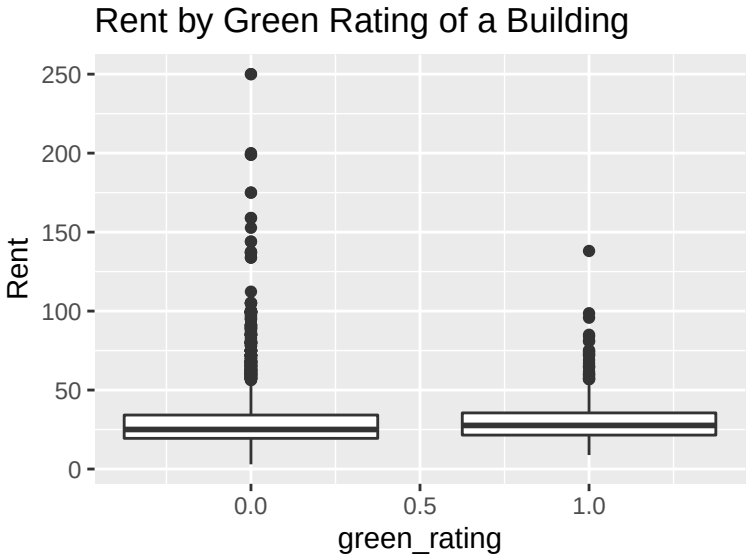
STA S380 Part 2

Anqi Lou (al44684)

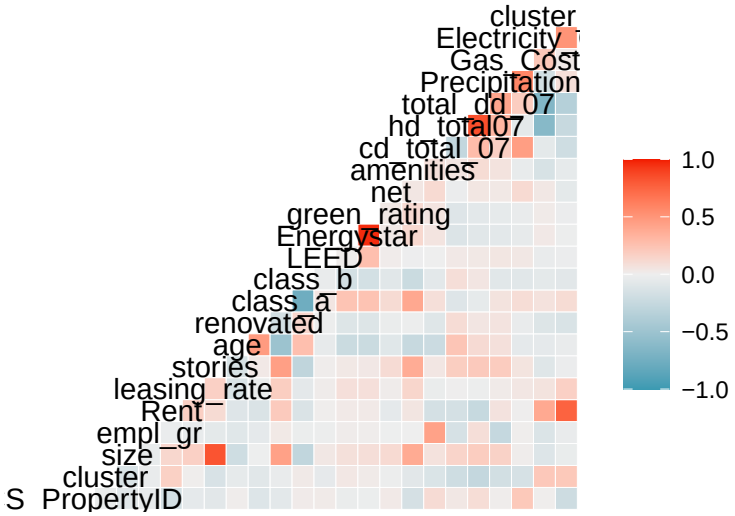
08/17/2020

Visual story telling part 1: green buildings



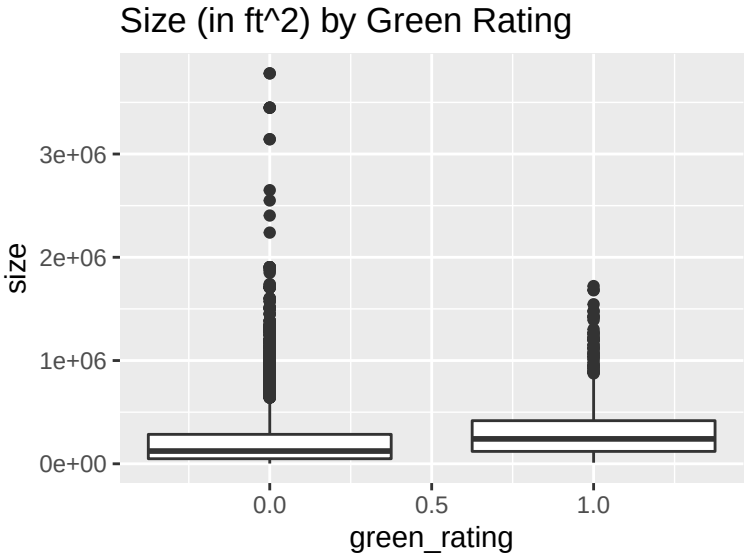

```
## [[1]]
##   ymin   lower middle upper  ymax
## 1 2.98 19.4300  25.03 34.18 56.27
## 2 8.87 21.4975  27.60 35.54 55.94
##
## 1 80.00, 85.00, 99.42, 80.00, 68.21, 74.76, 91.04, 57.66, 61.37, 60.00, 91.04, 61.00, 75.00, 60.00, 
## 2
##   notchupper notchlower x flipped_aes group PANEL ymin_final ymax_final   xmin
## 1   25.30865   24.75135 0       FALSE     1     1       2.98     250.00 -0.375
## 2   28.44835   26.75165 1       FALSE     2     1       8.87     138.07  0.625
##    xmax xid newx new_width weight colour  fill size alpha shape linetype
## 1 0.375   1    0      0.75      1 grey20 white  0.5    NA    19    solid
## 2 1.375   2    1      0.75      1 grey20 white  0.5    NA    19    solid
```

Based on the box plot as well as the quantile values for the rent of green buildings and non-green buildings respectively, it is true that the rent for green buildings is generally higher than that of non-green buildings. However, we cannot say for sure that such a difference in rent is caused by the environmental friendliness of the building. To check whether there are confounding variables, we first create a correlation matrix to find out the top variables that are closely correlated with rent.



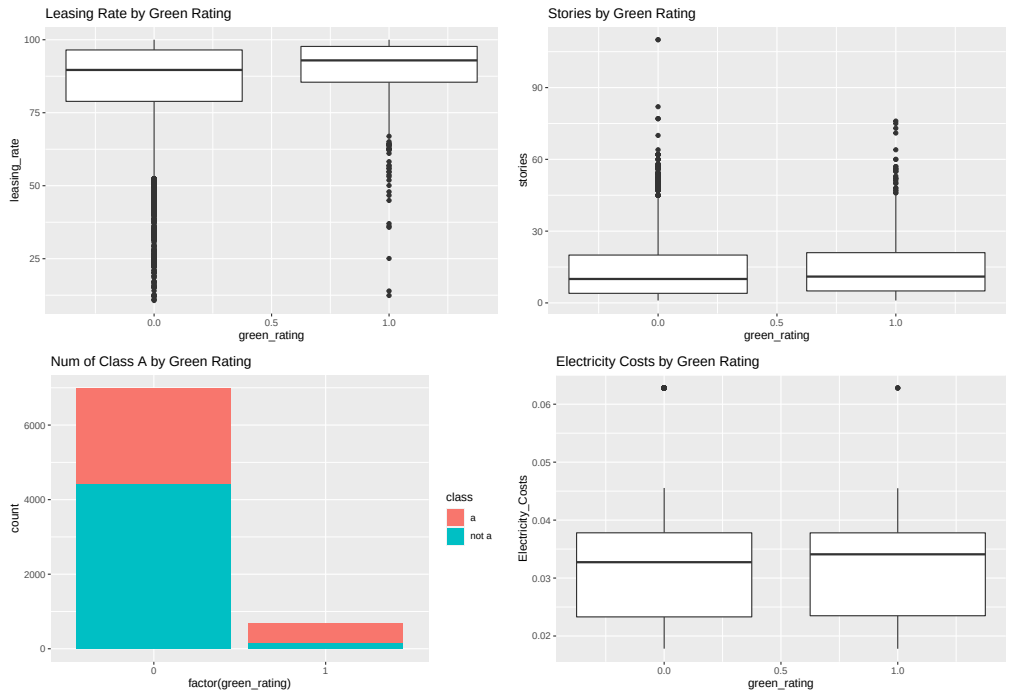
From the graph, we can tell that other than the green rating of a building, the size, leasing rate, stories, quality (whether it is classified as class a), electricity cost, and cluster rent of the building all show a greater positive relationship with the rent for that building. While it is obvious how the average rent in the buildings local market can have an influence on the rent, we need to study further into the rest of the variables to find out whether they also have some correlations with the green rating of a building. If so, it could confound the relationship that the stats guru points out and would make it arbitrary to say that green buildings generally have a higher rent.

In order to do so, we can plot some pairwise graphs of the above variables with the green rating of a building.

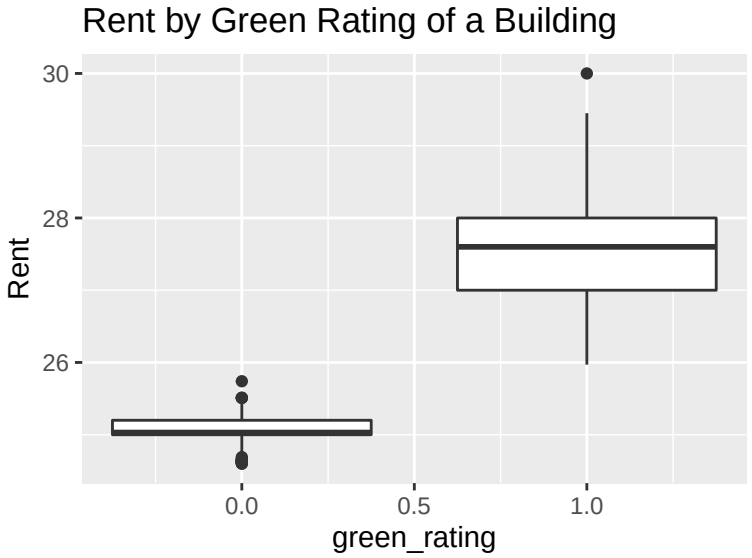


As we can see from the box plot, the green buildings generally have more available rental space than non-green ones. Therefore, it can be an implication that size is a confounding variable. It could be possible that the increase in rent for green house is not due to the fact that the building is environmentally conscious, but because of the coincident that these buildings happen to be larger and have more rental space.

For rest of the variable, the box plot/bar chart by the green rating of a building is as below:



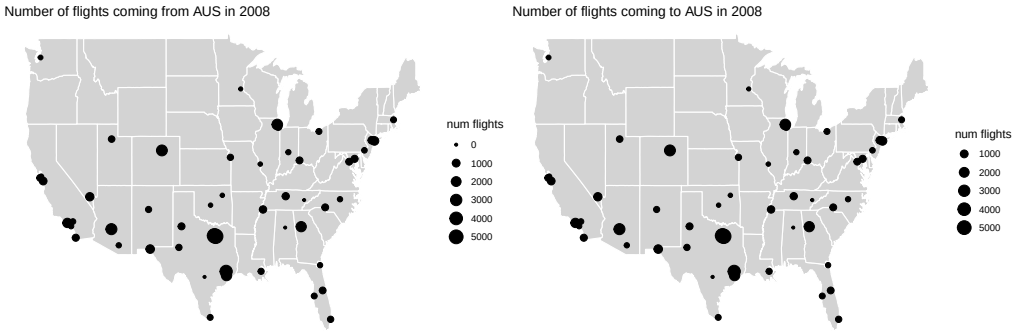
Similar to size, all the other four variables—— leasing rate (occupancy rate), stories, whether it is classified as class a, and electricity cost, all show some particular patterns with the green rating of a building. To be more specific, a green building is more likely to have a higher leasing rate, greater number of stories, better quality, and greater electricity cost. Therefore, these could all be confounding variables; it might not be green buildings that are having a higher rent, it might in fact, be those buildings that are having a higher occupancy rate, more stories, better qualities, and higher electricity rate that are having a higher rent. Therefore, in order to adjust for the impact of these potential confounding variables, we can divide our current data set into two, one being all green buildings, and the other being non-green buildings. Then we apply bootstrap to re-sample on the two data set separately 2500 times to get a better estimate of the population median of the two groups. In that way, we are less likely to be impacted by those confounding variables.



```
## [[1]]
##    ymin lower middle upper  ymax
## 1 24.71    25  25.03  25.2 25.50
## 2 25.97    27  27.60  28.0 29.45
##
## 1 25.51, 24.66, 24.60, 24.64, 24.64, 24.64, 24.62, 25.51, 24.62, 24.60, 24.64, 25.51, 25.74, 24.69, 2
## 2
##   notchupper notchlower x flipped_aes group PANEL ymin_final ymax_final   xmin
## 1   25.03632   25.02368 0       FALSE     1     1      24.60      25.74 -0.375
## 2   27.63160   27.56840 1       FALSE     2     1      25.97      30.00  0.625
##    xmax xid newx new_width weight colour  fill size alpha shape linetype
## 1 0.375   1    0      0.75      1 grey20 white  0.5    NA    19    solid
## 2 1.375   2    1      0.75      1 grey20 white  0.5    NA    19    solid
```

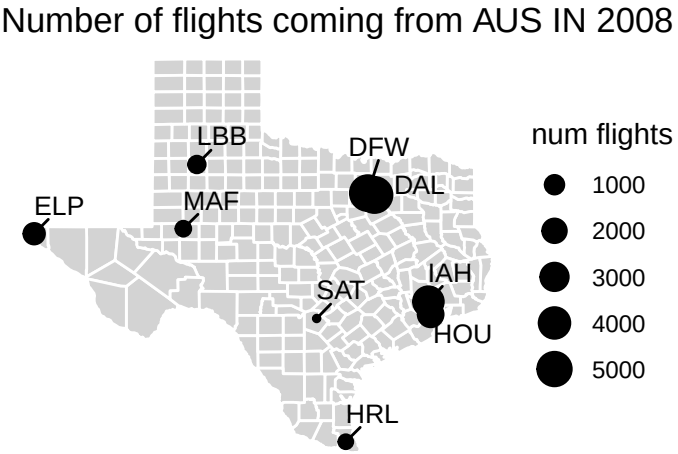
As it turns out, the median rent of a green building estimated using bootstrap is $27.6, while the median rent of a non_building is $25.03. This result is very similar to the one from the stats guru. Therefore, even thought there might be impact from confounding variables, we can say with more confident that the rent of a green building would be higher than that of a non-green one, and that it is "a good financial move to build the green building."

Visual story telling part 2: flights at ABIA

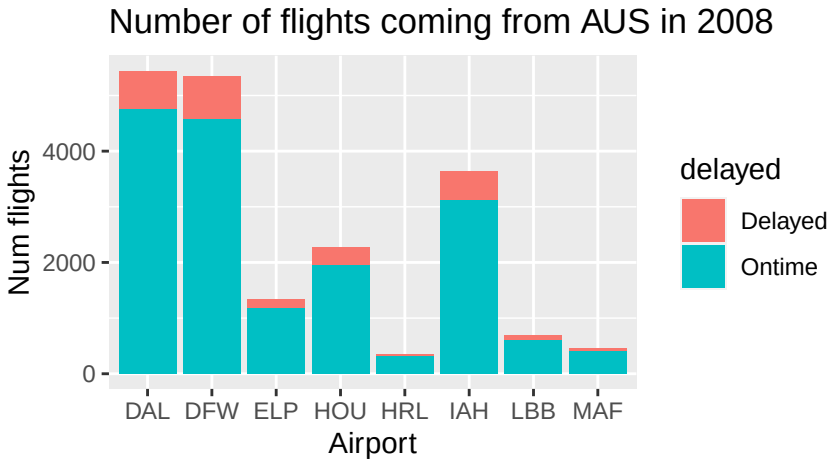


As it can be seen from the above two maps, the number of planes flying to Austin from airports over the nation is roughly the same as the number of planes flying out of Austin to each airport. This is not hard to explain, as planes usually fly in round trips, back and forth between two cities.

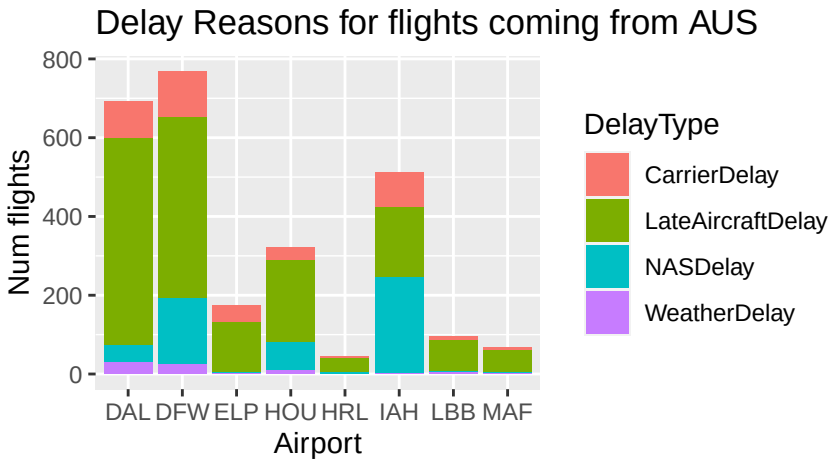
Now, let's focus on the flights/airports that are in Texas only.

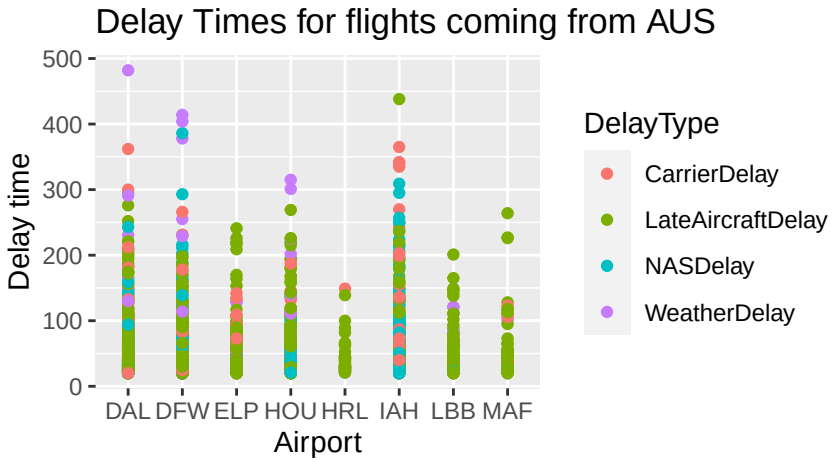


To see how many of them are delayed: (Classify a plane as delayed if ArrDelay is greater than 20min)



Focus on those flights that are delayed and see the reasons for the delay: (if more than two types of delay occurred, categorize it as the type that has the longest delay in time)





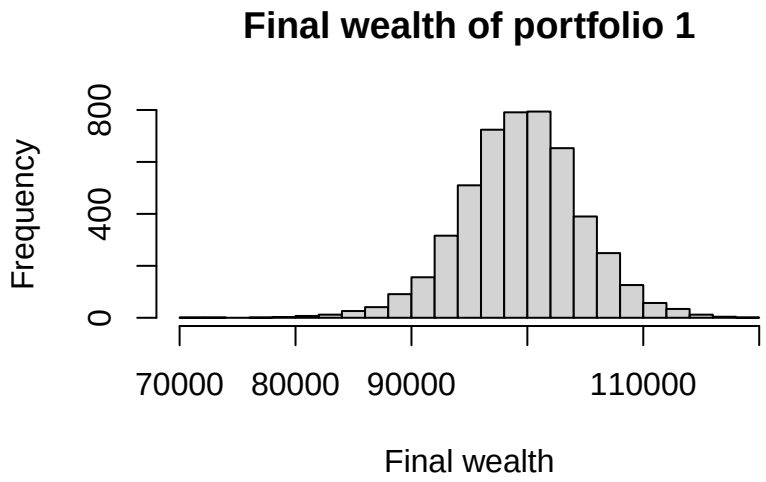
Portfolio modeling

We built three portfolios of ETFs. Each portfolios would consist of 5 equally-weighted different ETFs. The first would focus on commodity ETFs; the second would be on bonds, which can be considered as a relatively safe one; the third would focus on small & mid cap equities and can be considered as a riskier one.

Portfolio 1: CORN, WEAT, BNO, GSC, USCI
Portfolio 2: VCLT, QLTA, GOVT, TLH, SPIB
Portfolio 3: SCHC, VBR, DES, EEMS, PRFZ

Use bootstrap to simulate the performance for 20 trading days:

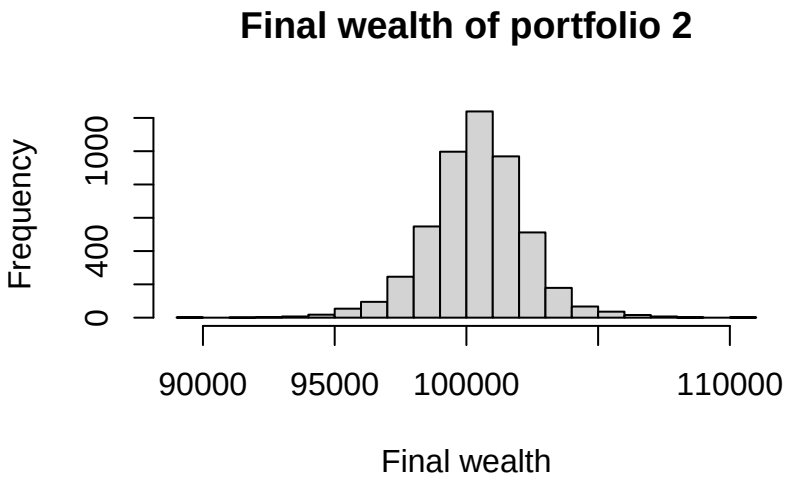
Portfolio 1



The value at risk at the 5% level is:

```
##       5%
## -9035.46
```

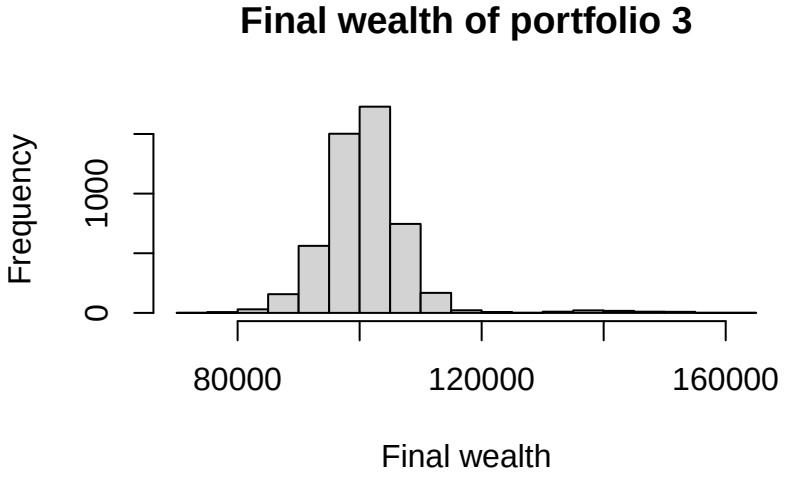
Portfolio 2



The value at risk at the 5% level is:

```
##        5%
## -2579.618
```

Portfolio 3



The value at risk at the 5% level is:

```
##        5%
## -9011.815
```

In terms of riskiness and volatility, portfolio 3 is the most aggressive one among the three, as it focuses on small to median cap securities. This characteristic can also be seen from the VaR AT 5%. Portfolio 3 has a VaR of 9602.088, which is the highest among the three. Even though portfolio 1 has a VaR that is very close to that of portfolio 3, if we take a look at the bar charts, the distribution of the final wealth of portfolio 3 is way more spread out than that of portfolio 1, which implies that portfolio 3 is a more risky one. On the other hand, portfolio 2 is the safest one. Not only is its VaR much smaller than that of the other two portfolios, its simulated final return is also more clustered around $100,000.

Market segmentation

First, we need to clean the data and do some pre-processing.

For data cleaning, we removed all the rows with missing entries. And for data pre-processing, there are mainly three things to do. One, remove the row if "spam" and "adult" tweets take up more than one third of the total tweets that the user has posted. Since these posts are indeed about inappropriate contents, posting too many of such tweets might imply the illegitimacy of the user. Thus, they definitely would not be the potential consumer for NutrientH20, and we should exclude them in the analysis. The second pre-processing job to be done involves removing the row if the user has posted less than five tweets. Posting too little contents means that the user is not very active online, which gives us little information to learn about their interests. Therefore, we would also like to keep them out of our study so that they are less likely to skew the result. (Actually, it turned out that almost all users have posted more than 5 tweets.) And lastly, we want to normalize tweet counts to tweet frequencies.

After cleaning the data, in order to learn about the market segmentation, we can use k-means++ on the cleaned data set to cluster the users and find their correlated interest. However, since there are too many variables, we first apply PCA to reduce the dimension.

Below is a PCA summary that could help us decide the number of variables to reduce to:

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     1.7023 1.62057 1.54477 1.46185 1.41119 1.27760 1.20410
## Proportion of Variance 0.0805 0.07295 0.06629 0.05936 0.05532 0.04534 0.04027
## Cumulative Proportion  0.0805 0.15345 0.21974 0.27910 0.33442 0.37976 0.42003
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.11417 1.08810 1.05459 1.02970 0.99936 0.98741 0.97874
## Proportion of Variance 0.03448 0.03289 0.03089 0.02945 0.02774 0.02708 0.02661
## Cumulative Proportion  0.45451 0.48740 0.51830 0.54775 0.57549 0.60257 0.62918
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.97161 0.95553 0.94452 0.92403 0.91448 0.88424 0.85313
## Proportion of Variance 0.02622 0.02536 0.02478 0.02372 0.02323 0.02172 0.02022
## Cumulative Proportion  0.65540 0.68077 0.70555 0.72927 0.75249 0.77421 0.79443
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.82648 0.82149 0.80329 0.78857 0.77243 0.76803 0.75934
## Proportion of Variance 0.01897 0.01875 0.01792 0.01727 0.01657 0.01639 0.01602
## Cumulative Proportion  0.81341 0.83215 0.85008 0.86735 0.88392 0.90031 0.91632
##                           PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     0.74865 0.73383 0.69897 0.64591 0.61913 0.56583 0.55145
## Proportion of Variance 0.01557 0.01496 0.01357 0.01159 0.01065 0.00889 0.00845
## Cumulative Proportion  0.93189 0.94685 0.96042 0.97201 0.98266 0.99155 1.00000
##                            PC36
## Standard deviation     3.563e-15
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

Based on the summary of importance of components, we can see that reducing the number of variables to a very small number might drastically sacrifice the variance among the observations. Therefore, we choose to reduce the dimension to 10 so that at least half of original variance can be maintained but also at the same time achieves the purpose of simplifying the data set.

Next, we apply K-means++ to cluster the observations into 5 groups to learn about the market segment. Below is the landings for the center of the five clusters on the ten principal components:

```
##          PC2         PC3         PC4        PC5         PC6         PC7
## 1 -0.6387307 -0.01140249  3.46065189 -0.1356922 -1.03445356  0.18152030
## 2 -0.7453529  0.69234183 -0.27842223  0.7040012  0.33496043  0.02266727
## 3  2.5286060 -1.10729690 -0.08331891  0.8810241  0.06008130  0.04412036
## 4 -1.2343294 -2.15528356 -0.80408043 -1.2497915 -0.32888922 -0.09116312
## 5  1.2200913  2.01445285 -0.55293588 -2.3187219 -0.09357201 -0.18460881
##           PC8         PC9          PC10
## 1 -0.01718250 -0.10121897  0.0004918195
## 2 -0.08877603  0.07831598  0.0231484726
## 3  0.12642810  0.02508373 -0.0060473716
## 4  0.07140823  0.02754418 -0.0341699214
## 5  0.06708414 -0.30265169 -0.0348943100
```

By looking at the extreme values in each cluster, we can see that cluster 1 has higher scores for PC6 and PC9; cluster 2 has the highest score for PC2; cluster 3 is somewhat average on the other PCs, but has a very low value for PC3; cluster 4 is most characterized by PC4; and cluster 5 is most characterized by PC3, the opposite of cluster 3. In order to get a better understanding of those principal components, we can look at the top five variables that load most heavily on those PCs.

Start from PC6 and PC9 for cluster 1:

```
##   Tweet_category       PC6
## 1        tv_film 0.4837416
## 2            art 0.4544130
## 3  uncategorized 0.2294394
## 4         crafts 0.2283398
## 5          music 0.1639799
```

```
##    Tweet_category       PC9
## 1          dating 0.6919188
## 2          school 0.3075298
## 3         chatter 0.1235630
## 4 home_and_garden 0.1161422
## 5             art 0.1129428
```

We can see that PC6 is more are the artistic side while PC9 is more about gossips and chores.

Then, the top variables that load on PC2 which most characterizes cluster 2:

```
##     Tweet_category       PC2
## 1 health_nutrition 0.4731415
## 2 personal_fitness 0.4451149
## 3         outdoors 0.3379407
## 4          cooking 0.3024274
## 5          fashion 0.1574466
```

Apparently, PC2 is all about health and personal fitness Thus, it is not hard to tell that cluster 2 is a group of users who care a lot about their body and the overall wellness. This would be a good potential market segment for NutrientH20. (Judging from the name, assume NutrientH20 is a company that sells health and nutrition related products, just like GNC.)

Now, look at PC3 which cluster 5 has a very high value for, but cluster 3 has an extremely low value for:

```
##   Tweet_category       PC3
## 1        fashion 0.2996091
## 2         beauty 0.2950506
## 3  photo_sharing 0.2637906
## 4        cooking 0.2192780
## 5         school 0.1892471
```

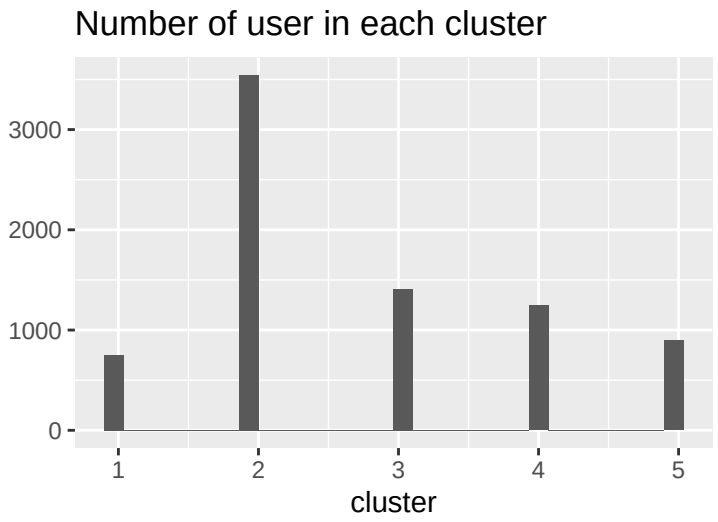
This principal component is all about the fashion/beauty area. Consequently, it implies that cluster 5 might consist of a group of young ladies who loves makeups and clothing, while cluster 3 is the exact opposite.

Lastly, the top variable captured by PC4, characterizing cluster 4:

```
##   Tweet_category       PC4
## 1    college_uni 0.5544121
## 2  online_gaming 0.5187851
## 3 sports_playing 0.3453087
## 4        tv_film 0.1980418
## 5            art 0.1483202
```

As we can tell from above, PC4 is more about college lives. Thus, we may imagine that cluster 4 might be a group of college kids who are really into sports and computer games.

In addition, we can also look at the number of people in each cluster to help us get a better idea of each market segment.



As we can tell from the above graph, while the second cluster might be those who are most likely to purchase from NutrientH20, they actually have the lowest number of users among all five clusters. On the other hand, more than half of all users fall into the first cluster, which talks more about movies, and music, and some random stuff. It implies that a lot of people following NutrientH20 online have no strong interest in company's product. The company should try to identify the users who fall into cluster 2, as they are more likely to be the potential customers.

Author attribution

Part One - Data Pre-processing

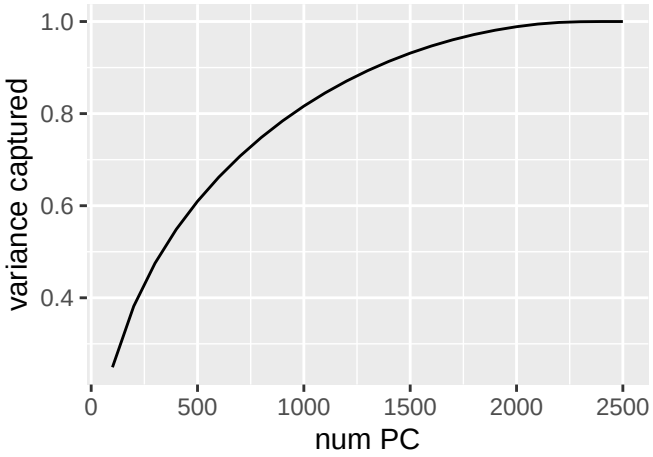
In order to create the data set we need for building and testing the model, we first read in all the files in the training folder and test folder. Each time we read in a folder for an author, we create a corpus for that author and apply a series of transformation to make the data more processable. This involves tokenizing the articles, making all the words lowercase, removing all numbers, punctuation, and white spaces, as well as removing stopwords from the "SMART" stopword list. In addition, words with more than 95% sparse entries all files from the same author will also be dropped. Too many parse entries mean that the word is only observed in very few files. Removing them can prevent us from learning from the "noise" in the long tail. After that, TF-IDF score will be calculated for all processed corpora. The scores for corpora from the training folder will be combined into a matrix and those from the test folder will be combined into another matrix.

Next, we need to adjust for the words that appeared in the testing matrix but not in the training one. In order to do so, we first create a pseudo word "pseuddo" in the training set with values all being zeros. Then, we bind the rows of the training set with the testing set, and number of non-zero entries in the columns after "pseuddo", which are the columns for words that are never seen in the training set, will be counted and recorded as the new value for "pseuddo". In that way, the values for "pseuddo" for all the files from the testing set will be the number of words that did not appear in the training set, while the value for pseuddo" for all files from the training set will still be zeros. After that, columns after "pseuddo" will be dropped.

As for the final step for pre-processing the data, we need to populate the response variables, in other words, add a categorical variable "author" to indicate the author of the article. After splitting the data back into training and testing, we again process the two sets of data in groups of fifty. For instance, for the first fifty instances in the training data set, the value for "author" will be the name of the first folder in the training folder, and the second fifty instances will have the value corresponding to the name of the second folder in the training folder, so on and so forth.

Part Two - Building Models

Now, there are more than 9,000 variables in both training and testing set, which is almost impossible to be directly used to build and test the model. Therefore, we first carry out a principal components analysis to reduce the dimension of our dataset. Below is a graph of vairance of the training set being captured versus the number of dimension:

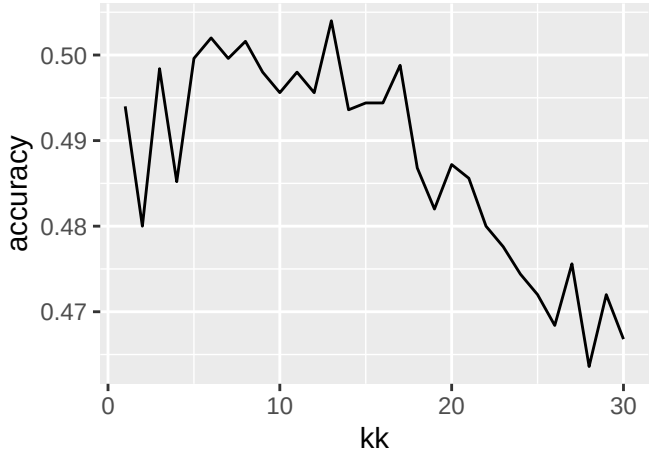


As can be seen, even reducing the number of variables to 400, a still relatively large number, can only preserve 50% of the variability of the data. Therefore, for the purpose of this exercise, we choose to reduce the dimension to 150. While much variance might be lost, it is a less costly model and would be a more

economic choice time-wise. One thing to note here is that, instead of projecting both training and testing set to a 150-dimensional PCA space at the same time, we need to project the training set first, and we will use the PCA space that the training set created to project the testing set.

In order to find a better-performing model, we will try out two methods to build the model: K-nearest neighbors, Random Forest. For KNN, different values of K will be tested, and for random forest, we will also try out different number of trees.

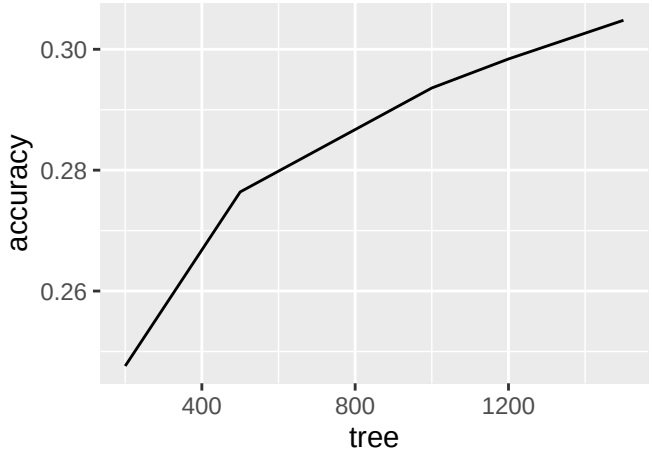
KNN with K = 1 to 30



Based on the above graph, we can tell that the optimal k is 6, and it gives an accuracy rate of:

```
## [1] 0.504
```

Random Forest with num trees = 200, 500, 1000, 1200, 1500



The optimal number of trees for Random Forest is 1200, but it can only give us an accuracy rate of

```
## [1] 0.3048
```

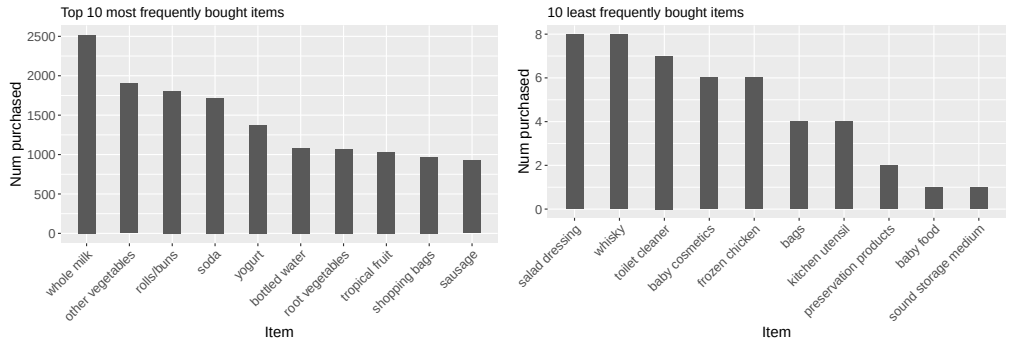
Part Three - Conclusion

Based on the above analysis, we can see that the KNN is performing, on average, much better than the Random Forest model. The best model we have tested is KNN with K = 6. However, we can only achieve roughly 50% accuracy rate with the model. Possible reasons for this rather low number might be that reducing the dimensions of the original data set to a small number discards many of the variances. Had it not been the constraint in time, we could have tested out different PCAs, and it is possible that the KNN could perform better if we feed it with a higher-dimensional data.

Association rule mining

After reading in the data, we first do some exploratory analysis.

The groceries.txt file recorded 9835 transactions and 169 unique grocery items. The top and bottom 10 most frequent items that people put in their shopping list are:



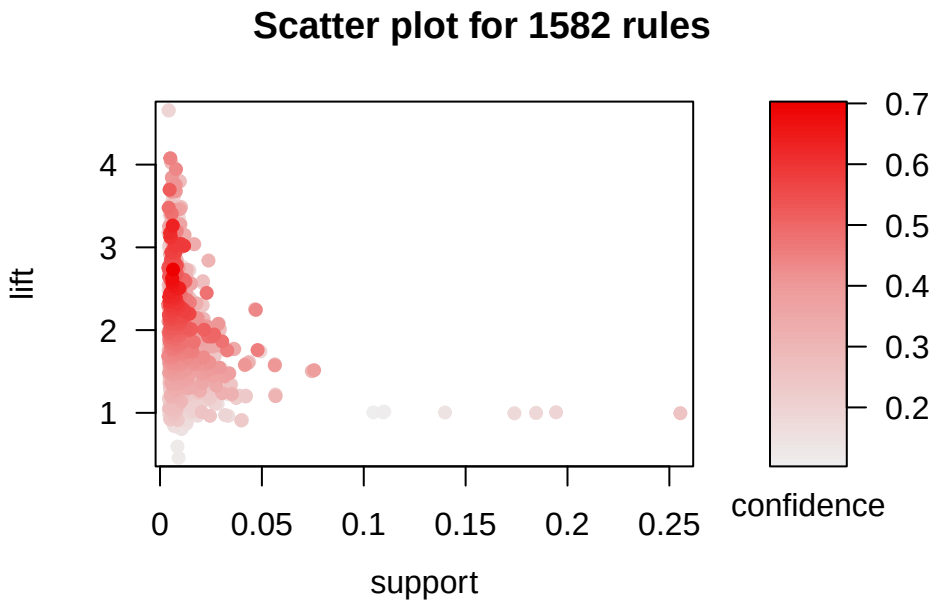
Based on the summary below, on average, people buy 3 or 4 items at a time. There is one case where a customer picked up 23 items. For the purpose of applying apriori algorithm to find the pattern, we will set the max length of a rule to be 3, as this is the median number of items that people will buy.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.636   5.000  23.000
```

In addition, we will set the minimum support of a rule to be 0.001, as salad dressing, the tenth least likely bought item, has roughly that level of support. And the minimum confidence for a rule will be set to 0.1.

Below is a scatter plot for the 1582 rules we found with apriori algorithm.

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5   0.005      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 49
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.03s].
## sorting and recoding items ... [120 item(s)] done [0.01s].
## creating transaction tree ... done [0.11s].
## checking subsets of size 1 2 3 4 done [0.02s].
## writing ... [1582 rule(s)] done [0.36s].
## creating S4 object  ... done [0.03s].
```



As we can tell from the graph, higher level of lifts are usually associated with lower level of support. To further inspect the rules we got, we will look at the result from two aspects, one sorted based on lift, and the other sorted based on confidence.

First, we sort the rules by lift and below is the top 10 rules that have the highest lifts:

```
##      lhs                     rhs                     support confidence   coverage     lift count
## [1]  {ham}                => {white bread}        0.005083884  0.1953125 0.02602949 4.639851    50
## [2]  {white bread}        => {ham}                0.005083884  0.1207729 0.04209456 4.639851    50
## [3]  {citrus fruit,
##       other vegetables,
##       whole milk}         => {root vegetables}    0.005795628  0.4453125 0.01301474 4.085493    57
## [4]  {butter,
##       other vegetables}   => {whipped/sour cream} 0.005795628  0.2893401 0.02003050 4.036397    57
## [5]  {herbs}              => {root vegetables}    0.007015760  0.4312500 0.01626843 3.956477    69
## [6]  {other vegetables,
##       root vegetables}    => {onions}             0.005693950  0.1201717 0.04738180 3.875044    56
## [7]  {citrus fruit,
##       pip fruit}          => {tropical fruit}     0.005592272  0.4044118 0.01382816 3.854060    55
## [8]  {berries}            => {whipped/sour cream} 0.009049314  0.2721713 0.03324860 3.796886    89
## [9]  {whipped/sour cream} => {berries}            0.009049314  0.1262411 0.07168277 3.796886    89
## [10] {other vegetables,
##       tropical fruit,
##       whole milk}         => {root vegetables}    0.007015760  0.4107143 0.01708185 3.768074    69
```

The results are very easy to interpret and make much sense. All the item sets are common combinations of foods that people will eat together. For instance, the top two rules are “ham” -> “white bread” and “white bread” -> “ham”, both with a lift of 4.639851. These are the most common ingredients for making a sandwich, which, of course, means that buying one will likely result in buying the other. The third rule has a length of three and a confidence of as high as 0.445. While the items in this set might not usually be eaten together, they are basic common foods that many people will choose to buy. Thus, it is also very easy

to understand why an association rule is made among those items. And just as mentioned above, these rules with large lifts all have relatively small support.

Next, we will look into the set of rules sorted based on confidence. Below are the top 10 rules with the highest confidence:

```
##      lhs                     rhs                    support confidence    coverage     lift count
## [1]  {root vegetables,
##       tropical fruit,
##       yogurt}             => {whole milk}       0.005693950  0.7000000 0.008134215 2.739554    56
## [2]  {other vegetables,
##       pip fruit,
##       root vegetables}    => {whole milk}       0.005490595  0.6750000 0.008134215 2.641713    54
## [3]  {butter,
##       whipped/sour cream} => {whole milk}       0.006710727  0.6600000 0.010167768 2.583008    66
## [4]  {pip fruit,
##       whipped/sour cream} => {whole milk}       0.005998983  0.6483516 0.009252669 2.537421    59
## [5]  {butter,
##       yogurt}             => {whole milk}       0.009354347  0.6388889 0.014641586 2.500387    92
## [6]  {butter,
##       root vegetables}    => {whole milk}       0.008235892  0.6377953 0.012913066 2.496107    81
## [7]  {curd,
##       tropical fruit}     => {whole milk}       0.006507372  0.6336634 0.010269446 2.479936    64
## [8]  {citrus fruit,
##       root vegetables,
##       whole milk}         => {other vegetables} 0.005795628  0.6333333 0.009150991 3.273165    57
## [9]  {other vegetables,
##       pip fruit,
##       yogurt}             => {whole milk}       0.005083884  0.6250000 0.008134215 2.446031    50
## [10] {domestic eggs,
##       pip fruit}          => {whole milk}       0.005388917  0.6235294 0.008642603 2.440275    53
```

Interestingly, 9 out of the top 10 rules with highest confidence has "whole milk" on the rhs. This pattern is not hard to understand either. Our previous analysis shows that "whole milk" is the most commonly bought item; more than one fourth of the transactions involve "whole milk". Therefore, it can be an implication that "whole milk" is a common combination with a lot many other items. No matter what people buy, they are very likely to grab a bottle of milk to put in their basket too. The only rule that does not have "whole milk" on the rhs is rule number 8. Its rhs is "other vegetables", which is the second most frequently bought item in our list of transactions.