  0.0003592184  3.303293e-04
##    5  0.0015917507 0.0041683278  2.177772e-04  0.0003244947  8.934522e-05
##
## Docs          PC31          PC32          PC33          PC34          PC35
##    1  4.410044e-04  0.0008426972 -1.523149e-04 -0.0007468991  0.0007050651
```

```
##    2  1.802110e-03  0.0023646294  9.288544e-05 -0.0009939327  0.0023013378
##    3  6.821039e-05  0.0004133910  6.925561e-04  0.0004064371 -0.0006978602
##    4 -9.246428e-04 -0.0040886392 -2.480011e-06  0.0028941170  0.0051647492
##    5 -3.495579e-04 -0.0031430905  2.124080e-05  0.0020768428  0.0037711759
##
## Docs          PC36          PC37          PC38          PC39          PC40
##    1  0.0006206781 -0.0009369781  0.0019501654 -0.0006729226 -0.0001509943
##    2  0.0007939533  0.0002640636  0.0012356199 -0.0006853796  0.0012925008
##    3 -0.0003905790 -0.0006586386  0.0012806578 -0.0001542362 -0.0007626668
##    4 -0.0028179371  0.0011251323 -0.0008111396  0.0020783823  0.0007534941
##    5 -0.0021008516  0.0008301907 -0.0004472236  0.0011925343  0.0006152223
##
## Docs          PC41          PC42          PC43         PC44          PC45
##    1 -6.306096e-04  0.0009912563  0.0015761838 1.166219e-03 -0.0013013930
##    2 -8.989249e-04  0.0011319697  0.0017146791 1.248559e-03 -0.0005516220
##    3  7.938458e-05  0.0006367009 -0.0005388579 4.772049e-05 -0.0001213916
##    4 -2.136000e-03 -0.0013005433 -0.0014761134 2.861928e-03  0.0005378748
##    5 -1.610887e-03 -0.0009277802 -0.0008994233 2.061293e-03 -0.0002957460
##
## Docs         PC46          PC47          PC48          PC49          PC50
##    1 0.0011110837 -9.176603e-05  0.0007271985 -7.836379e-05  0.0008414664
##    2 0.0005926856  1.936960e-04  0.0010312467  7.934573e-04 -0.0012347701
##    3 0.0002775758 -3.983484e-04 -0.0003621856  1.745823e-04  0.0001131771
##    4 0.0103676760 -8.326489e-03  0.0029035006 -3.412995e-04  0.0047395447
##    5 0.0074216215 -6.320590e-03  0.0020521129 -2.483532e-04  0.0035204148
##
## Docs          PC51          PC52          PC53          PC54          PC55
##    1  1.018348e-05 -6.574988e-04 -0.0008168494  0.0005808754  0.0010288870
##    2  2.152016e-04 -1.301946e-04  0.0002313982 -0.0004387357  0.0002744467
##    3 -1.372302e-04 -2.733957e-04 -0.0004994937 -0.0003947334 -0.0001899102
##    4  1.234965e-03  3.572444e-05  0.0020508446 -0.0025399292 -0.0019963505
##    5  8.413007e-04 -7.505688e-04  0.0016896830 -0.0017957748 -0.0007456234
##
## Docs          PC56          PC57          PC58         PC59          PC60
##    1  3.952303e-04 -0.0002544917  0.0004975825 -0.001148695  0.0001275939
##    2 -4.246262e-04  0.0002618947 -0.0003784814  0.000341262 -0.0012088194
##    3  3.603539e-04  0.0005053772 -0.0004656418  0.000583494  0.0006248742
##    4 -1.162354e-03  0.0009751003 -0.0060139844 -0.002261211  0.0031542204
##    5 -7.684031e-05  0.0012051239 -0.0045663795 -0.000690428  0.0014881897
##
## Docs          PC61          PC62          PC63          PC64          PC65
##    1  4.843232e-04  0.0004617346 -7.531077e-04  0.0004845933 -1.214018e-03
##    2 -9.771620e-04 -0.0010128878 -4.456029e-05 -0.0022559857  1.413605e-03
##    3 -5.657962e-04  0.0003472758 -6.560913e-04  0.0004590475 -4.947931e-05
##    4 -1.971640e-06  0.0021035796  2.219923e-04 -0.0031228669  2.575821e-03
##    5  9.736555e-05  0.0015911523 -2.225774e-04 -0.0025836986  2.399191e-03
##
## Docs         PC66          PC67          PC68          PC69          PC70
##    1 0.0006746093  1.100035e-03 -0.0012183641 -2.963814e-03 -6.866379e-05
##    2 0.0009035323 -1.411619e-03  0.0004519284  7.199797e-04  7.626564e-04
##    3 0.0003301128 -7.443424e-04 -0.0002463692 -8.542201e-05 -2.272607e-04
##    4 0.0036439921 -1.981215e-04 -0.0052638071  4.004959e-04 -1.572821e-03
##    5 0.0029543119  8.047849e-05 -0.0041133299  4.384082e-04 -1.331057e-03
##
```

```
## Docs          PC71          PC72          PC73          PC74          PC75
##    1 -0.0002340573  7.135404e-05  0.0007889806 -1.397094e-03 -9.039752e-04
##    2 -0.0009550298 -1.112166e-03  0.0001856242  7.362129e-05  8.780154e-04
##    3  0.0003208421  2.226942e-04  0.0004895186  7.383601e-04 -3.358765e-04
##    4  0.0044184717 -3.214812e-04 -0.0005247921  1.476771e-03  7.618332e-05
##    5  0.0028784646 -3.077911e-05  0.0004987829  2.008375e-03 -2.713581e-04
##
## Docs          PC76          PC77          PC78          PC79          PC80
##    1 -0.0009825660  1.404055e-03 -0.0002677716 -0.0004458773  0.0008683090
##    2  0.0010777961  1.061698e-03 -0.0006723257 -0.0020537008  0.0011204617
##    3  0.0008773744 -5.076061e-06 -0.0001381561 -0.0001671596 -0.0001671682
##    4  0.0058270384  8.517499e-04  0.0013423714  0.0015314371  0.0005319261
##    5  0.0036305446  1.715869e-03  0.0010981014  0.0005495727  0.0010596603
```

Part c

```
##
## Docs           PC1           PC2           PC3           PC4           PC5
##    1  0.0008966868  6.637073e-05 -0.0005268501 -9.950888e-04 -0.0008177394
##    2  0.0011475770 -1.268045e-03 -0.0001407575 -7.846740e-05  0.0008530545
##    3 -0.0028750986  9.193457e-03 -0.0134794380  6.442734e-03  0.0144446409
##    4  0.0008854198 -1.616914e-04 -0.0009428755  1.409142e-04  0.0016913331
##    5  0.0011892712  7.379597e-04 -0.0068817367  5.397483e-05  0.0090946629
##
## Docs           PC6           PC7           PC8           PC9          PC10
##    1  0.0013096721 -0.0010293287  0.0003787526 -0.0014400637 0.0038722959
##    2 -0.0000813146  0.0009041184 -0.0009340833 -0.0006090625 0.0012101665
##    3  0.0029963372 -0.0080758753  0.0157996839 -0.0050835940 0.0003460811
##    4  0.0011461757 -0.0001352851  0.0012172986 -0.0028077289 0.0022661470
##    5  0.0020440459 -0.0021631566  0.0105991262 -0.0043061708 0.0025545809
##
## Docs          PC11          PC12          PC13          PC14          PC15
##    1 -0.003991598  0.0004615621 -0.0017868255  1.075933e-04 -0.0020504425
##    2 -0.002082091  0.0009382981 -0.0004002533 -6.097292e-04 -0.0009805355
##    3 -0.005025037 -0.0036473777  0.0044031293  3.480986e-03  0.0019675076
##    4 -0.002077795  0.0008593669 -0.0012222275 -1.232817e-05 -0.0015729855
##    5 -0.001642071  0.0017894399  0.0012758073  2.420483e-03 -0.0011263447
##
## Docs          PC16          PC17          PC18          PC19          PC20
##    1  8.017531e-04  0.0023639698 -0.0010087430 -0.0014821814 0.0015067701
##    2 -8.845753e-04 -0.0011216333  0.0001679135 -0.0010289622 0.0009234458
##    3  1.607007e-03  0.0015438170  0.0029747238  0.0060365043 0.0102949985
##    4 -1.678417e-04 -0.0008544737 -0.0017952529 -0.0002692501 0.0006281845
##    5  1.749179e-05 -0.0018298853 -0.0011148376  0.0005106724 0.0043834558
##
## Docs          PC21          PC22          PC23          PC24          PC25
##    1 -1.939272e-05  0.0014534621 -3.629951e-04 -0.0008140837  4.711057e-04
##    2 -5.251861e-04  0.0006017764  7.925259e-05 -0.0003101270  3.368106e-05
##    3  3.051815e-04 -0.0060887683 -7.410304e-03  0.0059322578  4.906153e-03
##    4 -1.013897e-03  0.0002942905 -5.198425e-04  0.0012379344 -1.450102e-03
##    5  7.497464e-04 -0.0031493559 -5.849556e-03  0.0050210986  3.555267e-03
##
## Docs          PC26          PC27          PC28          PC29          PC30
```

```
##    1 -3.114555e-04 -0.0004655092 -0.0010347551  0.0023656289  1.946473e-03
##    2  1.305721e-04 -0.0001222300  0.0002526394 -0.0005562105 -9.511476e-05
##    3  2.825848e-03  0.0039240950 -0.0011437962  0.0045301419  3.485150e-03
##    4 -4.572641e-05 -0.0003457171 -0.0004279987 -0.0012319125 -1.070163e-03
##    5  3.497380e-03  0.0053640923 -0.0013927939 -0.0033792777 -4.285770e-03
##
## Docs          PC31          PC32         PC33          PC34          PC35
##    1 -4.666742e-04 -2.700699e-03 0.0004687860  0.0004126960  0.0004180725
##    2 -5.068303e-04  5.677215e-05 0.0004039588  0.0009181899 -0.0004016711
##    3  5.871445e-05  5.651580e-03 0.0001231961  0.0011978585 -0.0014269209
##    4  9.155544e-04  8.806614e-04 0.0029235401 -0.0006416410  0.0003509469
##    5  1.238521e-03  5.425167e-03 0.0012267556  0.0001723294  0.0031721447
##
## Docs          PC36          PC37          PC38          PC39          PC40
##    1  1.029075e-03 -1.775307e-03 -0.0013100665 -0.0014179058  1.128463e-03
##    2 -3.461009e-04 -2.070017e-04 -0.0006322484  0.0002551939  3.040542e-05
##    3 -7.954433e-05  7.159406e-05  0.0011647085 -0.0027003549 -3.375933e-03
##    4 -1.009750e-03  1.690892e-03  0.0031638891 -0.0031196620  1.332229e-03
##    5  5.417419e-04 -1.864268e-03  0.0033322600 -0.0027910037  1.692734e-03
##
## Docs          PC41          PC42         PC43         PC44          PC45
##    1 -1.371933e-03 -3.046581e-03 -0.001515232 0.0005498540 -0.0002020233
##    2 -2.856255e-04  4.692895e-04 -0.000133128 0.0001626426 -0.0004226903
##    3  8.539484e-05  1.582437e-03 -0.003165042 0.0013978057 -0.0054191152
##    4  2.435333e-03 -7.022046e-05  0.001294508 0.0003444139 -0.0011467638
##    5 -3.754148e-03  2.732731e-03  0.002221024 0.0014935550 -0.0044710969
##
## Docs          PC46          PC47          PC48          PC49          PC50
##    1  0.0040475908 -0.0012059661 -0.0003807751 -0.0005575076  0.0032600172
##    2  0.0013587614 -0.0007511988  0.0006204165  0.0003416724 -0.0005838330
##    3  0.0020031728 -0.0006013078  0.0003213371  0.0011816776  0.0016904629
##    4 -0.0001884376 -0.0021762697 -0.0001562336 -0.0004121451 -0.0001465954
##    5  0.0020323866 -0.0003954486  0.0008316883  0.0029561859  0.0026679784
##
## Docs          PC51          PC52          PC53          PC54         PC55
##    1  0.0012315078 -1.741166e-03  0.0011192354  0.0031321946 2.582193e-03
##    2  0.0004421469 -3.766461e-05  0.0002412894 -0.0005489304 3.985796e-05
##    3 -0.0028579386 -5.889356e-03 -0.0046314925 -0.0043058980 9.829458e-04
##    4  0.0019289913 -1.542587e-04  0.0006024567  0.0005323992 2.218914e-04
##    5 -0.0021505698 -1.906041e-03 -0.0030845185  0.0024841794 3.832564e-03
##
## Docs          PC56          PC57          PC58          PC59          PC60
##    1 -0.0001121058 -1.167304e-03  0.0011055334 -6.897669e-04  0.0009285585
##    2 -0.0005857175 -1.102960e-03 -0.0006466388 -1.975404e-05  0.0002453069
##    3  0.0029769177  2.044493e-03 -0.0021980535 -1.588589e-03 -0.0023669855
##    4 -0.0006326751 -5.831317e-04 -0.0027835294  6.176752e-04 -0.0023393361
##    5 -0.0001742959 -2.433439e-05 -0.0025793787 -4.068803e-03 -0.0017840740
##
## Docs          PC61          PC62          PC63         PC64          PC65
##    1  0.0008640154  2.935515e-04  0.0002949581  0.001082507  0.0021017603
##    2  0.0002716759 -2.088295e-04 -0.0004459066 -0.000164764  0.0004751944
##    3  0.0001514541  5.417635e-03  0.0002895541  0.002320638 -0.0055612784
##    4  0.0001757713  1.323645e-04  0.0012770371 -0.002330937  0.0030577933
##    5 -0.0011491523  4.960172e-05  0.0020975851 -0.001261406 -0.0026544034
```

```
##
## Docs          PC66          PC67          PC68          PC69          PC70
##    1 -0.0036449491  0.0020948646  0.0011055378 -0.0017150107  3.170221e-04
##    2 -0.0004360646  0.0003043963 -0.0005248678 -0.0009042723  2.647202e-05
##    3  0.0011034001  0.0015955063 -0.0047478240 -0.0030235166  1.536077e-03
##    4 -0.0016053683 -0.0014355859 -0.0003200835 -0.0012914873  1.827328e-03
##    5  0.0020481518  0.0049711344 -0.0048751424 -0.0045609719 -7.060710e-05
##
## Docs          PC71          PC72          PC73          PC74          PC75
##    1  0.0005625823 -3.396892e-04  0.0017217357  0.0022219550 -0.0012503282
##    2  0.0008682962 -1.861320e-05 -0.0004866762  0.0007768000 -0.0004024475
##    3  0.0064086380  7.374245e-06 -0.0021457868  0.0045245946  0.0013953400
##    4 -0.0004394266 -7.472087e-04  0.0005471041  0.0001547876  0.0012040203
##    5 -0.0003920024  2.111308e-03  0.0022831102 -0.0021424924 -0.0009850936
##
## Docs          PC76          PC77          PC78          PC79          PC80
##    1 -1.311619e-03 -2.524630e-04 -0.0001154905  0.0016605118  2.590765e-03
##    2  5.667896e-05 -1.131543e-04 -0.0009338305  0.0011102105  1.078560e-04
##    3  6.533880e-04  3.635725e-03  0.0008273532  0.0038523291 -4.327811e-03
##    4 -1.961492e-03 -9.051506e-05  0.0001224226  0.0001699783 -2.420021e-05
##    5 -5.405877e-04  5.420128e-03  0.0019723143 -0.0005074946  2.150378e-03
```

Part d

I ran a knn model with a k of 10 to try to predict the author of a document based on the PCA values of the testing set. The end result of the knn model is a training accuracy of 66% and a testing accuracy of 40%. As we can see at the k = 10 range, the training of 66% indicates that the training documents are very "close" to each other and that there is good seperation between the different authors. An accuracy of 40% may not seem that great but the baseline accuracy of this model is 2% due to there being 50 different possible authors. I would like a higher testing accuracy, but I believe given the limitation of the original data that an accuracy of over 40% is actually somewhat impressive.

I think the knn model (and any model chosen) suffers from the curse of dimensionality. Even running PCA to try to capture around 40% of the variance, the amount of dimensions is 80 which introduces a large amount of noise into the model. I believe a lot of the noise comes from the fact that the article topics can be far ranging making it hard to isolate the word choices corresponding to an author's writing style.

I believe the model could be optimized by using for loops to cross validate for the best testing accuracy. The best cross validated model would actually use three nested for loops around the factors of k-value, the sparcity of the DTM matrix, and the dimension reduction by PCA. Unfortunately due to both computational and run time limitations (I think it would probably take at least a few hours to do this which I do not really have), I was only able to adjust the factors one at a time and play around to improve my testing accuracy.

```
##            Actual     Predicted Freq
## 1   AaronPressman AaronPressman   33
## 2      AlanCrosby AaronPressman    0
## 3  AlexanderSmith AaronPressman    0
## 4 BenjaminKangLim AaronPressman    0
## 5   BernardHickey AaronPressman    0
```

```
## [1] "The accuracy of the model is"
```

```
## [1] 0.4056
```

Problem 6

Part a

This chunk is processing the text file, factoring the baskets together, and reindexing to prepare for apriori in the next part. I do not know if there was an easier way to read in the text file, but when I read it in the text file had four columns. The column corresponded to the different items in a basket in the row in the text file. I had to resort the data frame such that a apriori could be run on it.

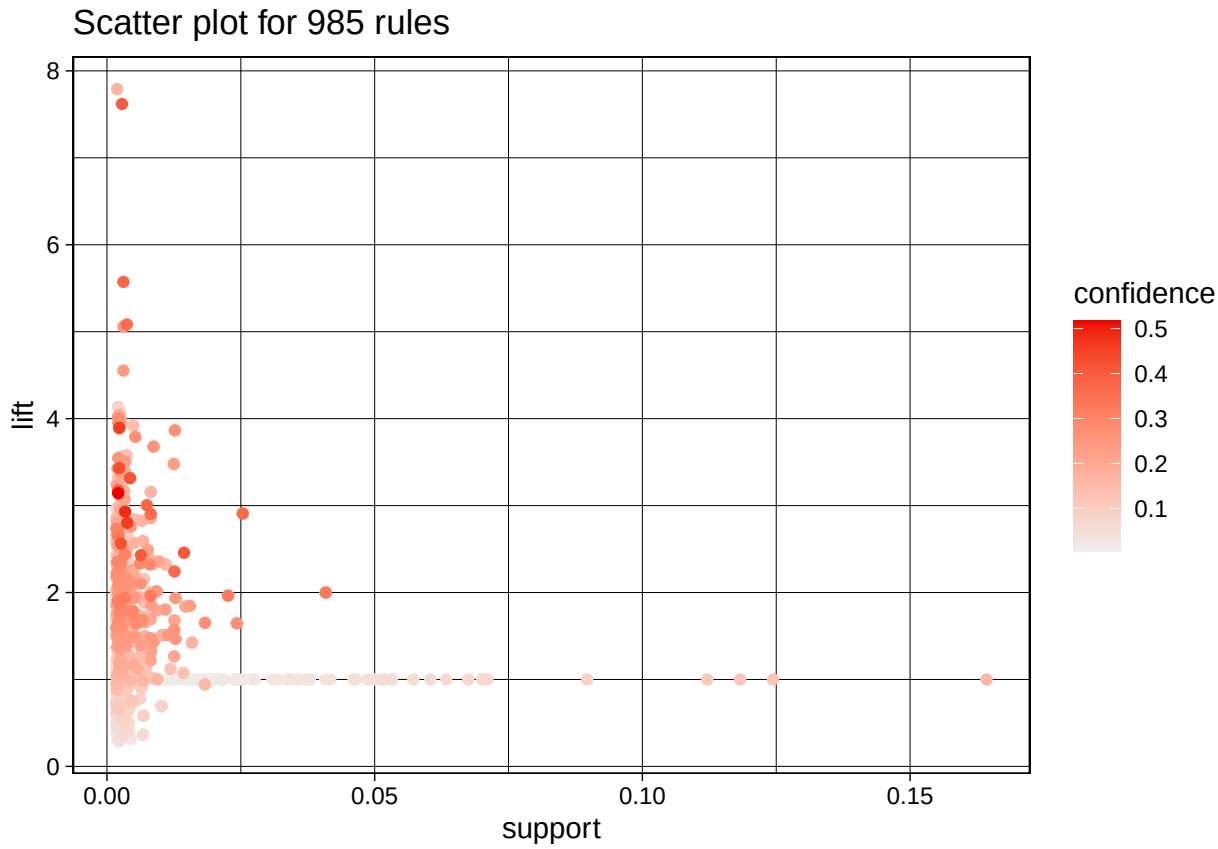
```
##
## Attaching package: 'arules'

## The following object is masked from 'package:tm':
##
##     inspect

## The following objects are masked from 'package:mosaic':
##
##     inspect, lhs, rhs

## The following object is masked from 'package:dplyr':
##
##     recode

## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

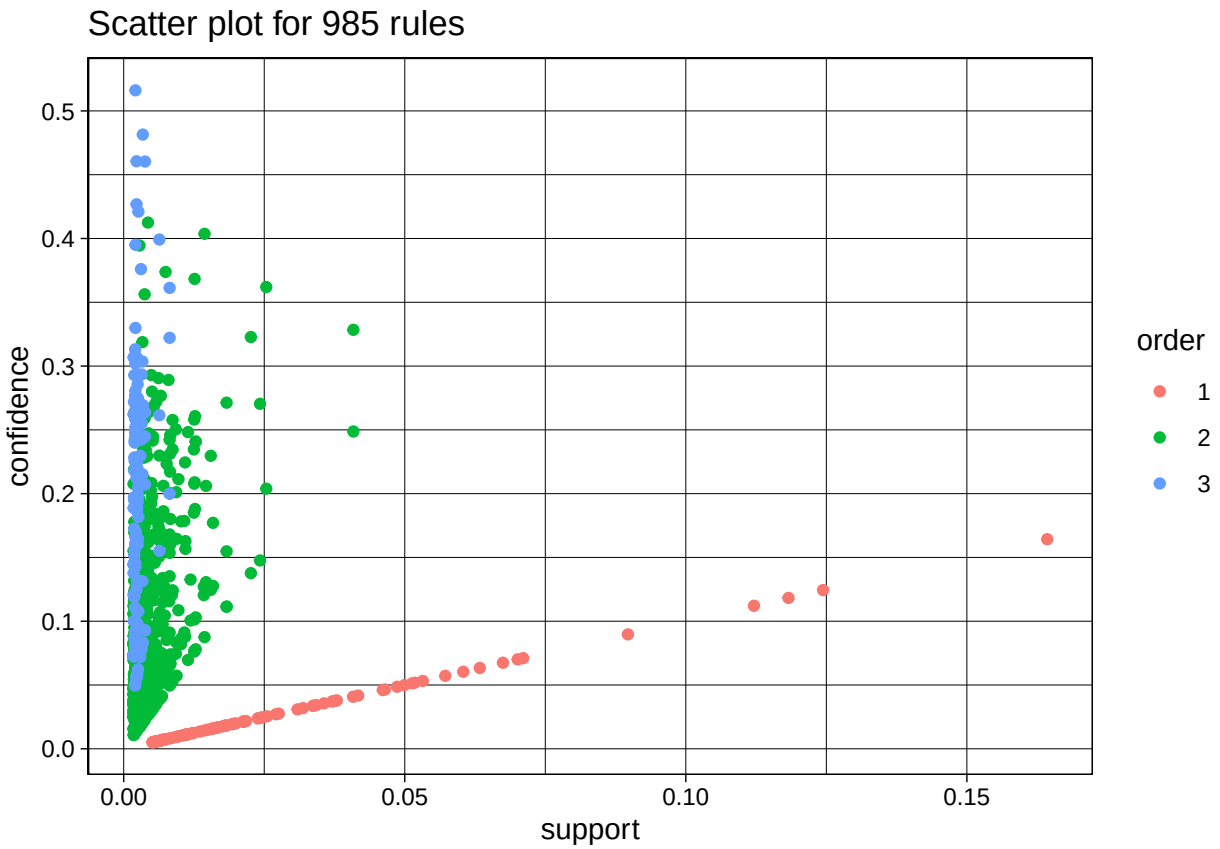
Part b

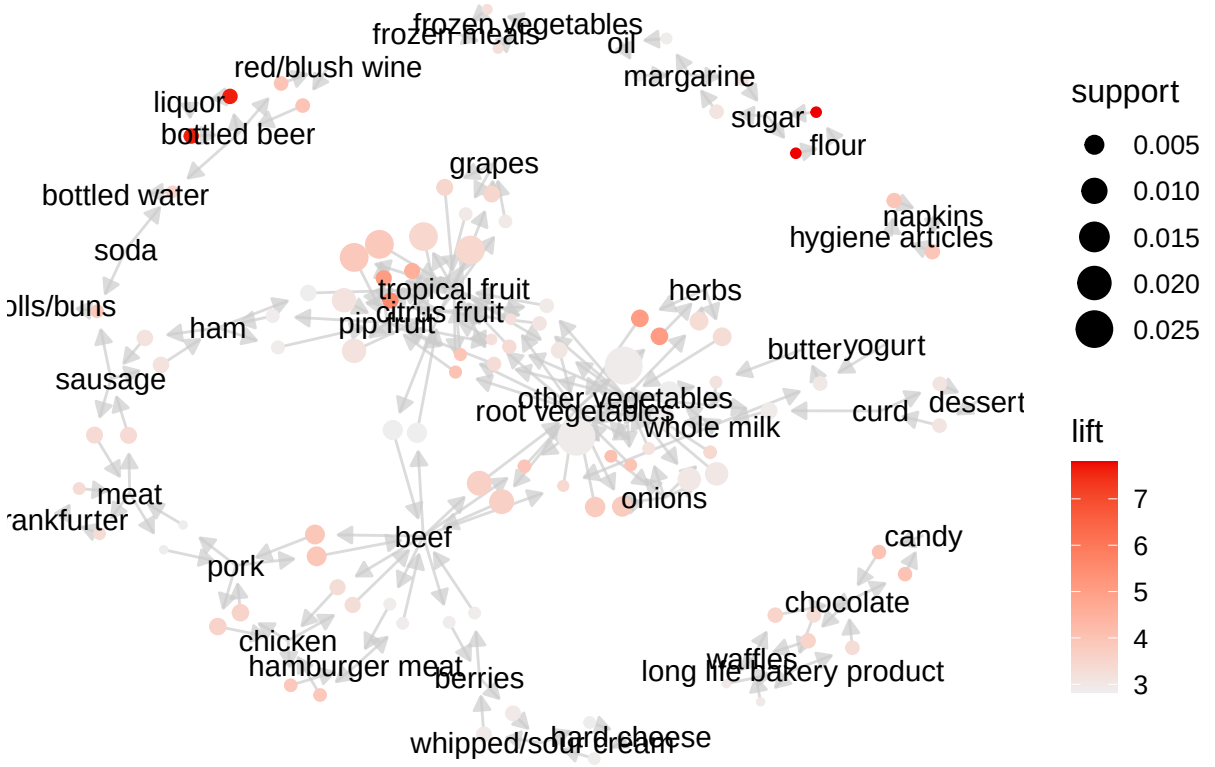
The important part of the association rule mining is the support and confidence parameters chosen. So the unique number groceries is 169 and the number of baskets is 15,296 which means there should be lots of combinations of baskets. However for my apriori, I will try to choose a support and confidence that gives me around 1000 association rules. For support, each basket has an average 2.83 items out of a possible 169 or a 1.7% chance of an item being in a basket on average. I am going to take a support value of 1/10 of this at 0.0017. Setting my support value to a set number, I varied my confidence until I had about 1000 association rules. Finally, I take half the rules into my gephi plot sorted by the highest values of lift.

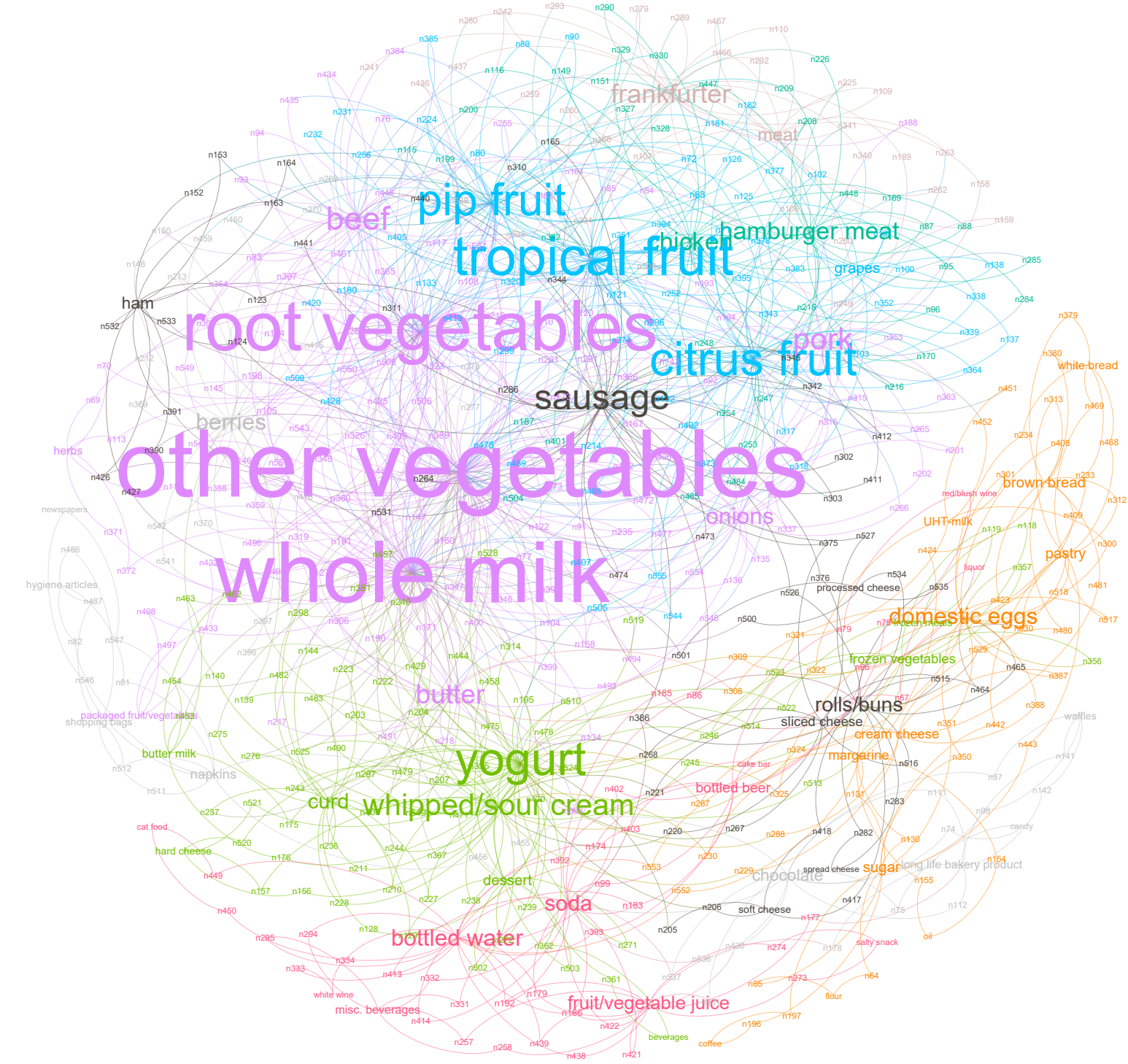
```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##       0.005    0.1    1 none FALSE            TRUE       5  0.0017      1
##  maxlen target  ext
##       4  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 26
```

```
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 15296 transaction(s)] done [0.00s].
## sorting and recoding items ... [140 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [985 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```




Scatter plot for 985 rules
confidence
0.0
0.1
0.2
0.3
0.4
0.5
0.00
0.05
0.10
0.15
support
order
1
2
3


frozen vegetables
frozen meals
oil
red/blush wine
margarine
liquor
sugar
bottled beer
flour
grapes
bottled water
napkins
hygiene articles
soda
tropical fruit
herbs
olls/buns
citrus fruit
ham
pip fruit
butter
yogurt
sausage
other vegetables
curd
dessert
root vegetables
whole milk
meat
onions
ankfurter
beef
candy
pork
chocolate
chicken
waffles
hamburger meat
long life bakery product
berries
whipped/sour cream
hard cheese
support
0.005
0.010
0.015
0.020
0.025
lift
7
6
5
4
3


whole milk
other vegetables
root vegetables
tropical fruit
yogurt
sausage
citrus fruit
pip fruit
beef
chicken
hamburger meat
frankfurter
meat
pork
grapes
ham
berries
herbs
onions
butter
curd
whipped/sour cream
soda
bottled water
fruit/vegetable juice
bottled beer
rolls/buns
sliced cheese
cream cheese
margarine
domestic eggs
frozen vegetables
processed cheese
pastry
brown bread
white bread
UHT-milk
dessert
chocolate
sugar
napkins
butter milk
hard cheese
misc. beverages
white wine
beverages
coffee
flour
oil
waffles
candy
soft cheese
spread cheese
long life bakery product
salty snack
cat food
packaged fruit/vegetables
shopping bags
hygiene articles
newspapers
red/blush wine
frozen meals
cake bar