Mason, Clay HW 1 - Try 2

STA 380, Part 2: Exercises 1

Part A.

Random_choser =.3 truthful_choser= .7 Random_choser_no = .5 Random_choser_yes = .5

Total_yes = .65 Total_no = .35

truthful_choser_yes = (Total_yes - (Random_choser_yes*Random_choser))/ (truthful_choser)

'(.5 x .3)+(.7 x truthful_choser_yes)=.65")'

truthful choosers who answered 'yes' represent 5/(5+2) or 71% of the Truthful population

```
rm(list=ls())
Random_choser =.3
truthful_choser= .7
Random_choser_no = .5
Random_choser_yes = .5

Total_yes = .65
Total_no = .35

truthful_choser_yes = (Total_yes - (Random_choser_yes*Random_choser))/ (truthful_choser)
truthful_choser_yes
```

```
## [1] 0.7142857
```

```
truthful_choser_no = 1 - truthful_choser_yes
truthful_choser_no
```

```
## [1] 0.2857143
```

```
print("Overall probabability table is shown below")
```

```
## [1] "Overall probabability table is shown below"
```

```
prob_problem = matrix(c(15,50,15,20), ncol=2)
colnames(prob_problem) <- c('Yes', 'No')
rownames(prob_problem) <- c('Random', 'Truthful')
prob_problem.table <- as.table(prob_problem)
prob_problem.table
```

```
##          Yes No
## Random    15 15
## Truthful  50 20
```

```
print("(.5*.3)+(.7*truthful_choser_yes)=.65")
```

```
## [1] "(.5*.3)+(.7*truthful_choser_yes)=.65"
```

```
print(truthful_choser_yes)
```

```
## [1] 0.7142857
```

```
print("truthful choosers who answered 'yes' represent 5/(5+2) or 71% of the Truthful population")
```

```
## [1] "truthful choosers who answered 'yes' represent 5/(5+2) or 71% of the Truthful population"
```

Part B.

```
rm(list=ls())


print("Seeking true_positives_P(D|PT). Find PT first")
```

```
## [1] "Seeking true_positives_P(D|PT). Find PT first"
```

```
sensitivity = 0.993 #P(PT|D)
specificity = 0.9999 #P(NT|ND)
Disease = 0.000025 #P(D)
No_Disease = (1-Disease) #P(ND)
No_Disease
```

```
## [1] 0.999975
```

```
false_positives = (1-specificity) #P(PT|ND)

Positive_Test = (sensitivity * Disease) + (false_positives*No_Disease) #P(PT)
Positive_Test
```

```
## [1] 0.0001248225
```

```
true_positives = sensitivity*Disease / Positive_Test #P(D|PT)
true_positives
```

```
## [1] 0.1988824
```

sensitivity = true_positives / (true_positives + false_negatives) #P(PT|D) specificity = true_negatives / (true_negatives + false_positives) #P(NT|ND) Negative_Test = #P(NT) true_negatives = #P(ND|NT) true_positives = #P(D|PT) false_negatives = #P(NT|D) false_positives = #P(PT|ND)

```
print("Suppose someone tests positive. What is the probability that they have the disease?
In light of this calculation, do you envision any problems in implementing a
universal testing policy for the disease?")
```

```
## [1] "Suppose someone tests positive. What is the probability that they have the disease? \nIn light
```

```
print("Given that someone tests positive on the test, there is approximately 19.88%
chance that the person will have the disease. The odds are pretty low. Depending on the disease and trea
it could be detrimental to treat a healthy person")
```

```
## [1] "Given that someone tests positive on the test, there is approximately 19.88%\nchance that the p
```

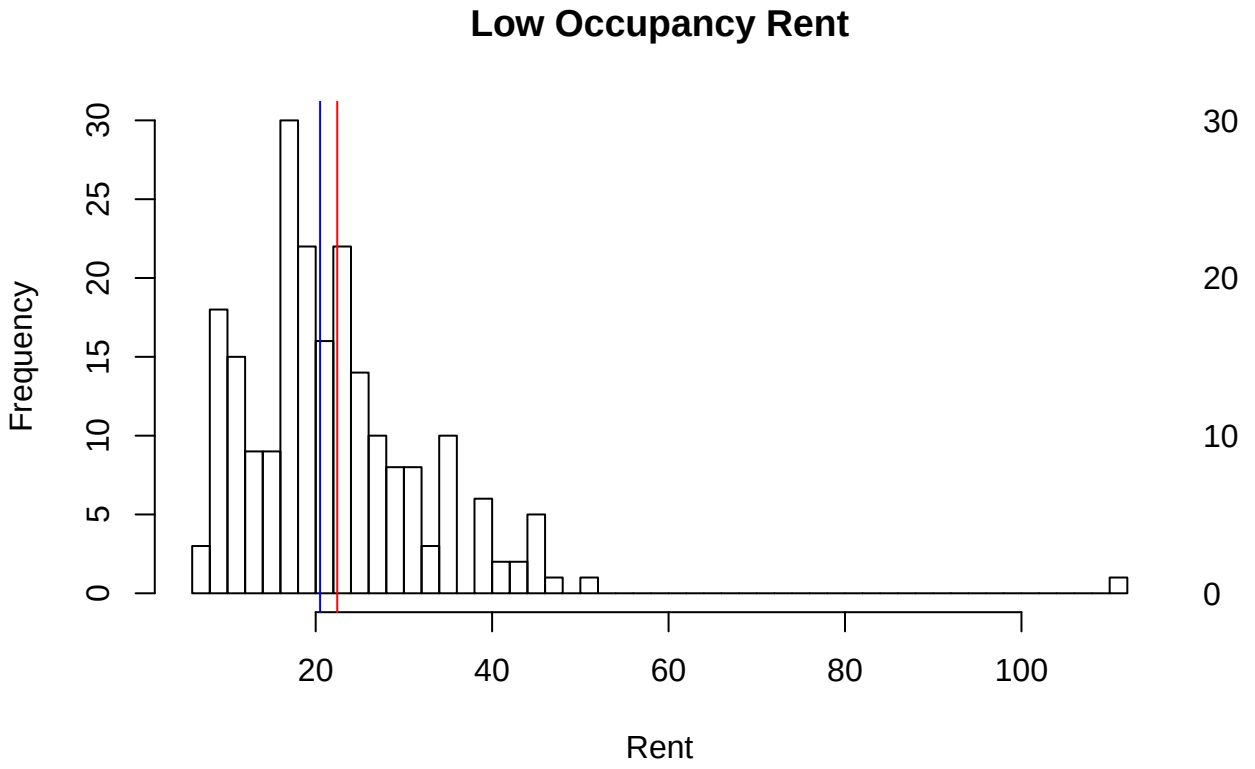
Exploratory analysis: green buildings

Examine analyst 10% occupancy rate comment

Analyst: very low occupancy rates (less than 10% of available space occupied). I decided to remove these buildings from consideration, on the theory that these buildings might have something weird going on with

them, and could potentially distort the analysis. Once I scrubbed these low-occupancy buildings from the data set, I looked at the green buildings and non-green buildings separately.

Clay: I think it's fine to remove these buildings. there are 215 out of 8k



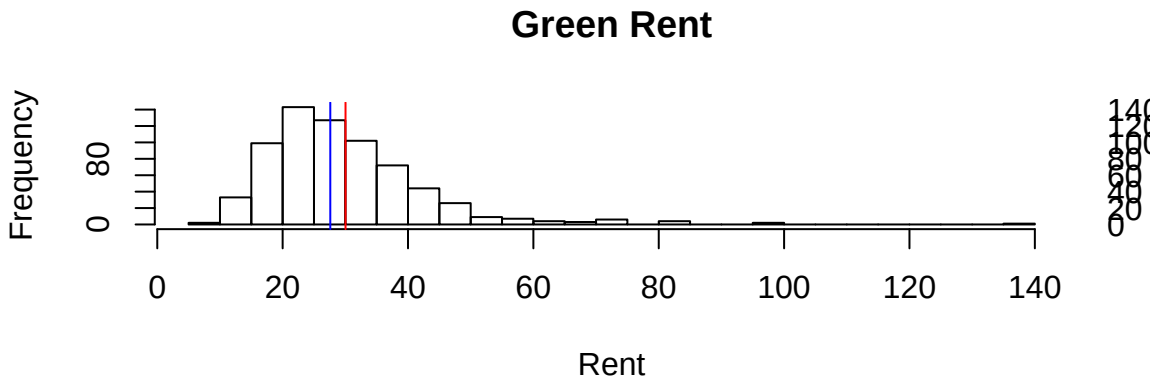
I looked at the green buildings and non-green buildings separately.

```
# Extract the buildings with green ratings
green_only = subset(green, green_rating==1)
#dim(green_only)
#summary(green_only)

# define other side of the data set
non_green = subset(green, green_rating !=1)
#dim(non_green)
#summary(non_green)
```

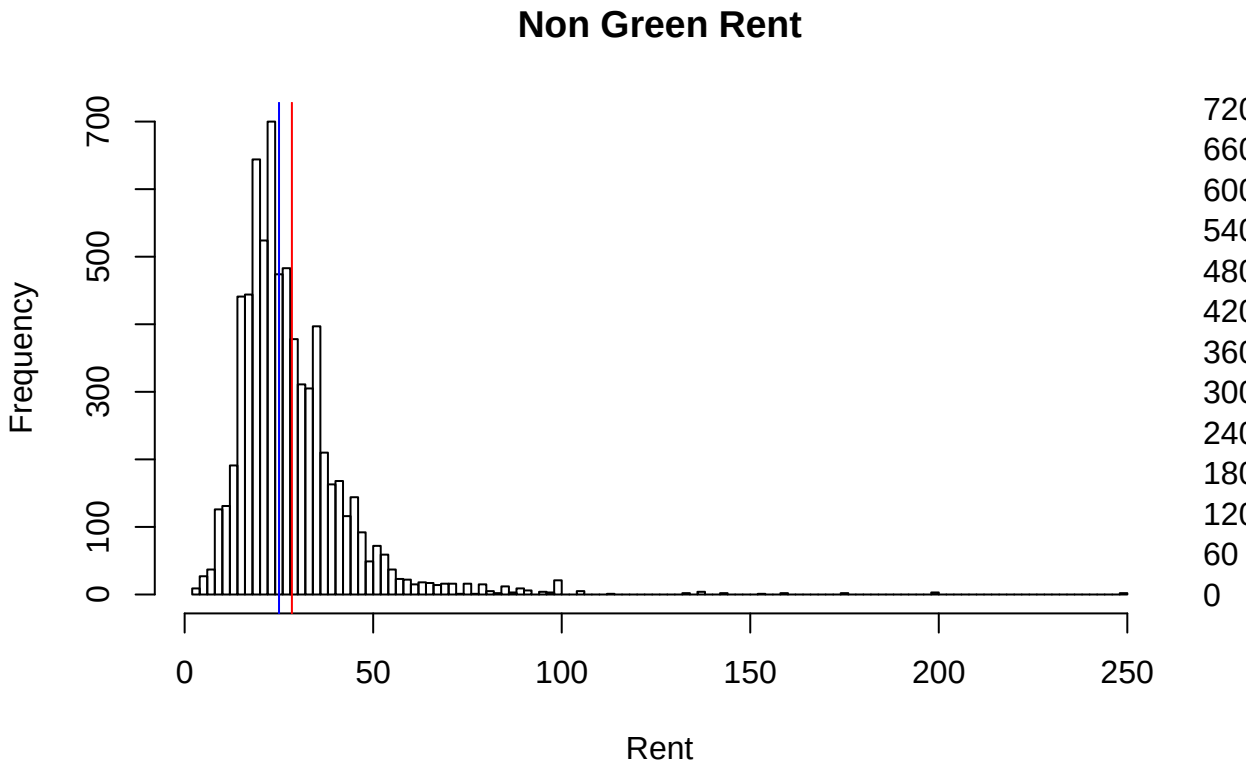
Examine analyst 2.60/sq ft premium comment

Analyst: The median market rent in the non-green buildings was $25 per square foot per year, while the median market rent in the green buildings was $27.60 per square foot per year: about $2.60 more per square foot. (I used the median rather than the mean, because there were still some outliers in the data, and the median is a lot more robust to outliers.)

Clay: the histogram data does show skewed data. the mean is much higher than the median. the histogram graphs plotted on top of each other shows the difference in the median price for the data sets. It appears that the Green buildings do have a premium.

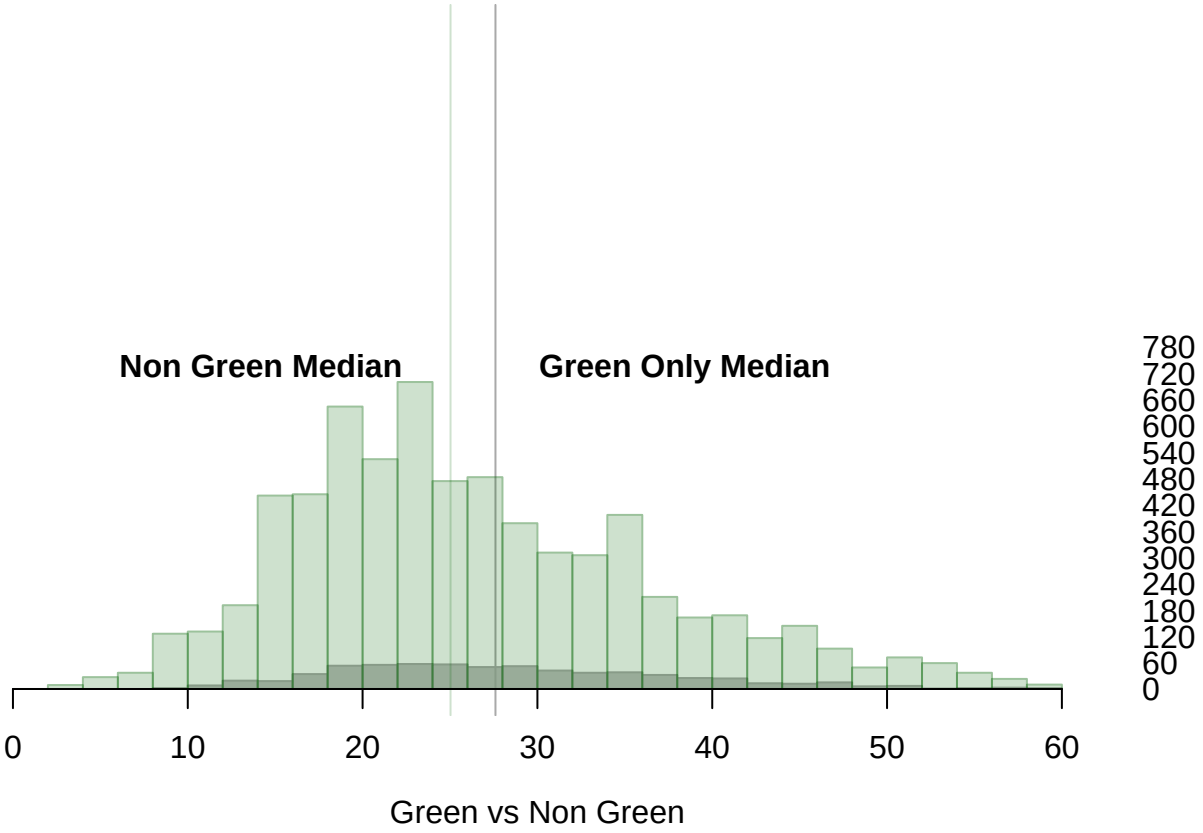



```
# non_green histogram- skewed data (mean is higher than median)
mean_non_green = mean(non_green$Rent)
hist(non_green$Rent, 100, main="Non Green Rent", xlab='Rent')
axis(4,at=seq(0,800,by=60), las=1,tick=FALSE)
abline(v=mean_non_green, col='red')
abline(v=median(non_green$Rent), col='blue')
```




```
#plot histograms on top of each other.
mybreaks = seq(0, 60, by=2)
par(mfrow=c(1,1), mar=c(3,0,1,3), mgp=c(2,1,0))
hist(green_only$Rent[green_only$Rent < 60], breaks=mybreaks, xlab="Green vs Non Green", main="", border=

abline(v=median(green_only$Rent),col ='darkgrey')

hist(non_green$Rent[non_green$Rent < 60],breaks=mybreaks,add=TRUE, border=rgb(0,100,0,100,maxColorValue=
abline(v=median(non_green$Rent),col =rgb(0,100,0,50,maxColorValue=255))
axis(4,at=seq(0,800,by=60), las=1,tick=FALSE)
axis(1,pos=0)
```

```
text(5, 730, "Non Green Median", pos=4, font=2)
text(29, 730, "Green Only Median", pos=4, font=2)
```



examine cost benefit analysis

Analyst: Because our building would be 250,000 square feet, this would translate into an additional $250000 x 2.6 = $650000 of extra revenue per year if we build the green building. Our expected baseline construction costs are $100 million, with a 5% expected premium for green certification. Thus we should expect to spend an extra $5 million on the green building.

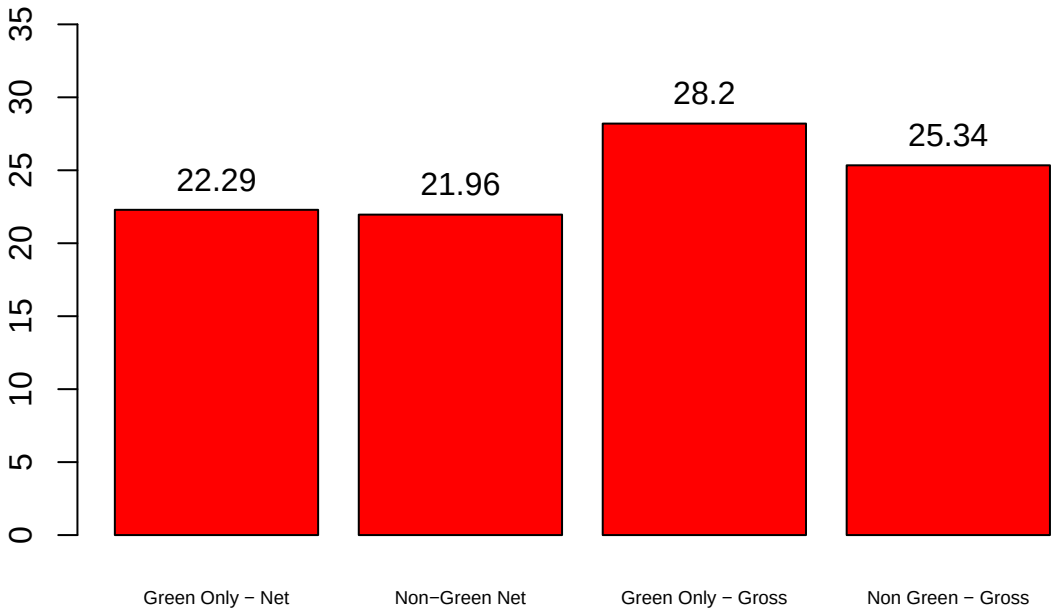
Based on the extra revenue we would make, we would recuperate these costs in $5000000/650000 = 7.7 years. Even if our occupancy rate were only 90%, we would still recuperate the costs in a little over 8 years. Thus from year 9 onwards, we would be making an extra $650,000 per year in profit. Since the building will be earning rents for 30 years or more, it seems like a good financial move to build the green building.

Clay: let's try to see if premiums are consistent based on net and gross rents

```
green_only_net = green_only[green_only$net==1,]
non_green_net = non_green[non_green$net==1,]
green_only_gross = green_only[green_only$net==0,]
non_green_gross = non_green[non_green$net==0,]

A = median(green_only_net$Rent)
B = median(non_green_net$Rent)
C = median(green_only_gross$Rent)
D = median(non_green_gross$Rent)

#A
#B
```

```
#C
#D
```

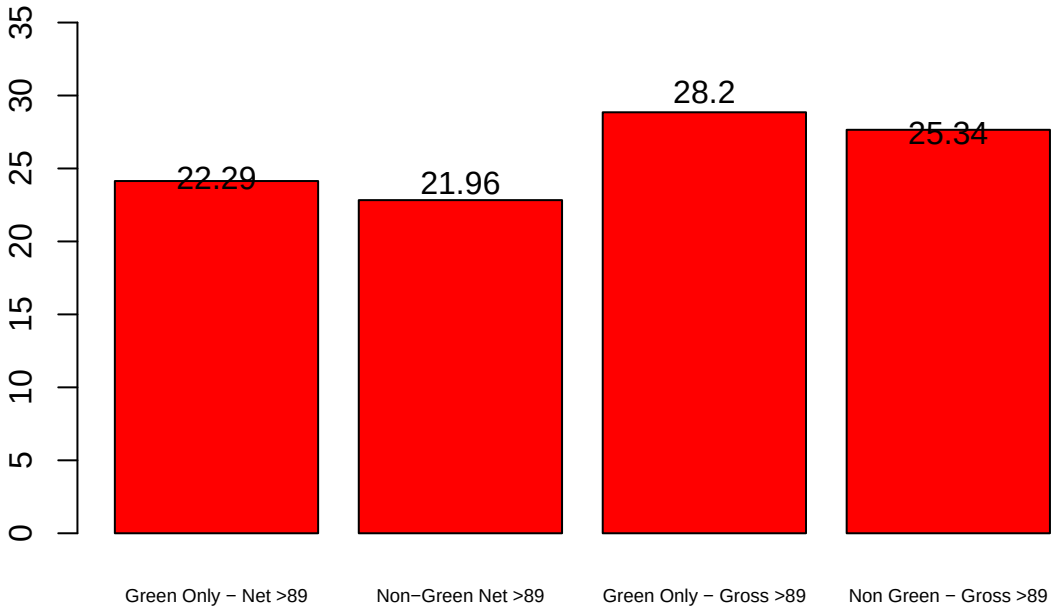
Clay: it appears that utilities are driving the price difference between Green and Non Green Rental Rates based on the Gross Rental rates shown in this bar chart. the rental rates are pretty close between green and not green buildings when utilities are not included. The company may be able to receive a higher premium than previously thought. The net rents dragged the overall median rentals closer together.



Clay: let's examine just buildings with over 90% occupancy to make sure that the premium is consistent despite occupancy rates. occupancy rate could be considered a proxy for how in demand a building is, so I wondered if it was possible that the premium could be higher.

```
green_only_net_90 = green_only_net[green_only_net$leasing_rate>89,]
non_green_net_90 = non_green_net[non_green_net$leasing_rate>89,]
green_only_gross_90 = green_only_gross[green_only_gross$leasing_rate>89,]
non_green_gross_90 = non_green_gross[non_green_gross$leasing_rate>89,]

A_90 = median(green_only_net_90$Rent)
B_90 = median(non_green_net_90$Rent)
C_90 = median(green_only_gross_90$Rent)
D_90 = median(non_green_gross_90$Rent)

#A_90
#B_90
#C_90
#D_90

Type_Building_90 <- c('Green Only - Net >89','Non-Green Net >89','Green Only - Gross >89', 'Non Green -
Median_building_90 <- c(A_90, B_90, C_90,D_90)

Building_plot_90 <- data.frame(Type_Building_90, Median_building_90)


posbar_90 = barplot(Building_plot_90$Median_building_90,
                    names.arg=Building_plot_90$Type_Building_90,
```

```
                  col = 'red',
                  axisnames = TRUE,
                  axes = TRUE,
                  beside = TRUE,
                  cex.names=0.6,
                  ylim = c(0,35))
text(y=Building_plot$Median_building, x=posbar_90, pos=3,labels=Building_plot$Median_building)
```




```
# dev.off()
```

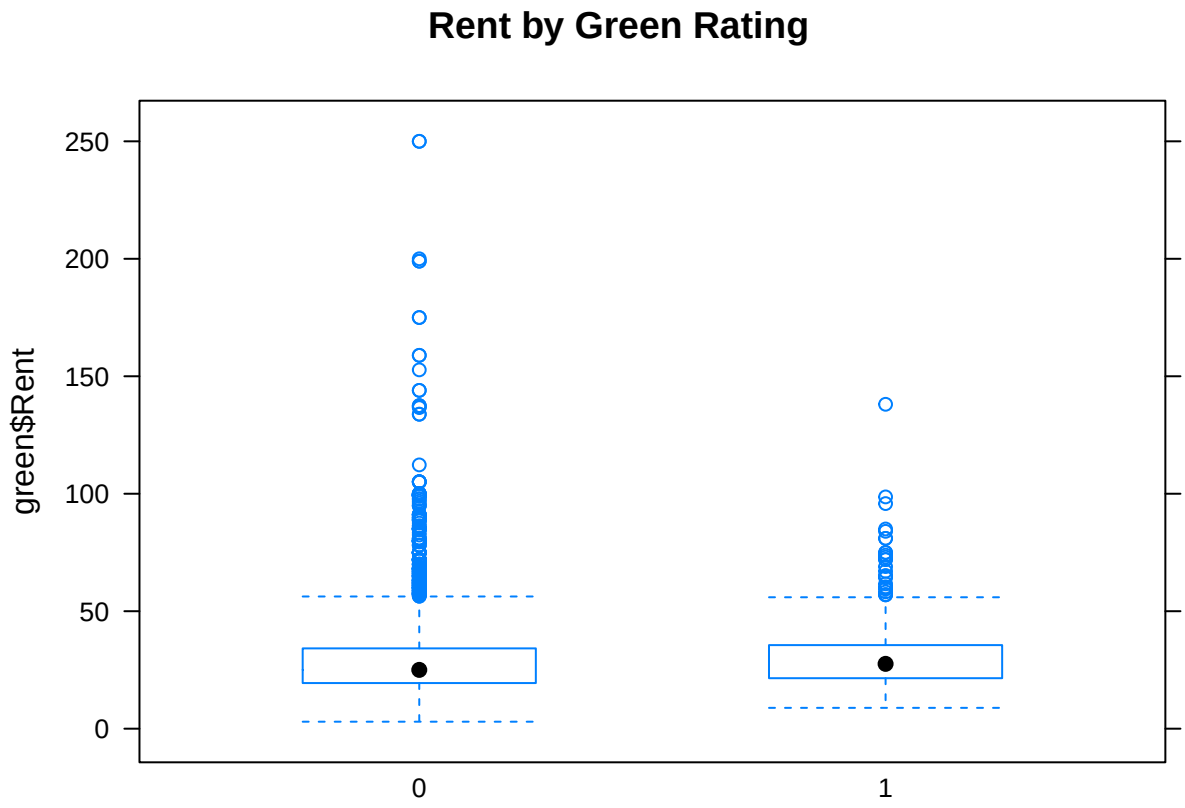
Clay: let's view the correlation matrix to determine if there are any other relationships worth noting. obvious correlations exist between Size and Stories, # of Cooling days and Gas costs, and # of heating days and electricity costs. cold places require more gas to heat buildings and warm places require more electricity to cool the building. As far as our target variable, Rent, the main correlation driver is Cluster Rent, which is not surprising given that field is based on buildings that sit geographically close to each other.

```
green_numeric = green[,c("size",
                         "Rent",
                         "leasing_rate",
                         "stories",
                         "age",
                         "cd_total_07",
                         "hd_total07",
                         "total_dd_07",
                         "Precipitation",
                         "Gas_Costs",
                         "Electricity_Costs",
                         "cluster_rent")]

#head(green_numeric,10)
matrixxx = cor(green_numeric)
round(matrixxx, 2)
```

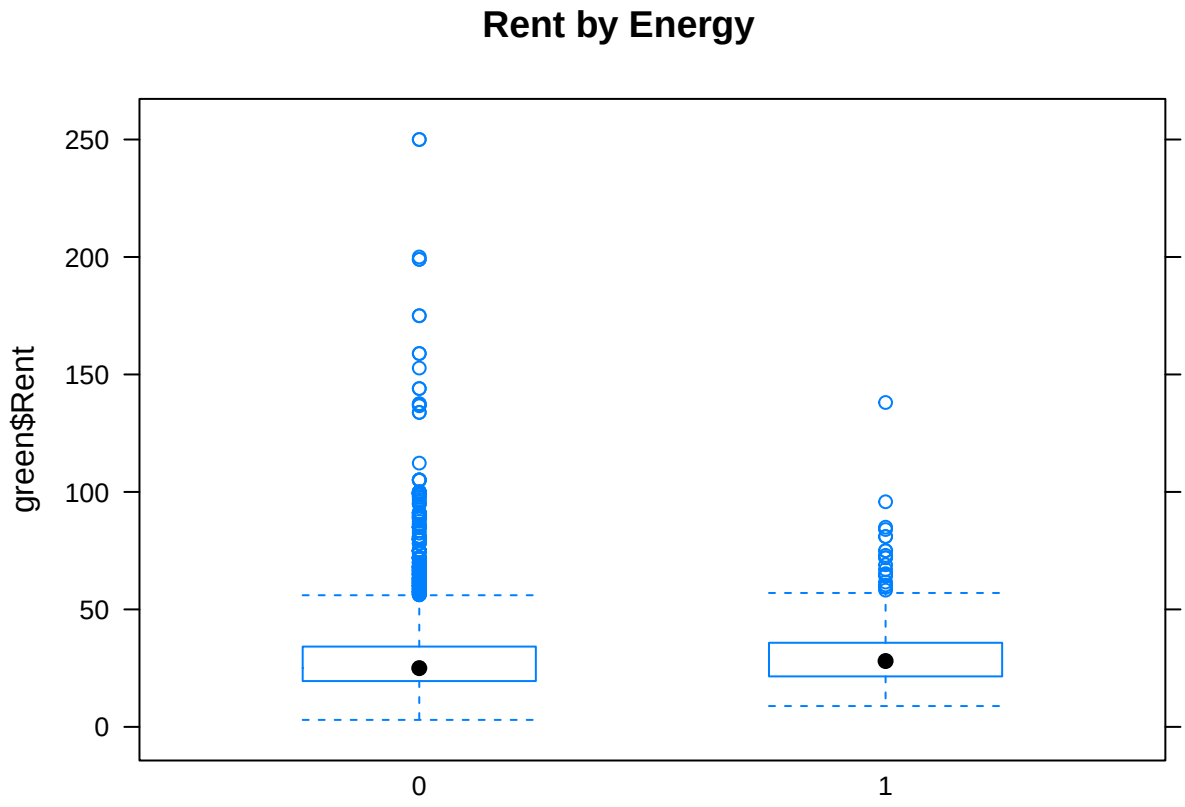
```
##                    size  Rent leasing_rate stories   age cd_total_07
## size               1.00  0.13         0.17    0.83 -0.20        0.07
```

```
## Rent               0.13  1.00         0.18    0.11 -0.10       -0.16
## leasing_rate       0.17  0.18         1.00    0.17 -0.13       -0.02
## stories            0.83  0.11         0.17    1.00 -0.15        0.04
## age               -0.20 -0.10        -0.13   -0.15  1.00       -0.22
## cd_total_07        0.07 -0.16        -0.02    0.04 -0.22        1.00
## hd_total07         0.15 -0.16         0.00    0.19  0.24       -0.27
## total_dd_07        0.19 -0.25        -0.01    0.21  0.11        0.29
## Precipitation      0.16  0.07         0.02    0.19  0.08        0.18
## Gas_Costs          0.01  0.00         0.04    0.06 -0.06        0.46
## Electricity_Costs -0.10  0.40         0.06   -0.09 -0.05       -0.06
## cluster_rent      -0.03  0.76         0.17   -0.02 -0.03       -0.20
##                   hd_total07 total_dd_07 Precipitation Gas_Costs
## size                    0.15       0.19          0.16      0.01
## Rent                   -0.16      -0.25          0.07      0.00
## leasing_rate            0.00      -0.01          0.02      0.04
## stories                 0.19       0.21          0.19      0.06
## age                     0.24       0.11          0.08     -0.06
## cd_total_07            -0.27       0.29          0.18      0.46
## hd_total07              1.00       0.85          0.32     -0.06
## total_dd_07             0.85       1.00          0.41      0.19
## Precipitation           0.32       0.41          1.00      0.59
## Gas_Costs              -0.06       0.19          0.59      1.00
## Electricity_Costs      -0.63      -0.66         -0.17      0.23
## cluster_rent           -0.24      -0.35          0.09      0.03
##                   Electricity_Costs cluster_rent
## size                          -0.10       -0.03
## Rent                           0.40        0.76
## leasing_rate                   0.06        0.17
## stories                       -0.09       -0.02
## age                           -0.05       -0.03
## cd_total_07                   -0.06       -0.20
## hd_total07                    -0.63       -0.24
## total_dd_07                   -0.66       -0.35
## Precipitation                 -0.17        0.09
## Gas_Costs                      0.23        0.03
## Electricity_Costs              1.00        0.51
## cluster_rent                   0.51        1.00
```

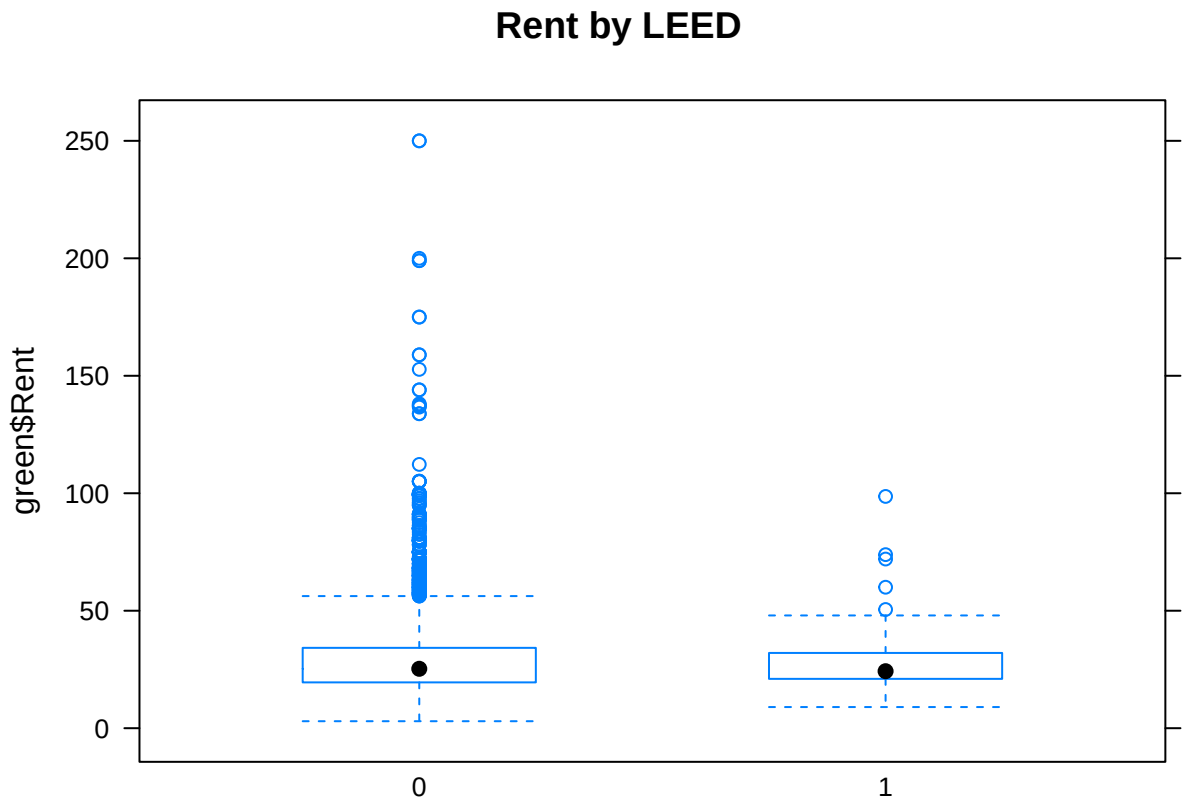
```
#corrplot(matrixxx)
#head(green_numeric,5)

#colnames(green)




bwplot(green$Rent ~ green_rating, data=green, main="Rent by Green Rating")
```



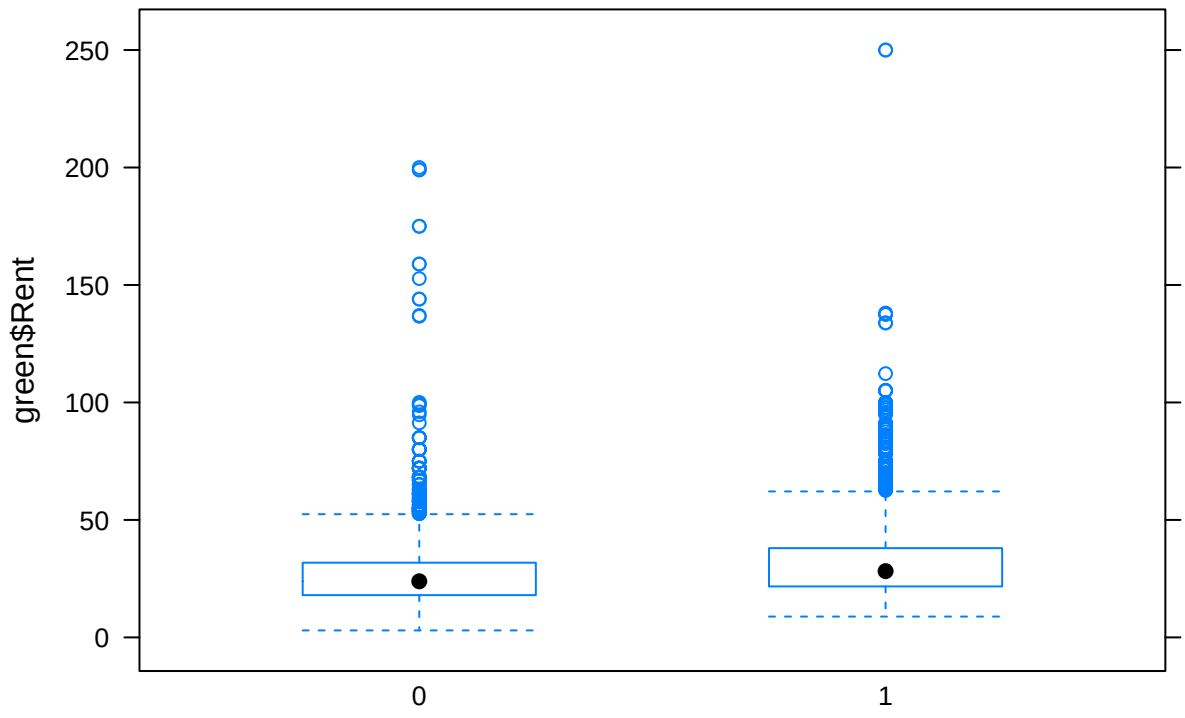

```
bwplot(green$Rent ~ Energystar, data=green, main="Rent by Energy")
```



```
bwplot(green$Rent ~ LEED, data=green, main="Rent by LEED")
```

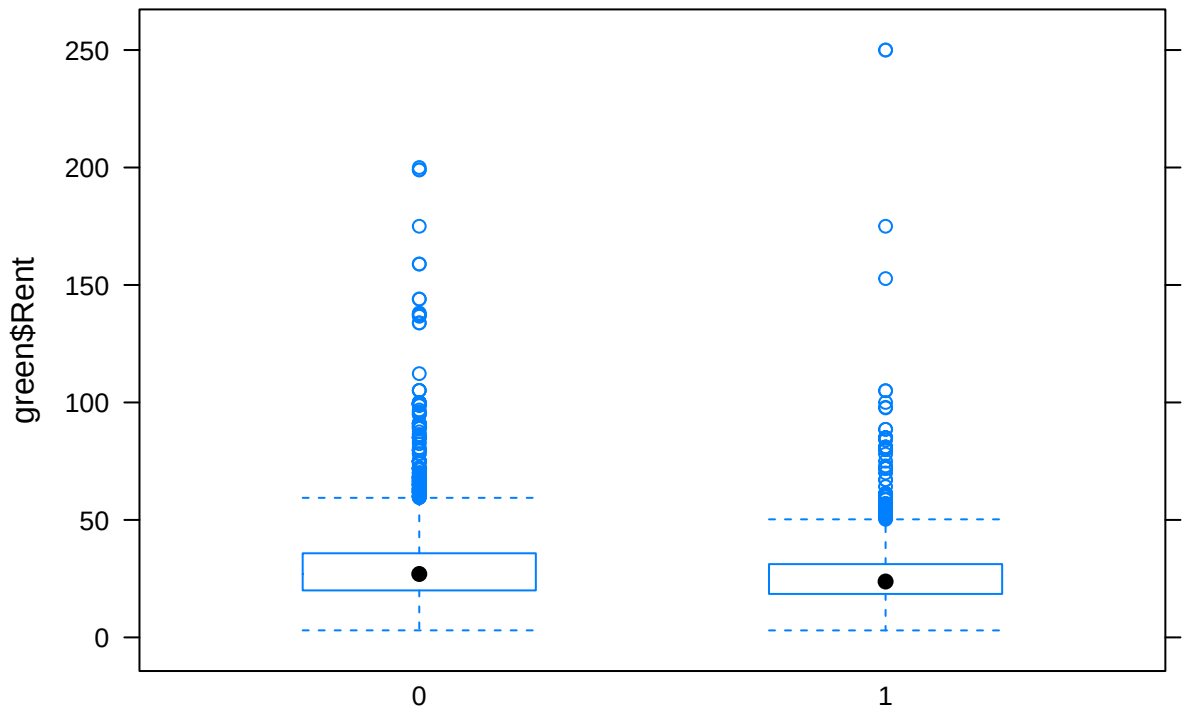



```
bwplot(green$Rent ~ class_a, data=green, main="Rent by class_a")
```

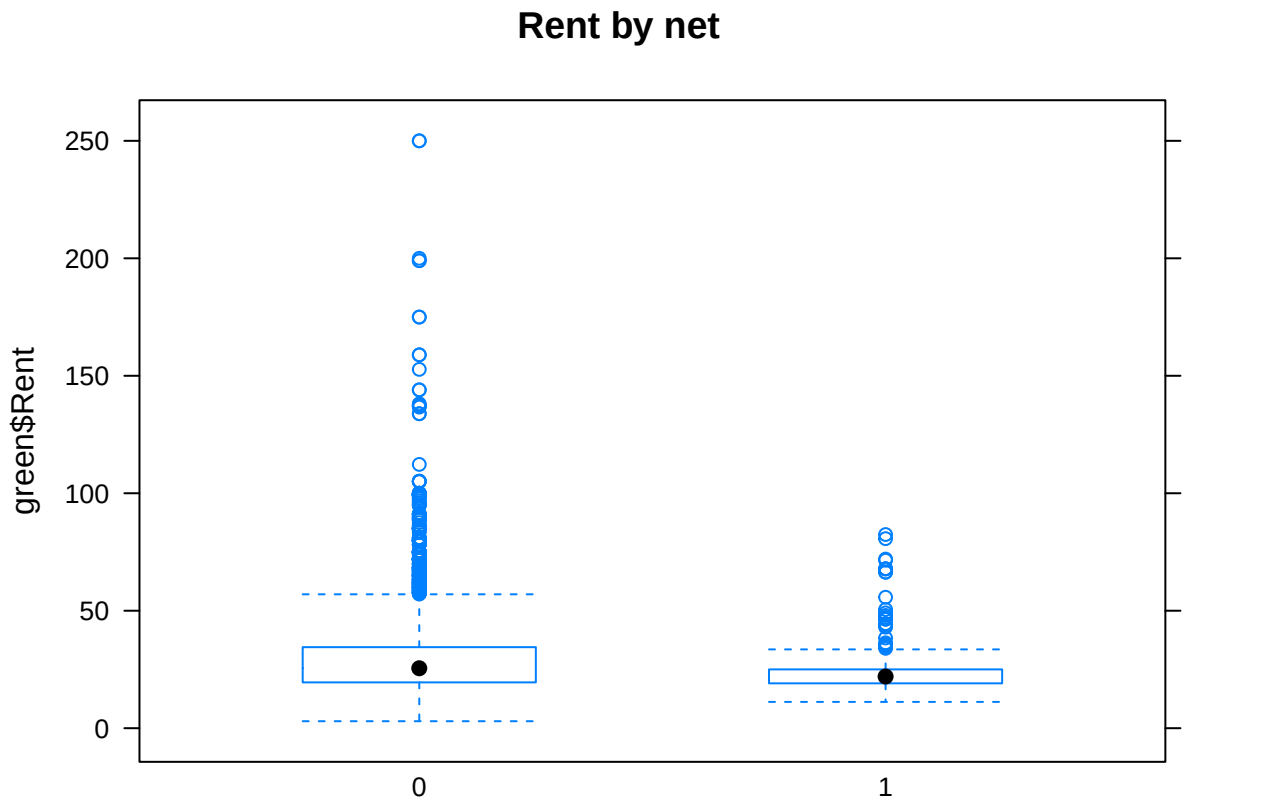


```
bwplot(green$Rent ~ class_b, data=green, main="Rent by class_b")
```

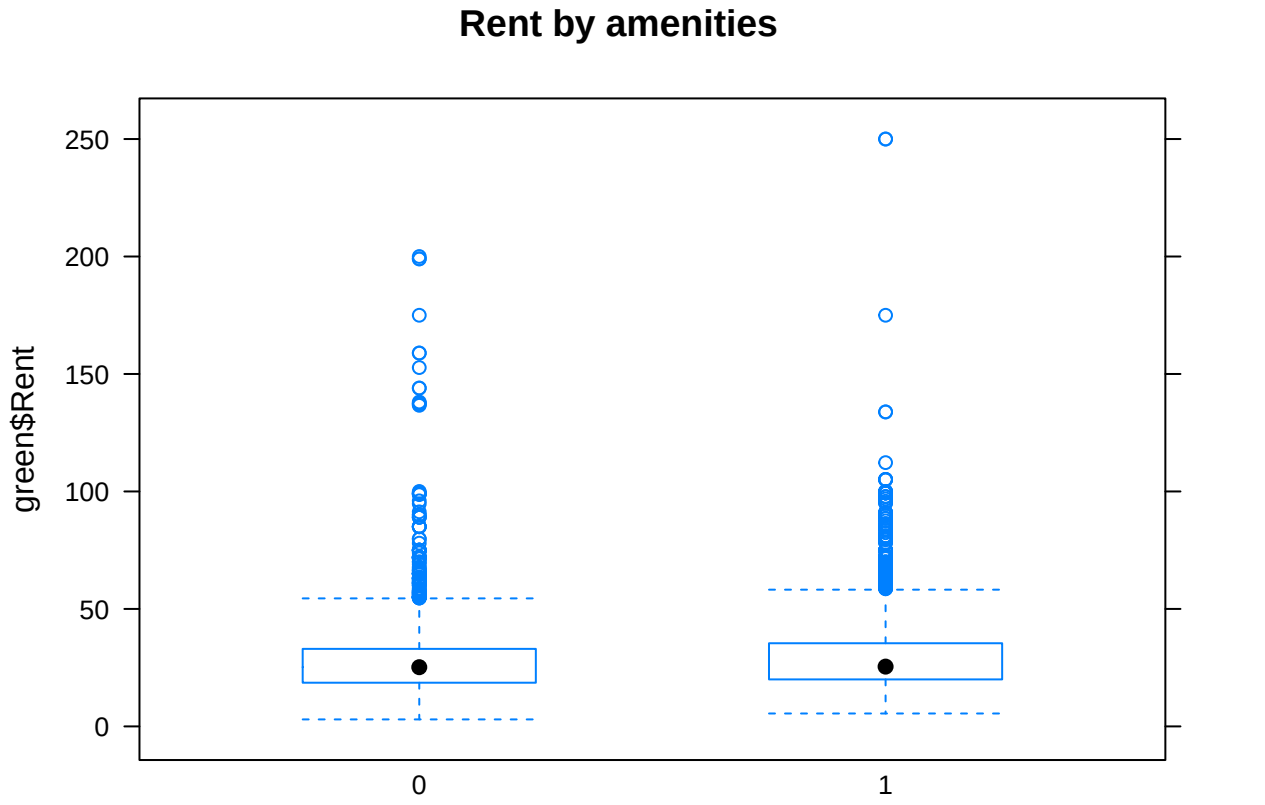

```
bwplot(green$Rent ~ renovated, data=green, main="Rent by renovation")
```



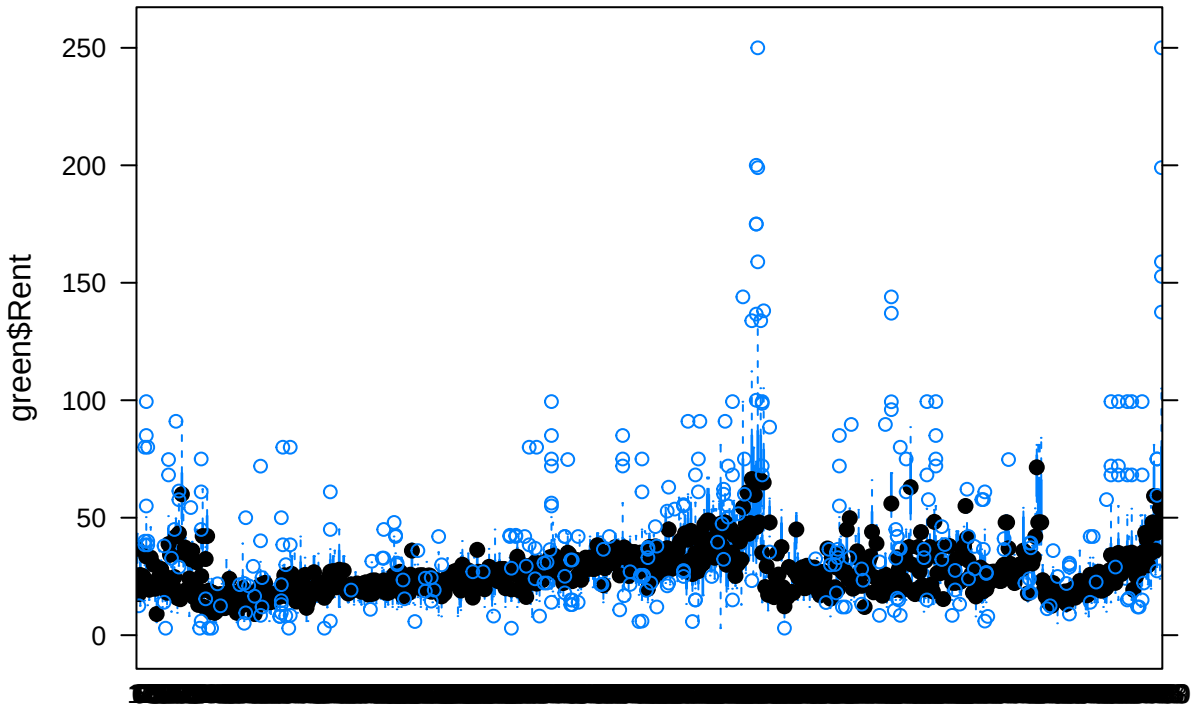
```
bwplot(green$Rent ~ net, data=green, main="Rent by net")
```




```
bwplot(green$Rent ~ amenities, data=green, main="Rent by amenities")
```



```
bwplot(green$Rent ~ cluster, data=green, main="Rent by Green Rating")
```



explore class a vs class b

```
mask1 = (green$class_a ==1)
mask2 = (green$class_b ==1)



dev.off()
```

```
## null device
##           1
```

```
hist(green$Rent[mask2], breaks=80, xlab="class a vs class b", main="", border=rgb(0,100,0,100,maxColorV
hist(green$Rent[mask1], breaks = 200, add=TRUE,, border="darkgrey", col="grey", axes=FALSE, ylim=c(0, 8
abline(v=median(green$Rent[mask1]),col ='darkgrey')
abline(v=median(green$Rent[mask2]),col =rgb(0,100,0,50,maxColorValue=255))
axis(4,at=seq(0,840,by=60), las=1,tick=FALSE)
# axis(1,pos=0)
text(-5, 360, "class b", pos=4, font=2)
text(33, 360, "class a", pos=4, font=2)



mask1 = (green_only$class_a ==1)
mask2 = (non_green$class_a ==1)
mask3 = (green_only$class_b ==1)
mask4 = (non_green$class_b ==1)
```

```
#median(green_only$Rent[mask1])
#median(non_green$Rent[mask2])
#median(green_only$Rent[mask3])
#median(non_green$Rent[mask4])

Type_Building2 <- c('Green Only - Class a','Non-Green - class a','Green Only - class b', 'Non Green - cl
Median_building2 <- c(A, B, C,D)

Building_plot2 <- data.frame(Type_Building2, Median_building2)

#
# ggplot(Building_plot, aes(x = Building_plot$Type_Building, y = Building_plot$Median_building)) +
#   geom_bar(stat = "identity") +
#   geom_text(aes(label = format(Building_plot$Median_building, digits = 4), y = 0.7))


posbar = barplot(Building_plot2$Median_building2,
                 names.arg=Building_plot2$Type_Building2,
                 col = 'red',
                 axisnames = TRUE,
                 axes = TRUE,
                 beside = TRUE,
                 cex.names=0.6,
                 ylim = c(0,35))
text(y=Building_plot2$Median_building2, x=posbar, pos=3,labels=Building_plot2$Median_building2)

dev.off()
```

```
## null device
##           1
```

explore clusters

Visually you can see that green buildings charge higher rent for each cluster. The premium is about 2.48 at the cluster level if you subtract the green minus cluster median rent

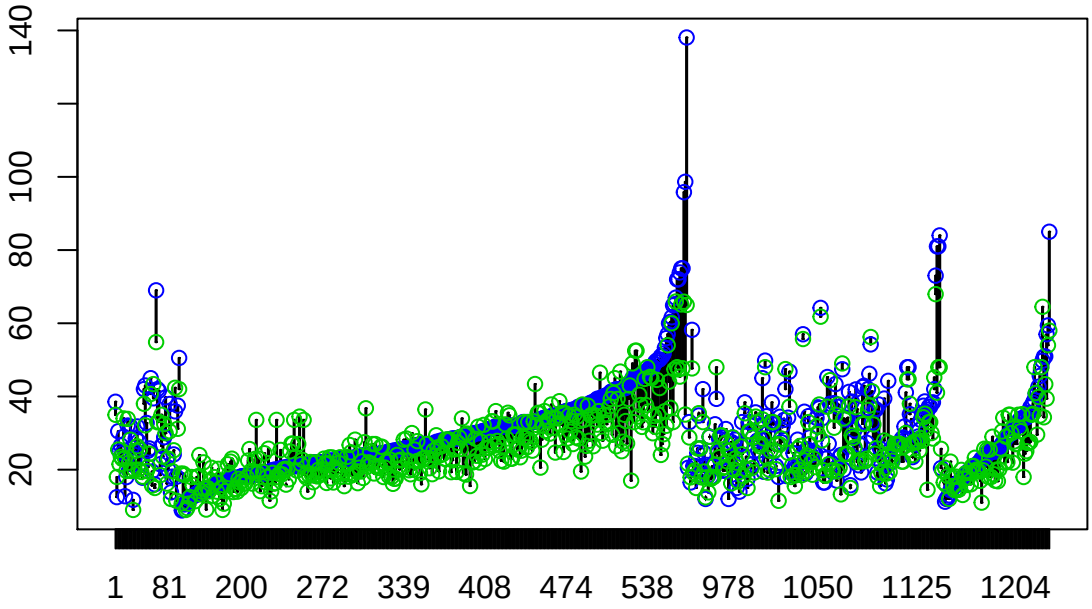
```
library(dplyr)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:Matrix':
##
##     expand
```

```
x = green %>%
  group_by(cluster, green_rating) %>%
  summarise(median(Rent))
#colnames(x)
cluster_green_only = x[x$green_rating==1,]
cluster_non_green = x[x$green_rating==0,]
```

```
premium = median(cluster_green_only$`median(Rent)`) - median(cluster_non_green$`median(Rent)`)
plot(x$cluster,x$`median(Rent)`,type = 'p')
points(x$cluster[x$green_rating==1],x$`median(Rent)`[x$green_rating==1],col=4,pch=1)
points(x$cluster[x$green_rating==0],x$`median(Rent)`[x$green_rating==0],col=3,pch=1)
```




```
premium
```

```
## [1] 2.48
```

Final Conclusion

Based on the fact that the premium is verified through multiple analyses, i would recommend pushing forward with the project. this is not an exhaustive analysis and there could be other factors at work, but i have not found anything that rejects the Analysts' claims.

Bootstrapping Finance Problem

```
# Import a few stocks #
mystocks = c("SPY", "TLT", "LQD","EEM","VNQ")
getSymbols(mystocks, from = "2005-01-01")
```

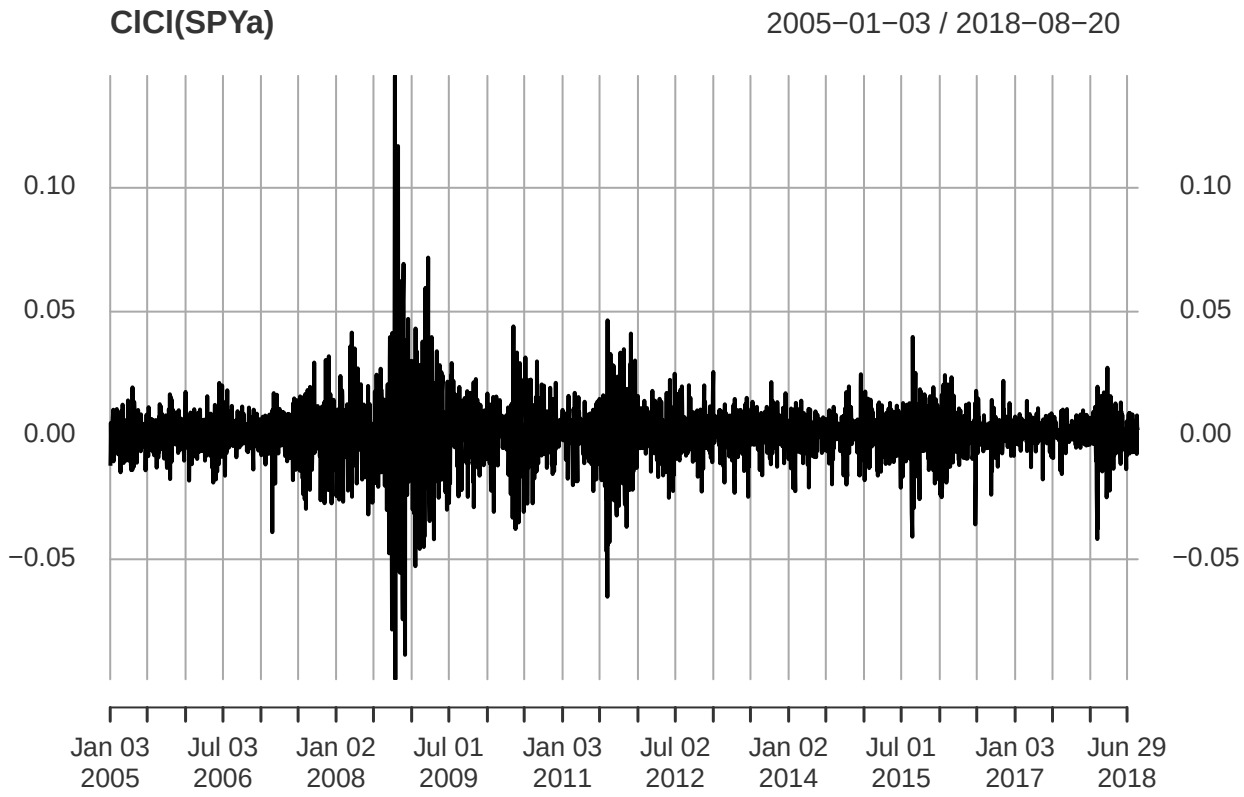
```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
```

```
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).
```

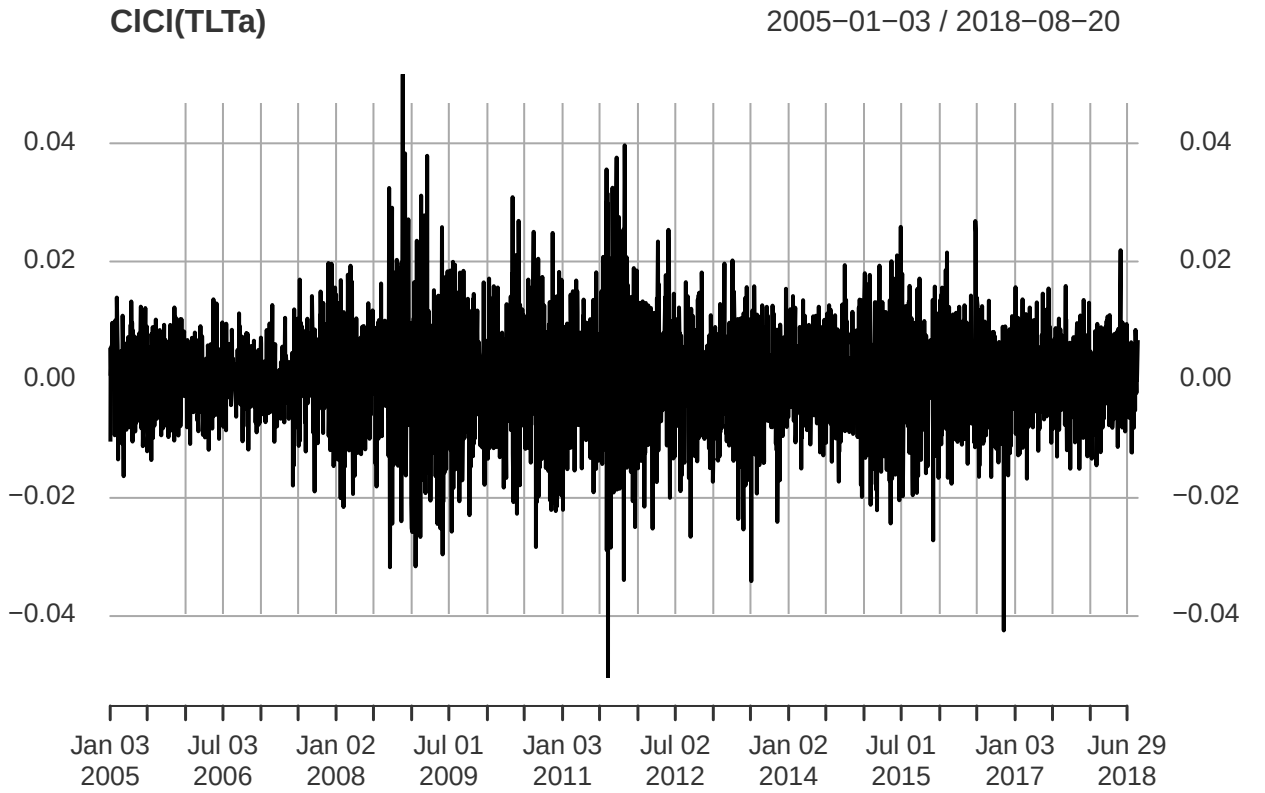
```
## [1] "SPY" "TLT" "LQD" "EEM" "VNQ"
```

```
# Adjust for splits and dividends #

for(ticker in mystocks) {
  expr = paste0(ticker, "a = adjustOHLC(", ticker, ")")
  eval(parse(text=expr))
}
```

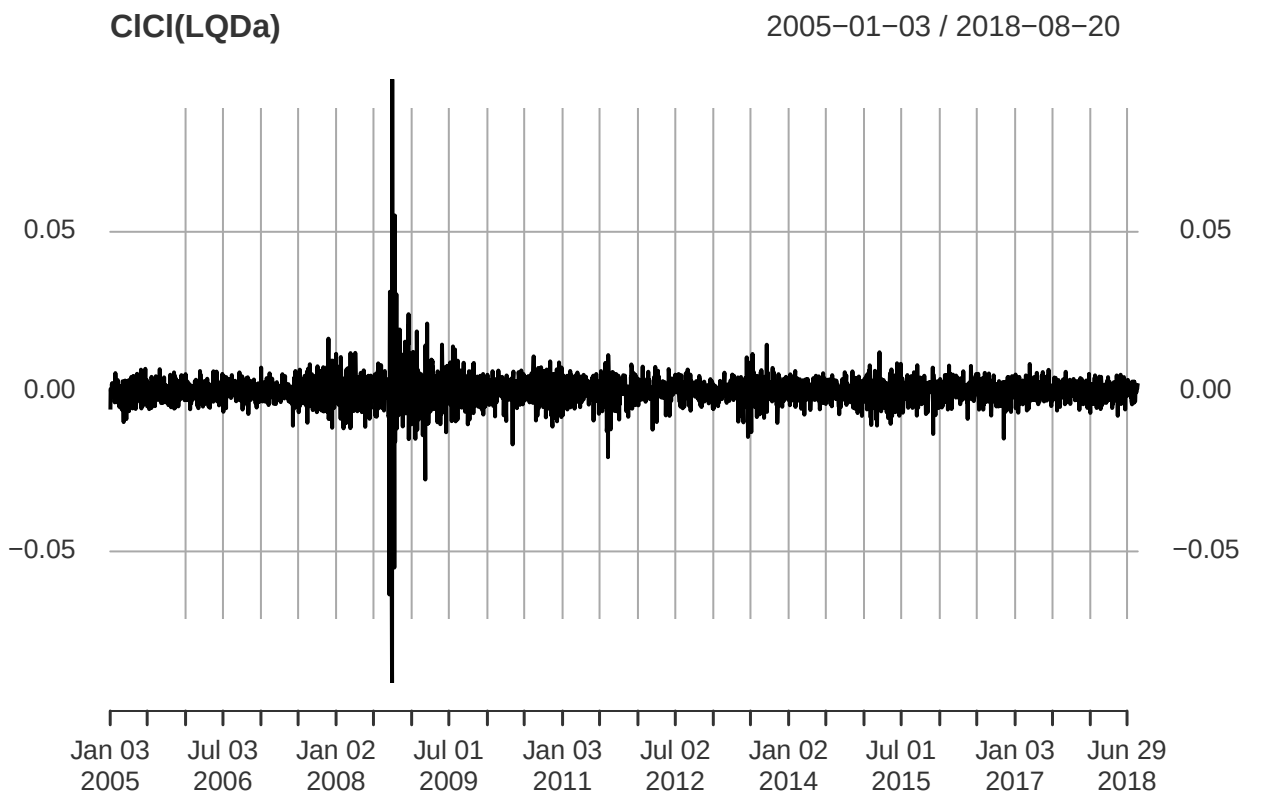
```
# Look at close-to-close changes #

plot(ClCl(SPYa))
```



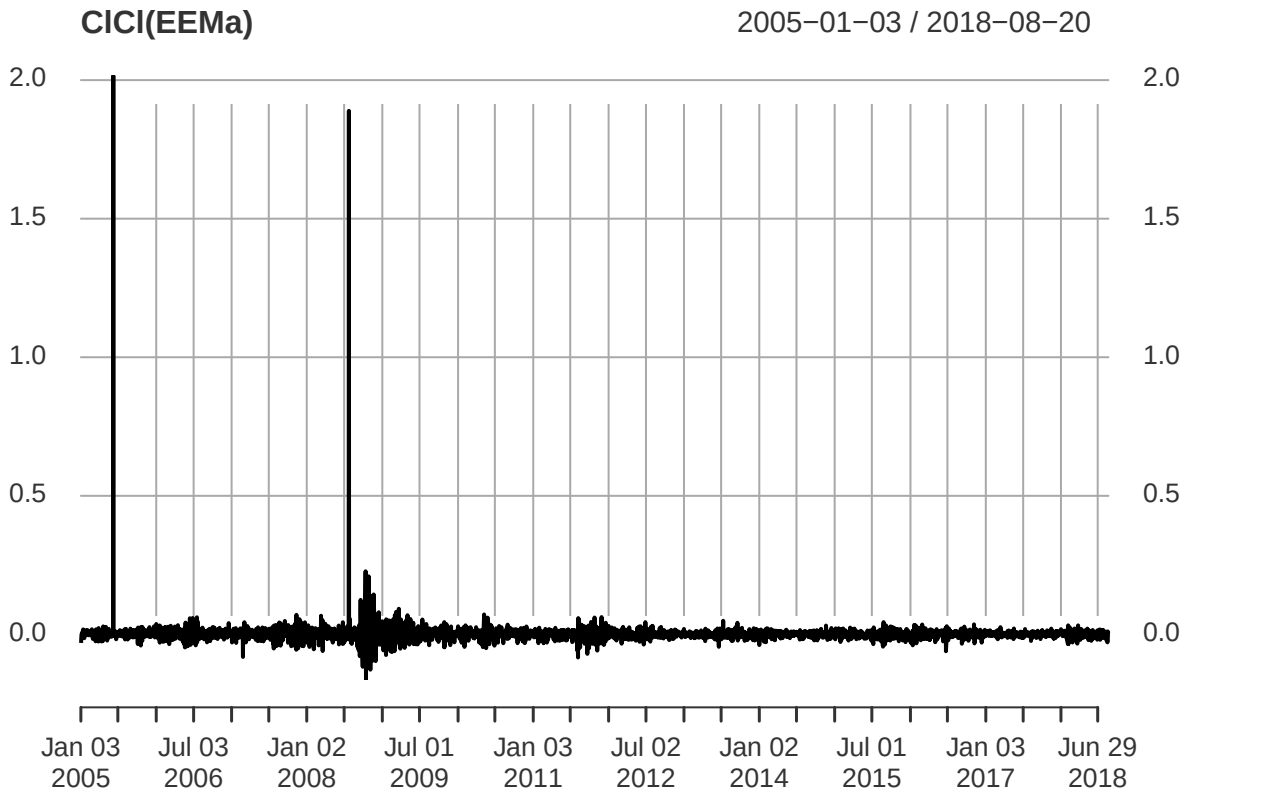

```
plot(ClCl(TLTa))
```



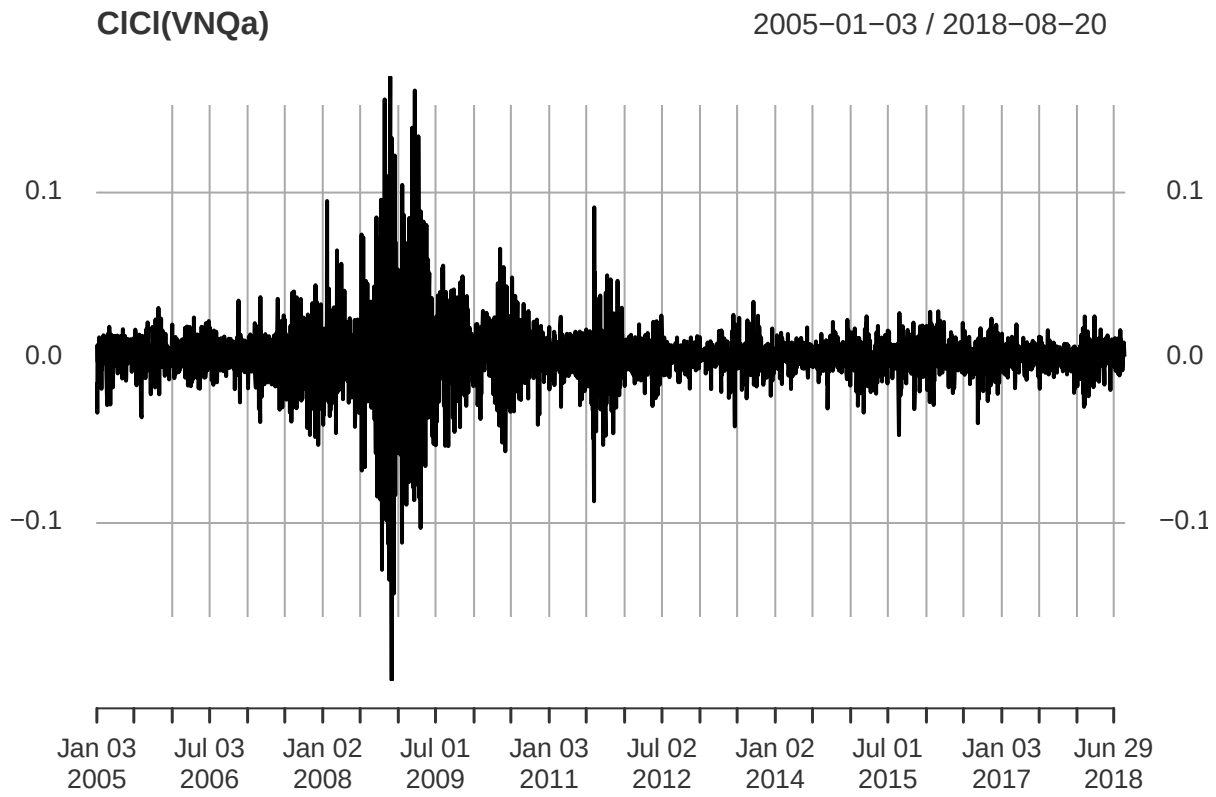

```
plot(ClCl(LQDa))
```



```
plot(ClCl(EEMa))
```



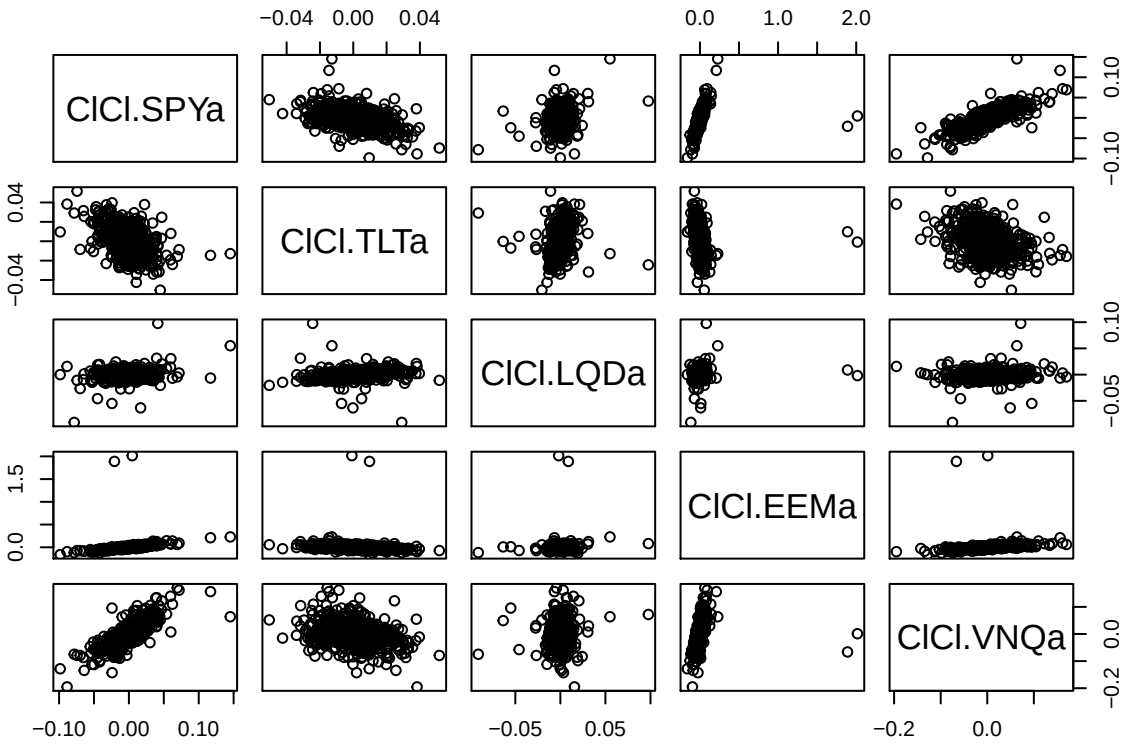

```
plot(ClCl(VNQa))
```



```
# make sure we have returns for the full sample. 2005 was first year for all 5 indices to have their ret
#head(SPYa)
#head(TLTa)
#head(LQDa)
#head(EEMa)
#head(VNQa)



# Combine close to close changes in a single matrix #

all_returns = cbind(
  ClCl(SPYa),
  ClCl(TLTa),
  ClCl(LQDa),
  ClCl(EEMa),
  ClCl(VNQa)
  )
```

Look at returns of the various asset classes

```
#head(all_returns,2)
all_returns = as.matrix(na.omit(all_returns))

# These returns can be viewed as draws from the joint distribution #
# Compute the returns from the closing prices #
pairs(all_returns)
```




```
print ("Look at the market/S&P 500 returns over time")
```

```
## [1] "Look at the market/S&P 500 returns over time"
```

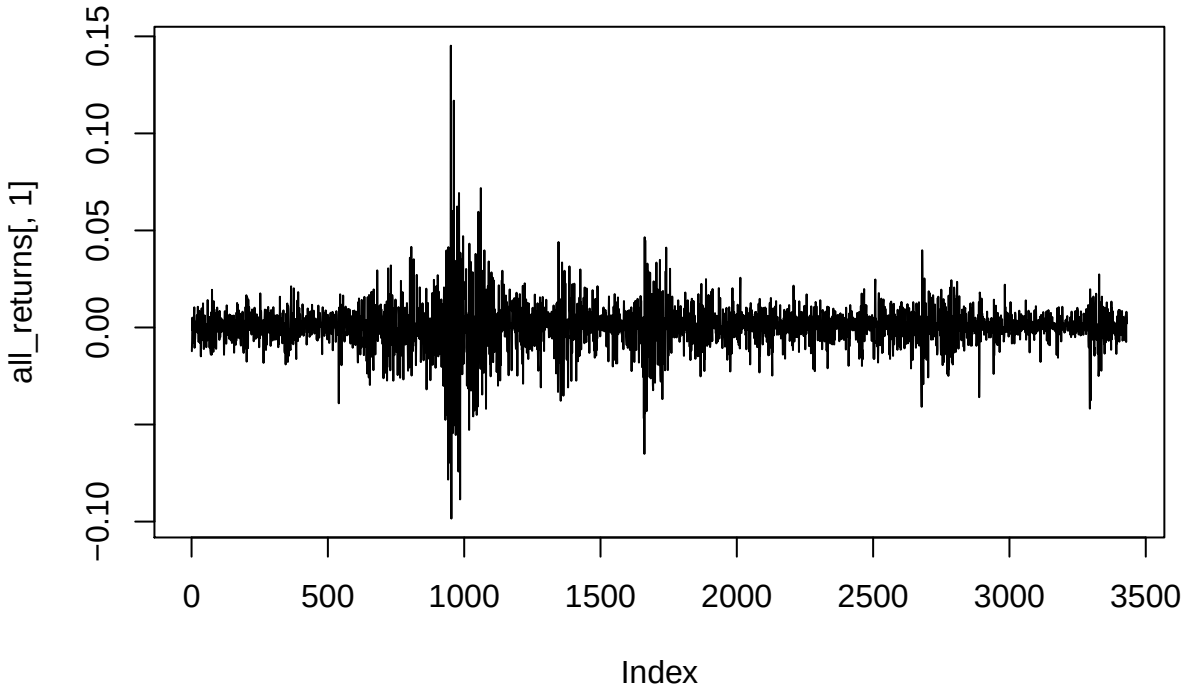
```
print ("Annual Standard Deviation: ")
```

```
## [1] "Annual Standard Deviation: "
```

```
sd(all_returns[,1])*sqrt(250)
```

```
## [1] 0.1857294
```

```
plot(all_returns[,1], type='l')
```




```
print ("Look at US Treasury bonds (TLT) returns over time")
```

```
## [1] "Look at US Treasury bonds (TLT) returns over time"
```

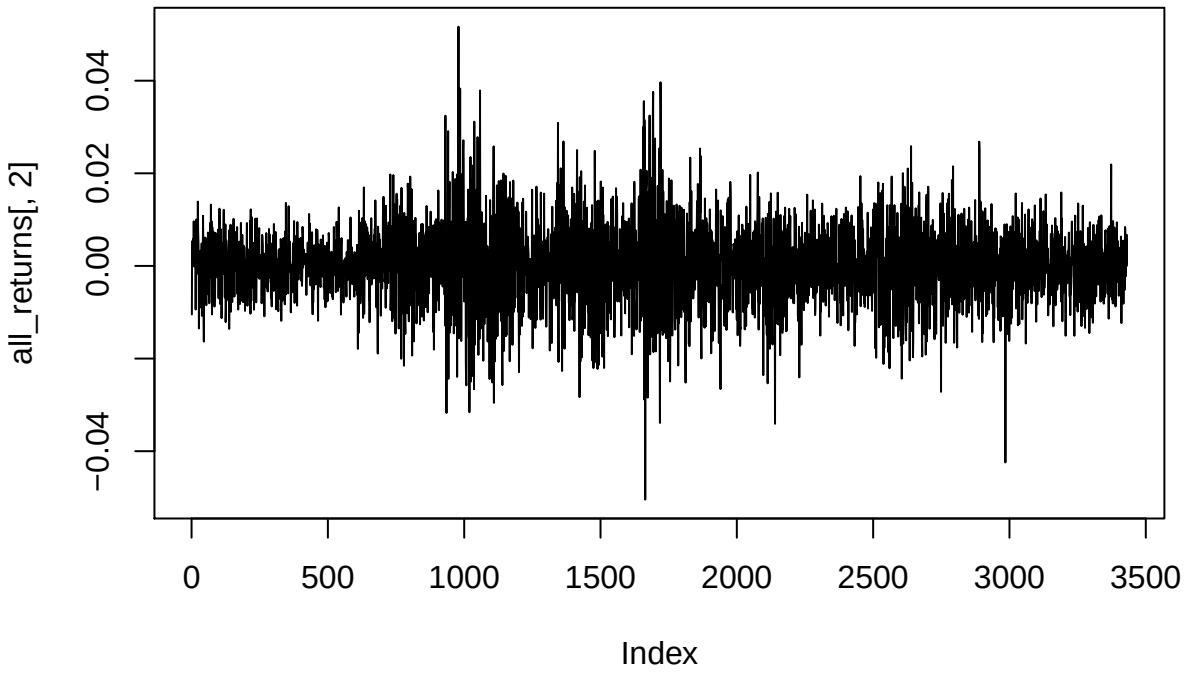
```
print ("Annual Standard Deviation: ")
```

```
## [1] "Annual Standard Deviation: "
```

```
sd(all_returns[,2])*sqrt(250)
```

```
## [1] 0.1374053
```

```
plot(all_returns[,2], type='l')
```




```
print ("Look at Investment-grade corporate bonds (LQD)returns over time")
```

```
## [1] "Look at Investment-grade corporate bonds (LQD)returns over time"
```

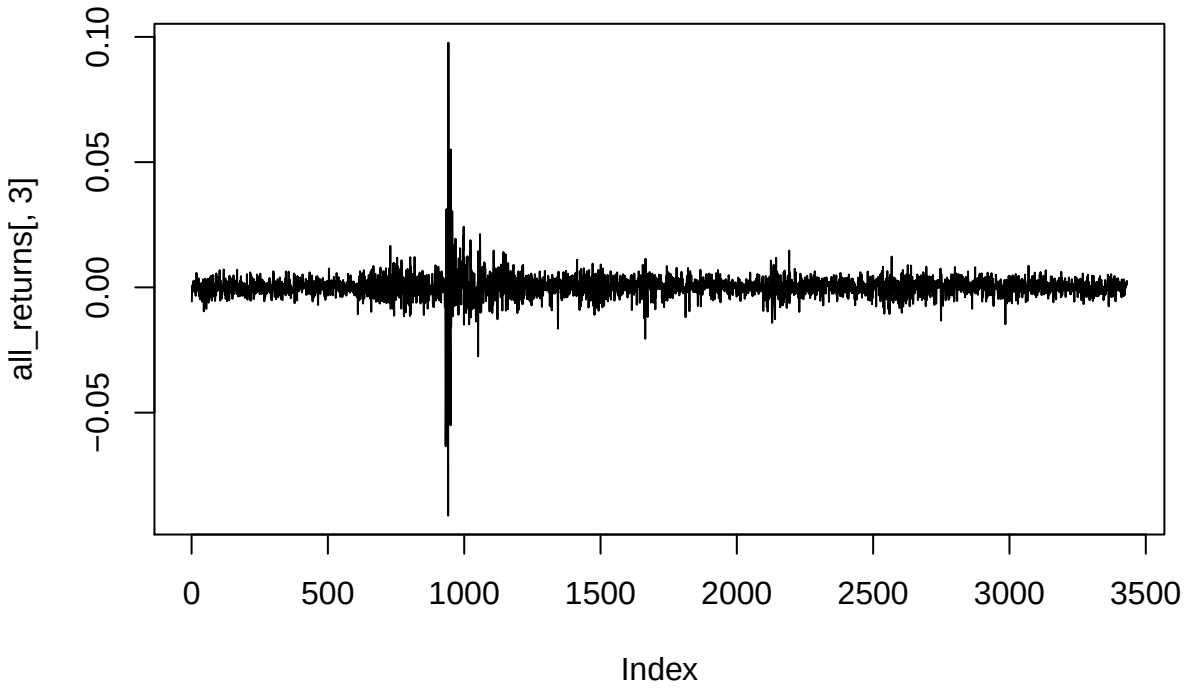
```
print ("Annual Standard Deviation: ")
```

```
## [1] "Annual Standard Deviation: "
```

```
sd(all_returns[,3])*sqrt(250)
```

```
## [1] 0.07792789
```

```
plot(all_returns[,3], type='l')
```



```
print ("Look at the Emerging-market equities (EEM) returns over time")
```

```
## [1] "Look at the Emerging-market equities (EEM) returns over time"
```

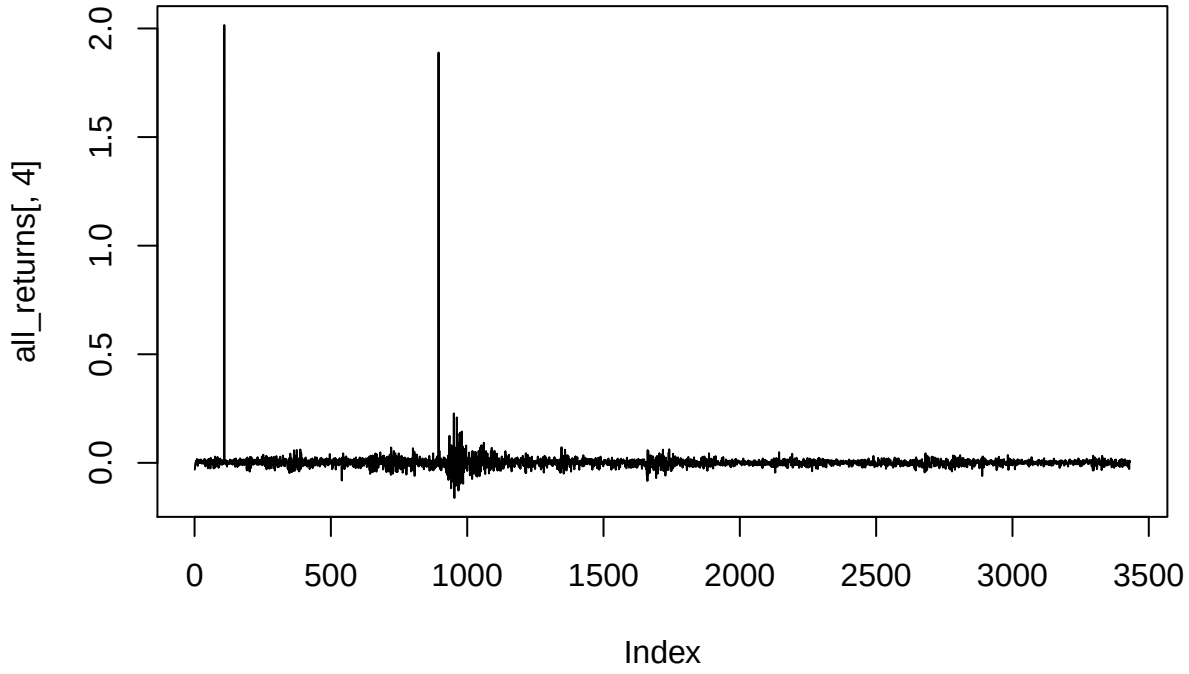
```
print ("Annual Standard Deviation: ")
```

```
## [1] "Annual Standard Deviation: "
```

```
sd(all_returns[,4])*sqrt(250)
```

```
## [1] 0.805501
```

```
plot(all_returns[,4], type='l')
```




```
print ("Look at the Real estate (VNQ) returns over time")
```

```
## [1] "Look at the\tReal estate (VNQ) returns over time"
```

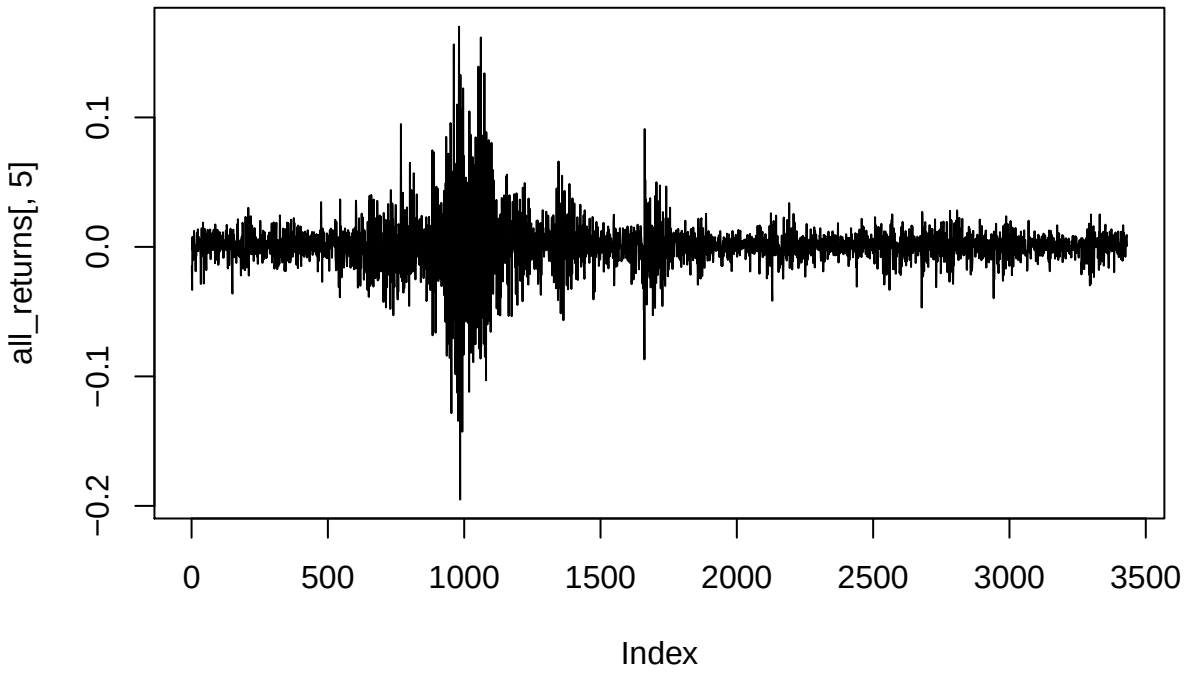
```
print ("Annual Standard Deviation: ")
```

```
## [1] "Annual Standard Deviation: "
```

```
sd(all_returns[,5])*sqrt(250)
```

```
## [1] 0.3140011
```

```
plot(all_returns[,5], type='l')
```




```
print ("The sample correlation matrix")
```

```
## [1] "The sample correlation matrix"
```

```
print (cor(all_returns))
```

```
##            ClCl.SPYa  ClCl.TLTa  ClCl.LQDa   ClCl.EEMa   ClCl.VNQa
## ClCl.SPYa  1.0000000 -0.4136226 0.10196844  0.31226649  0.76171688
## ClCl.TLTa -0.4136226  1.0000000 0.45256073 -0.12047197 -0.23560736
## ClCl.LQDa  0.1019684  0.4525607 1.00000000  0.06000322  0.07582542
## ClCl.EEMa  0.3122665 -0.1204720 0.06000322  1.00000000  0.21898826
## ClCl.VNQa  0.7617169 -0.2356074 0.07582542  0.21898826  1.00000000
```

```
print("Sample a random return from the empirical joint distribution")
```

```
## [1] "Sample a random return from the empirical joint distribution"
```

```
print("one day analysis - event weighting")
```

```
## [1] "one day analysis - event weighting"
```

```
#"This simulates a random day"
return.today = resample(all_returns, 1, orig.ids=FALSE)
print("random return today =")
```

```
## [1] "random return today ="
```

```
print(return.today)
```

```
##              ClCl.SPYa  ClCl.TLTa    ClCl.LQDa   ClCl.EEMa   ClCl.VNQa
## 2010-09-15 0.003817133 -0.0148348 -0.001614069 0.001387306 0.008680864
```

look at initial value of the weights The first portfolio contains equal weights

```
# Update the value of your holdings
# Assumes an equal allocation to each asset
total_wealth = 100000
```

```
my_weights = c(0.2,
               0.2,
               0.2,
               0.2,
               0.2)
holdings = total_wealth*my_weights
holdings = holdings*(1 + return.today)

# Compute your new total wealth
total_wealth = sum(holdings)
sum(holdings)
```

```
## [1] 99948.73
```

```
print("two trading week simulation")
```

```
## [1] "two trading week simulation"
```

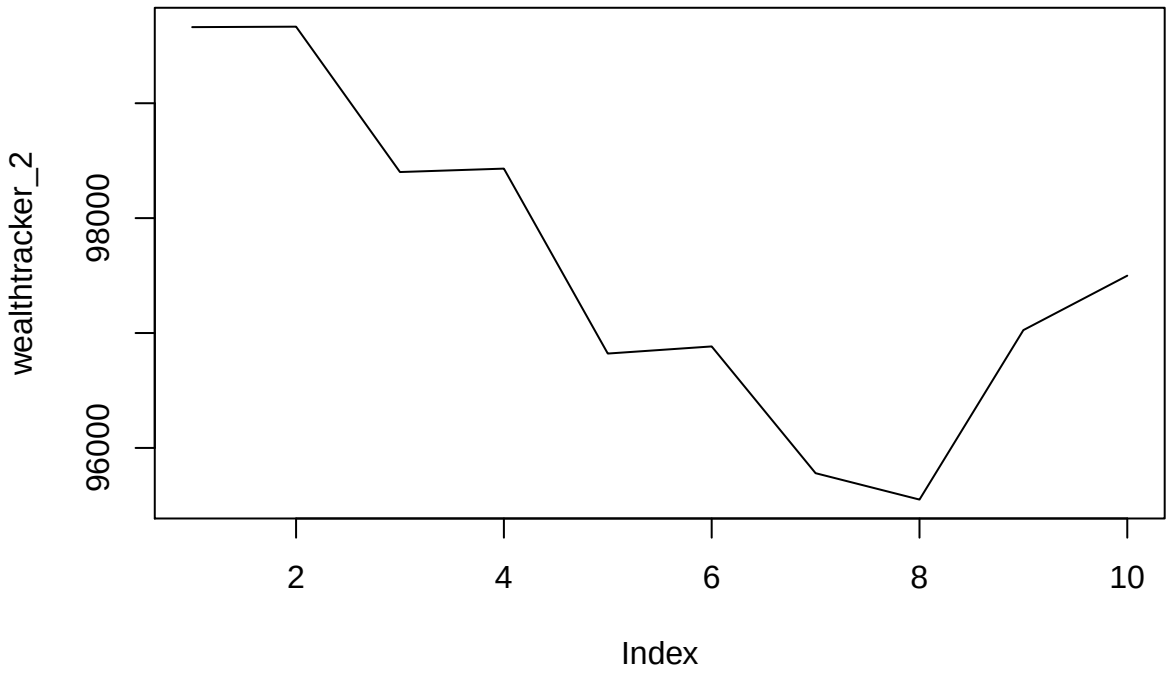
```
total_wealth_2 = 100000
weights_2 = c(0.2,
            0.2,
            0.2,
            0.2,
            0.2)
holdings_2 = weights_2 * total_wealth_2
n_days_2 = 10
wealthtracker_2 = rep(0, n_days_2) # Set up a placeholder to track total wealth
for(today in 1:n_days_2) {
  return.today = resample(all_returns, 1, orig.ids=FALSE)
  holdings_2 = holdings_2 + holdings_2*return.today
  total_wealth_2 = sum(holdings_2)
  wealthtracker_2[today] = total_wealth_2
}
print("Total Wealth")
```

```
## [1] "Total Wealth"
```

```
total_wealth_2
```

```
## [1] 97499.09
```

```
plot(wealthtracker_2, type='l')
```



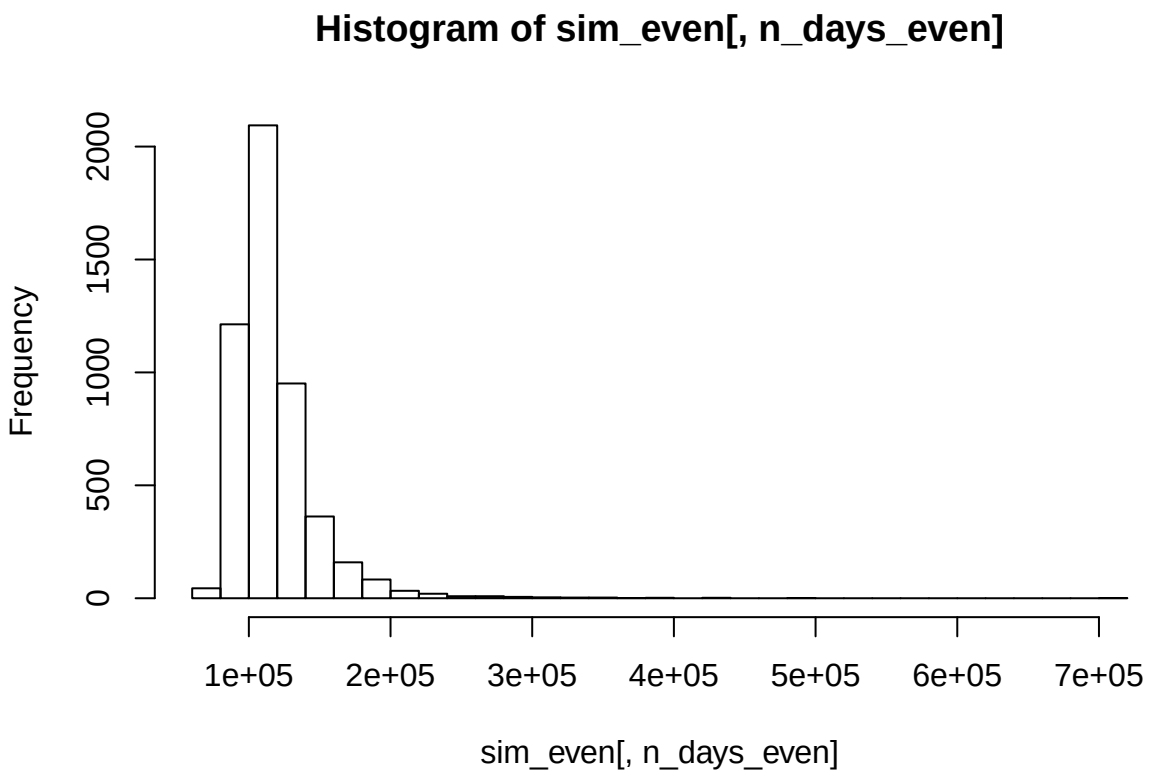

```
# Even Split Portfolio #
print ("simulate many different possible scenarios over a one year period")
```

```
## [1] "simulate many different possible scenarios over a one year period"
```

```
print ("Even Split Portfolio")
```

```
## [1] "Even Split Portfolio"
```

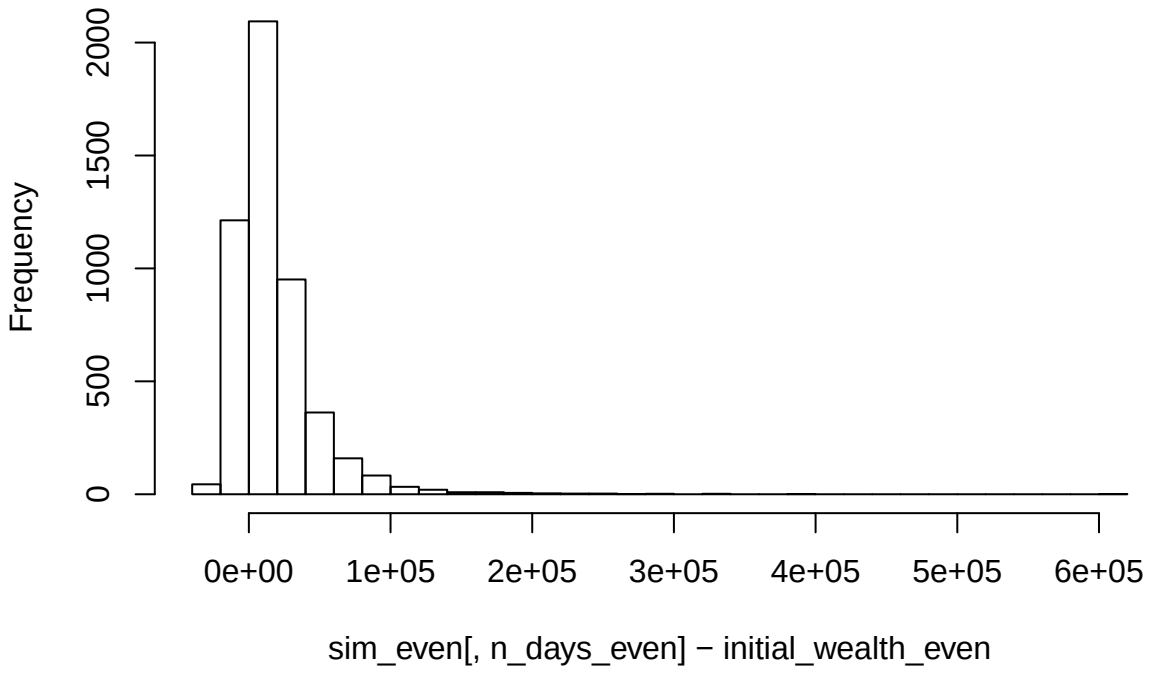
```
initial_wealth_even = 100000
sim_even = foreach(i=1:5000, .combine='rbind') %do% {
  total_wealth_even = initial_wealth_even
  weights_even = c(0.2,
                   0.2,
                   0.2,
                   0.2,
                   0.2)
  holdings_even = weights_even * total_wealth_even
  n_days_even = 250
  wealthtracker_even= rep(0, n_days_even)
  for(today in 1:n_days_even) {
    return.today = resample(all_returns, 1, orig.ids=FALSE)
    holdings_even = holdings_even + holdings_even*return.today
    total_wealth_even = sum(holdings_even)
    wealthtracker_even[today] = total_wealth_even
  }
  wealthtracker_even
}

#head(sim_even,2)
hist(sim_even[,n_days_even], 25)
```




```
 # Profit/loss #
mean(sim_even[,n_days_even])
```

```
## [1] 117449.8
```

```
hist(sim_even[,n_days_even] - initial_wealth_even, breaks=30)
```




```
print(" 5% value at risk for 20 days")
```

```
## [1] " 5% value at risk for 20 days"
```

```
quantile(sim_even[,20], 0.05) - initial_wealth_even
```

```
##        5%
## -5815.441
```

```
Var_even = initial_wealth_even - quantile(sim_even[,20], 0.05)
print(Var_even)
```

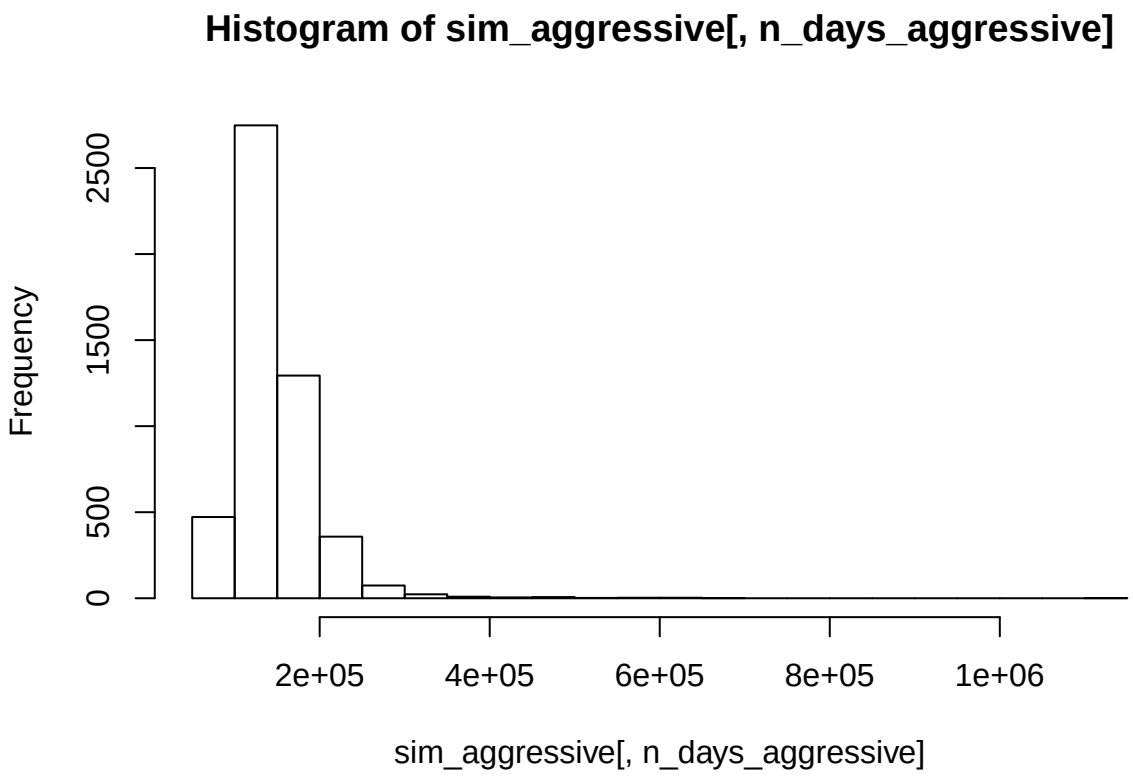
```
##       5%
## 5815.441
```

** Here we look at the aggressive portfolio, which is high in equities**

```
 # aggressive #
spy_w_a = .40
tlt_w_a = .05
lqd_w_a = .05
eem_w_a = .30
vnq_w_a = .4

print(cat(paste("Aggressive Split Portfolio",
         "US domestic equities (SPY: the S&P 500 stock index)", spy_w_a,
         "US Treasury bonds (TLT)",tlt_w_a,
        "Investment-grade corporate bonds (LQD)",lqd_w_a,
        "Emerging-market equities (EEM)",eem_w_a,
        "Real estate (VNQ)", vnq_w_a
          ,sep="\n")))
```

```
## Aggressive Split Portfolio
## US domestic equities (SPY: the S&P 500 stock index)
## 0.4
## US Treasury bonds (TLT)
## 0.05
## Investment-grade corporate bonds (LQD)
## 0.05
## Emerging-market equities (EEM)
## 0.3
## Real estate (VNQ)
## 0.4NULL
```

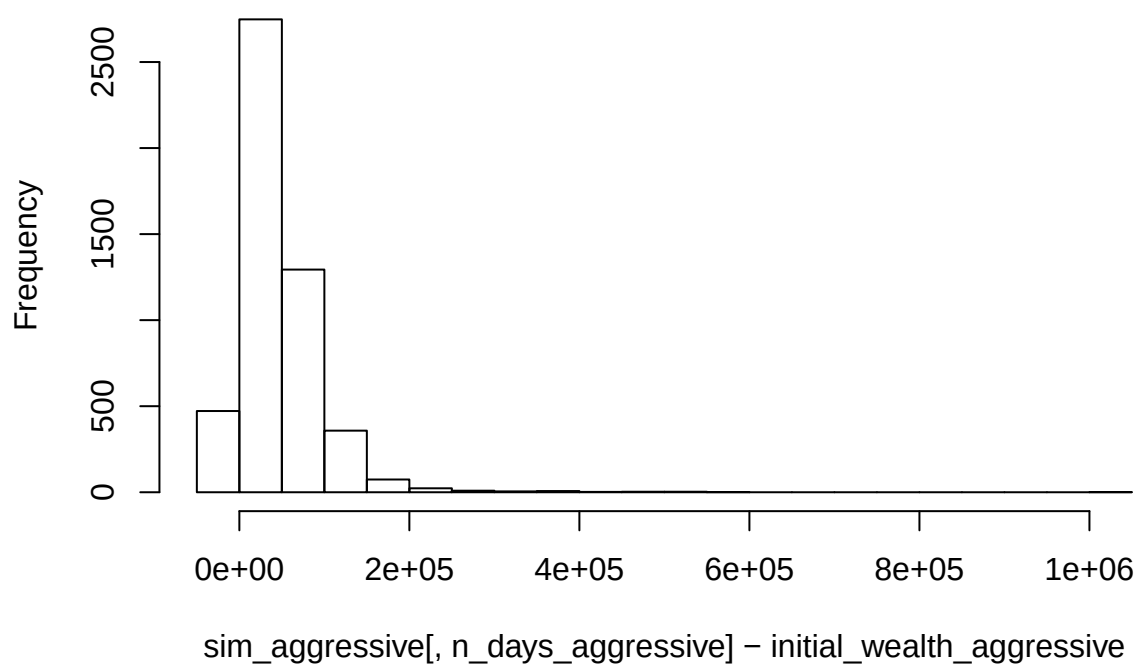
```
initial_wealth_aggressive = 100000
sim_aggressive = foreach(i=1:5000, .combine='rbind') %do% {
  total_wealth_aggressive = initial_wealth_aggressive
  weights_aggressive = c(spy_w_a,
                         tlt_w_a,
                         lqd_w_a,
                         eem_w_a,
                         vnq_w_a)
  holdings_aggressive = weights_aggressive * total_wealth_aggressive
  n_days_aggressive = 250
  wealthtracker_aggressive= rep(0, n_days_aggressive)
  for(today in 1:n_days_aggressive) {
    return.today = resample(all_returns, 1, orig.ids=FALSE)
    holdings_aggressive = holdings_aggressive + holdings_aggressive*return.today
    total_wealth_aggressive = sum(holdings_aggressive)
    wealthtracker_aggressive[today] = total_wealth_aggressive
  }
  wealthtracker_aggressive
}

#head(sim_aggressive,2)
hist(sim_aggressive[,n_days_aggressive], 25)
```




```
 # Profit/loss  #
mean(sim_aggressive[,n_days_aggressive])
```

```
## [1] 145421
```

```
hist(sim_aggressive[,n_days_aggressive] - initial_wealth_aggressive, breaks=30)
```




```
print(" 5% value at risk for 20 days")
```

```
## [1] " 5% value at risk for 20 days"
```

```
quantile(sim_aggressive[,20], 0.05) - initial_wealth_aggressive
```

```
##       5%
## 9327.473
```

```
Var_aggressive = initial_wealth_aggressive - quantile(sim_aggressive[,20], 0.05)
print(Var_aggressive)
```

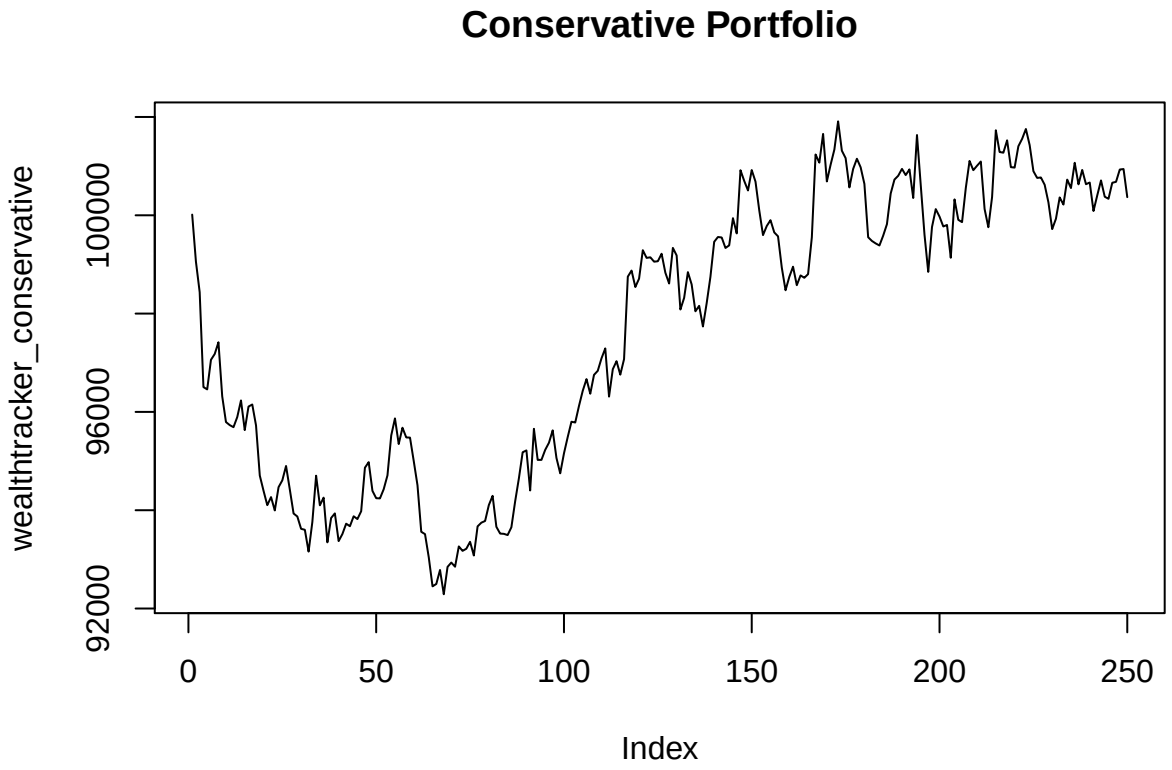
```
##        5%
## -9327.473
```

** Here we look at the aggressive portfolio, which is high in fixed income**

```
 # conservative #
spy_w_c = .05
tlt_w_c = .40
lqd_w_c = .40
eem_w_c = .05
vnq_w_c = .10

print(cat(paste("Conservative Split Portfolio",
                "US domestic equities (SPY: the S&P 500 stock index)", spy_w_c,
                "US Treasury bonds (TLT)",tlt_w_c,
                "Investment-grade corporate bonds (LQD)",lqd_w_c,
                "Emerging-market equities (EEM)",eem_w_c,
                "Real estate (VNQ)", vnq_w_c
                ,sep="\n")))
```

```
## Conservative Split Portfolio
## US domestic equities (SPY: the S&P 500 stock index)
## 0.05
## US Treasury bonds (TLT)
## 0.4
## Investment-grade corporate bonds (LQD)
## 0.4
## Emerging-market equities (EEM)
## 0.05
## Real estate (VNQ)
## 0.1NULL
```

```
initial_wealth_conservative = 100000
sim_conservative = foreach(i=1:5000, .combine='rbind') %do% {
  total_wealth_conservative = initial_wealth_conservative
  weights_conservative = c(spy_w_c,
                           tlt_w_c,
                           lqd_w_c,
                           eem_w_c,
                           vnq_w_c)
  holdings_conservative = weights_conservative * total_wealth_conservative
  n_days_conservative = 250
  wealthtracker_conservative= rep(0, n_days_conservative)
  for(today in 1:n_days_conservative) {
    return.today = resample(all_returns, 1, orig.ids=FALSE)
    holdings_conservative = holdings_conservative + holdings_conservative*return.today
    total_wealth_conservative = sum(holdings_conservative)
    wealthtracker_conservative[today] = total_wealth_conservative
  }

}

#head(sim_conservative,2)
#hist(sim_conservative[,n_days_conservative], 25)
plot(wealthtracker_conservative,type = "l",  main = "Conservative Portfolio")
```




```
 # Profit/loss  #
mean(sim_conservative[,n_days_conservative])
```

```
## Warning in mean.default(x, ..., na.rm = na.rm): argument is not numeric or
## logical: returning NA
```

```
## [1] NA
```

```
#hist(sim_conservative[,n_days_conservative] - initial_wealth_conservative, breaks=30)

print(" 5% value at risk for 20 days")
```

```
## [1] " 5% value at risk for 20 days"
```

```
quantile(sim_conservative[,20], 0.05) - initial_wealth_conservative
```

```
## 5%
## NA
```

```
Var_conservative = initial_wealth_conservative - quantile(sim_conservative[,20], 0.05)
print(Var_conservative)
```

```
## 5%
## NA
```

```
print("# •marshals appropriate evidence to characterize the risk/return properties of the five major ass
```

```
## [1] "# •marshals appropriate evidence to characterize the risk/return properties of the five major as
```

```
print('# •  outlines your choice of the ""safe"" and ""aggressive"" portfolios.')
```

```
## [1] "# •\toutlines your choice of the \"\"safe\"\" and \"\"aggressive\"\" portfolios."
```

```
print(cat(paste("Aggressive Split Portfolio",
                "US domestic equities (SPY: the S&P 500 stock index)", spy_w_a,
                "US Treasury bonds (TLT)",tlt_w_a,
                "Investment-grade corporate bonds (LQD)",lqd_w_a,
                "Emerging-market equities (EEM)",eem_w_a,
                "Real estate (VNQ)", vnq_w_a
                ,sep="\n")))
```

```
## Aggressive Split Portfolio
## US domestic equities (SPY: the S&P 500 stock index)
## 0.4
## US Treasury bonds (TLT)
## 0.05
## Investment-grade corporate bonds (LQD)
## 0.05
## Emerging-market equities (EEM)
## 0.3
## Real estate (VNQ)
## 0.4NULL
```

```
print("The aggressive portfolio is heavy biased towards equities. Equity biased portfolios have a highe
```

```
## [1] "The aggressive portfolio is heavy biased towards equities. Equity biased portfolios have a high
```

```
print(cat(paste("Conservative Split Portfolio",
                "US domestic equities (SPY: the S&P 500 stock index)", spy_w_c,
                "US Treasury bonds (TLT)",tlt_w_c,
                "Investment-grade corporate bonds (LQD)",lqd_w_c,
                "Emerging-market equities (EEM)",eem_w_c,
                "Real estate (VNQ)", vnq_w_c
                ,sep="\n")))
```

```
## Conservative Split Portfolio
## US domestic equities (SPY: the S&P 500 stock index)
## 0.05
## US Treasury bonds (TLT)
## 0.4
## Investment-grade corporate bonds (LQD)
## 0.4
## Emerging-market equities (EEM)
## 0.05
## Real estate (VNQ)
## 0.1NULL
```

```
print("The more conservative portfolio is primarily invested in treasuries (considered 'risk free') and
```

```
## [1] "The more conservative portfolio is primarily invested in treasuries (considered 'risk free') an
```

```
print("use bootstrap resampling to estimate the 4-week (20 trading day) value at risk of each of your t
```

```
## [1] "use bootstrap resampling to estimate the 4-week (20 trading day) value at risk of each of your
```

```
print("Aggressive")
```

```
## [1] "Aggressive"
```

```
print(Var_aggressive)
```

```
##        5%
## -9327.473
```

```
print("Even Weight")
```

```
## [1] "Even Weight"
```

```
print(Var_even)
```

```
##       5%
## 5815.441
```

```
print("Conservative")
```

```
## [1] "Conservative"
```

```
print(Var_conservative)
```

```
## 5%
## NA
```

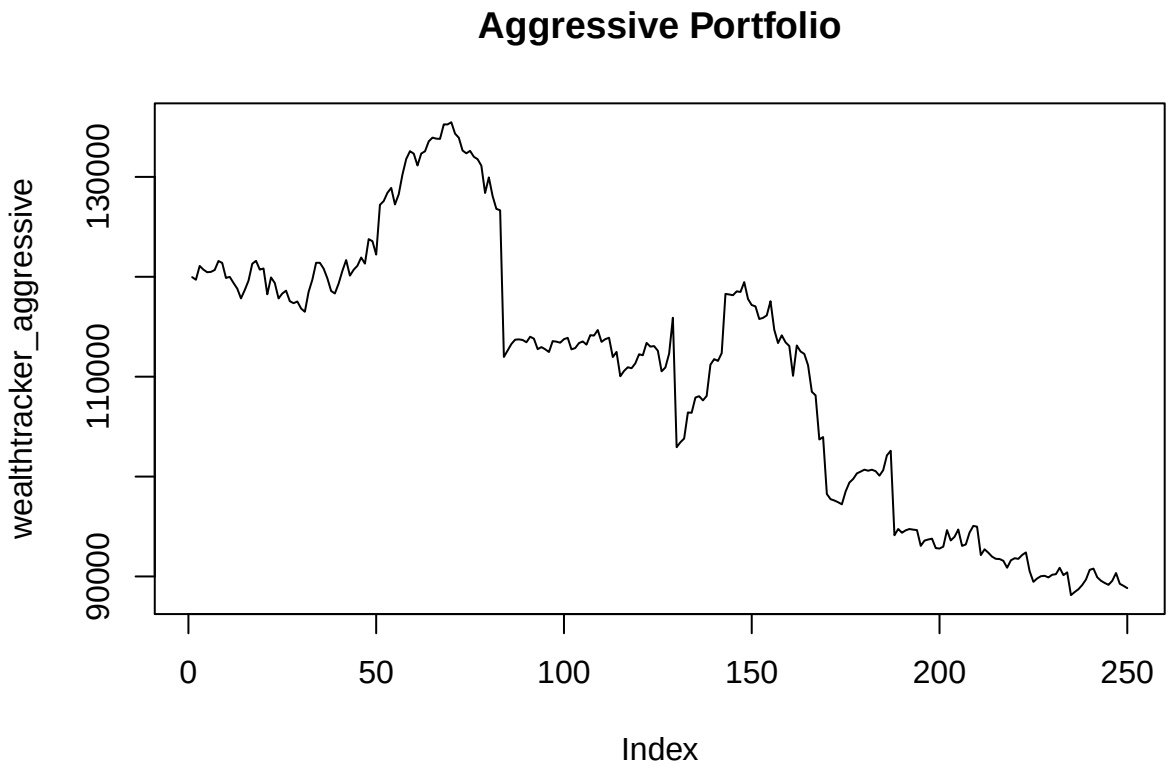
```
print("compares the results for each portfolio in a way that would allow the reader to make an intellig
```

```
## [1] "compares the results for each portfolio in a way that would allow the reader to make an intelli
```

```
print("Aggressive Portfolio")
```

```
## [1] "Aggressive Portfolio"
```

```
plot(wealthtracker_aggressive,type = "l",  main = "Aggressive Portfolio")
```




```
cat("One year total estimate wealth ",tail(wealthtracker_aggressive,n=1))
```

```
## One year total estimate wealth  88832.81
```

```
cat("20 day VAR at 95% confidence", Var_aggressive)
```

```
## 20 day VAR at 95% confidence -9327.473
```

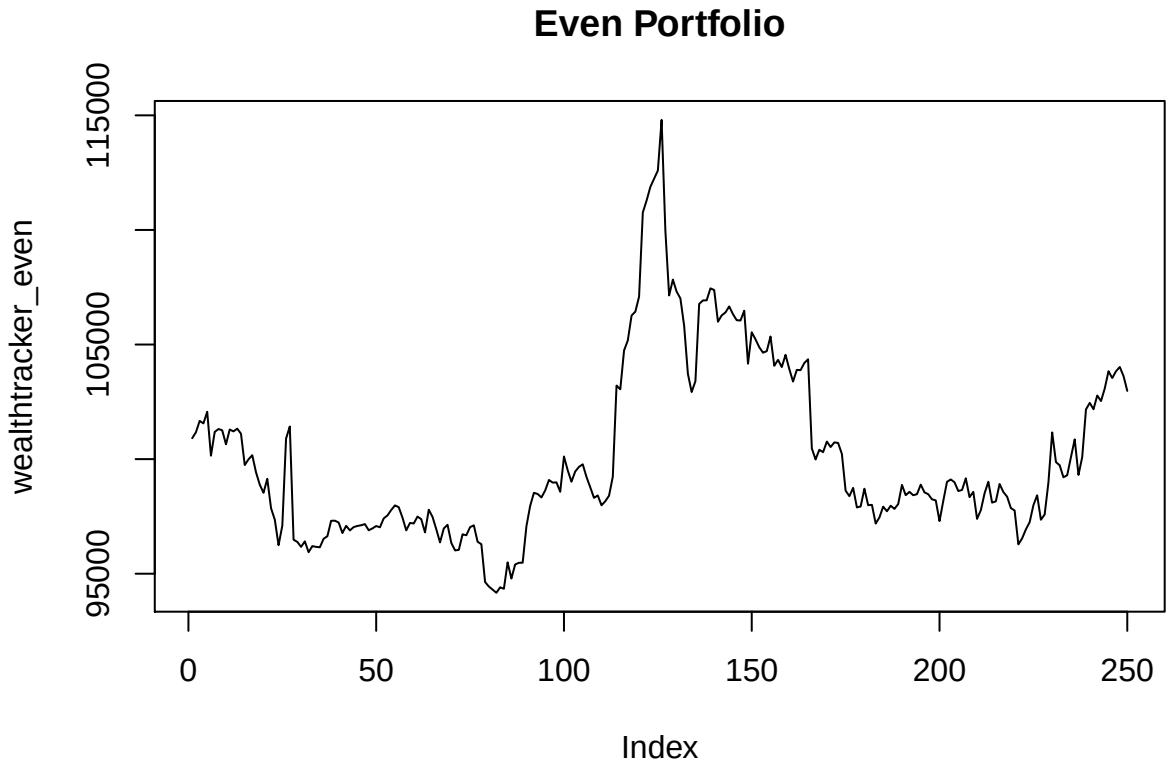
```
cat("Projected One Year Performance: ",((tail(wealthtracker_aggressive,n=1))-100000)/100000*100,"%")
```

```
## Projected One Year Performance:  -11.16719 %
```

```
print("Even Portfolio")
```

```
## [1] "Even Portfolio"
```

```
plot(wealthtracker_even,type = "l",  main = "Even Portfolio")
```




```
cat("One year total estimate wealth ",tail(wealthtracker_even,n=1))
```

```
## One year total estimate wealth  102977.3
```

```
cat("20 day VAR at 95% confidence", Var_even)
```

```
## 20 day VAR at 95% confidence 5815.441
```

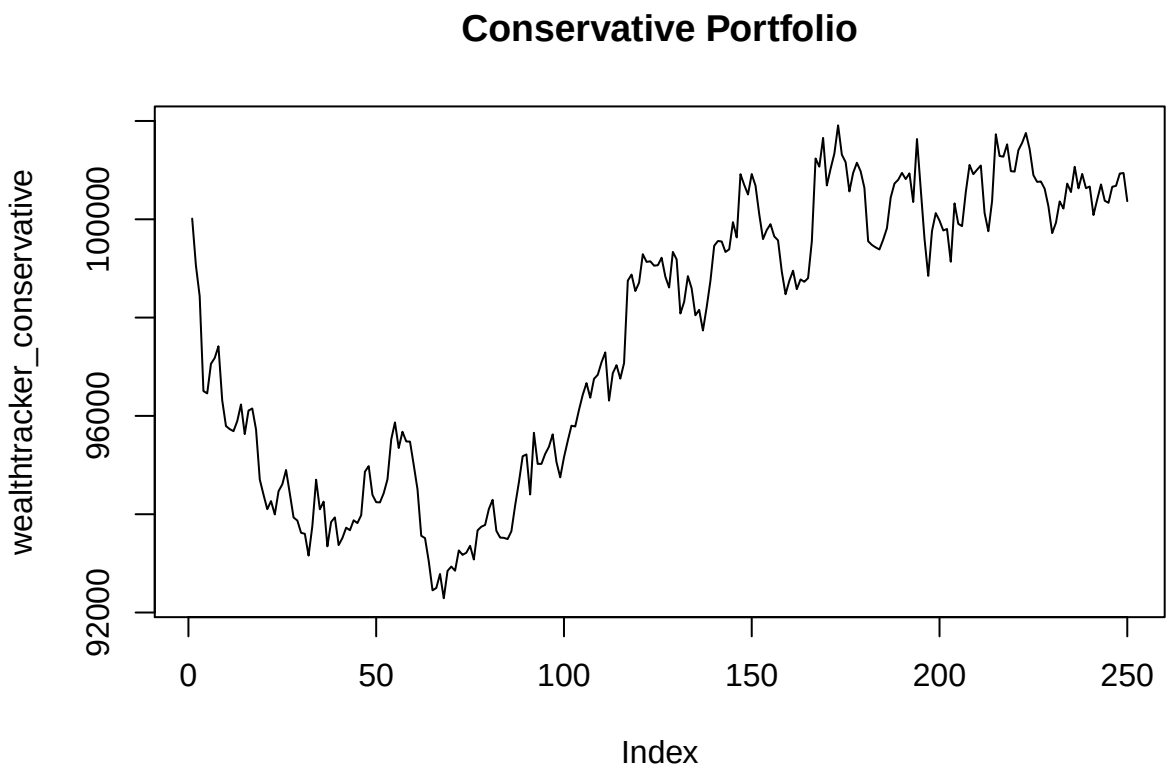
```
cat("Projected One Year Performance: ",((tail(wealthtracker_even,n=1))-100000)/100000*100,"%")
```

```
## Projected One Year Performance:  2.97733 %
```

```
print("Conservative Portfolio")
```

```
## [1] "Conservative Portfolio"
```

```
plot(wealthtracker_conservative,type = "l",  main = "Conservative Portfolio")
```




```
cat("One year total estimate wealth ",tail(wealthtracker_conservative,n=1))
```

```
## One year total estimate wealth  100368.8
```

```
cat("20 day VAR at 95% confidence", Var_conservative)
```

```
## 20 day VAR at 95% confidence NA
```

```
cat("Projected One Year Performance: ",((tail(wealthtracker_conservative,n=1))-100000)/100000*100,"%")
```

```
## Projected One Year Performance:  0.3688495 %
```

Market segmentation

```
#7882 rows and 37 variables
#cut porn 'bots'
#501 with more than one adult.
#426 with 2 or more
#cut 2 or more in case of error
social_clean1 = social[social["adult"]<3,]
#dim(social_clean1)



#guidance https://github.com/jgscott/STA380/blob/master/R/cars.R
```

Data Cleaning I removed the porn bots by considering anything with 2 or more in this category. I also removed spam bots, but the porn bots had the bigger impact. Now we have 7,456 observations after previously having 7,882. There are 37 variables in the file.

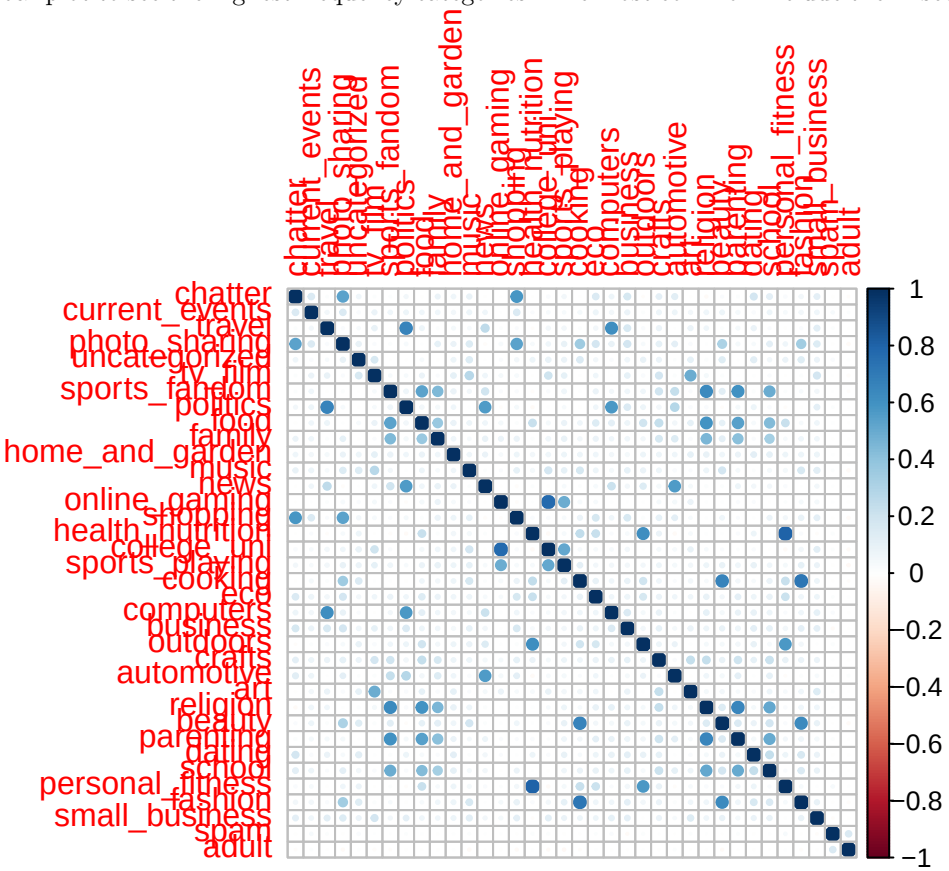
```r
#cut spam 'bots'
#cut 1 or more in case of error
social_clean2 = social_clean1[social_clean1["spam"]<2,]
# dim(social_clean2)
# social_clean2
#summary(social_clean2)
social_clean3 = social_clean2

str(social_clean3)
```

```
## 'data.frame':    7456 obs. of  37 variables:
##  $ X                : Factor w/ 7882 levels "123pxkyqj","12grikctu",..: 3720 2540 4096 596 3197 3609 4
##  $ chatter          : int  2 3 6 1 5 6 1 5 6 5 ...
##  $ current_events   : int  0 3 3 5 2 4 2 3 2 2 ...
##  $ travel           : int  2 2 4 2 0 2 7 3 0 4 ...
##  $ photo_sharing    : int  2 1 3 2 6 7 1 6 1 4 ...
##  $ uncategorized    : int  2 1 1 0 1 0 0 1 0 0 ...
##  $ tv_film          : int  1 1 5 1 0 1 1 1 0 5 ...
##  $ sports_fandom    : int  1 4 0 0 0 1 1 1 0 9 ...
##  $ politics         : int  0 1 2 1 2 0 11 0 0 1 ...
##  $ food             : int  4 2 1 0 0 2 1 0 2 5 ...
##  $ family           : int  1 2 1 1 1 1 0 0 2 4 ...
##  $ home_and_garden  : int  2 1 1 0 0 1 0 0 1 0 ...
##  $ music            : int  0 0 1 0 0 1 0 2 1 1 ...
##  $ news             : int  0 0 1 0 0 0 1 0 0 0 ...
##  $ online_gaming    : int  0 0 0 0 3 0 0 1 2 1 ...
##  $ shopping         : int  1 0 2 0 2 5 1 3 0 0 ...
##  $ health_nutrition: int  17 0 0 0 0 0 1 1 22 7 ...
##  $ college_uni      : int  0 0 0 1 4 0 1 0 1 4 ...
##  $ sports_playing   : int  2 1 0 0 0 0 1 0 0 1 ...
##  $ cooking          : int  5 0 2 0 1 0 1 10 5 4 ...
##  $ eco              : int  1 0 1 0 0 0 0 0 2 1 ...
##  $ computers        : int  1 0 0 0 1 1 1 1 1 2 ...
##  $ business         : int  0 1 0 1 0 1 3 0 1 0 ...
##  $ outdoors         : int  2 0 0 0 1 0 1 0 3 0 ...
##  $ crafts           : int  1 2 2 3 0 0 0 1 0 0 ...
##  $ automotive       : int  0 0 0 0 0 1 0 1 0 4 ...
##  $ art              : int  0 0 8 2 0 0 1 0 1 0 ...
##  $ religion         : int  1 0 0 0 0 0 1 0 0 13 ...
##  $ beauty           : int  0 0 1 1 0 0 0 5 5 1 ...
##  $ parenting        : int  1 0 0 0 0 0 0 1 0 3 ...
##  $ dating           : int  1 1 1 0 0 0 0 0 0 0 ...
##  $ school           : int  0 4 0 0 0 0 0 0 1 3 ...
##  $ personal_fitness: int  11 0 0 0 0 0 0 0 12 2 ...
##  $ fashion          : int  0 0 1 0 0 0 0 4 3 1 ...
##  $ small_business   : int  0 0 0 0 1 0 0 0 1 0 ...
##  $ spam             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ adult            : int  0 0 0 0 0 0 0 0 0 0 ...
```

```r
#colnames(social)

#count total amount in each column so we can plot
sum_columns = data.frame(value_columns = apply(social_clean3[,-1],2,sum))
sum_columns$key = rownames(sum_columns)
```

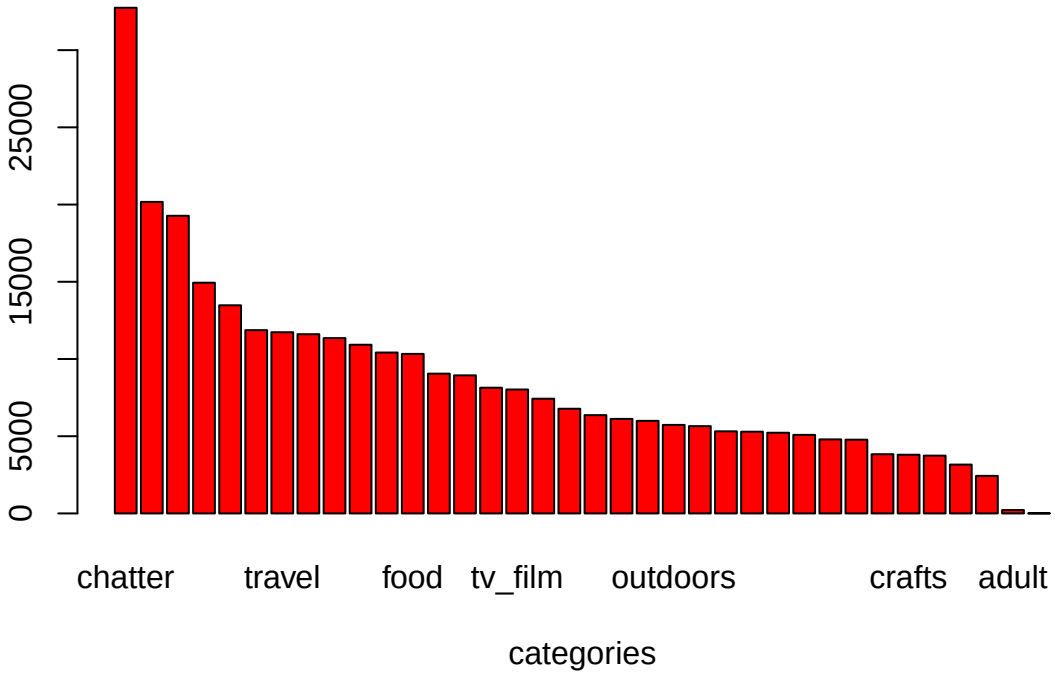
```
sum_columns2 = sum_columns[order(-(sum_columns$value)),]
```

Data Exploration Simple correlation matrix to see if there are any obvious relationships to dig into. Simple Top 10 Table and bar plot to see the highest frequency categories. The most common include the miscellaneous



chatter category.

```
##                   value_columns             key
## chatter                   32743         chatter
## photo_sharing             20171    photo_sharing
## health_nutrition          19272 health_nutrition
## cooking                   14940         cooking
## politics                  13477        politics
## sports_fandom             11867   sports_fandom
## travel                    11733          travel
## college_uni               11605     college_uni
## current_events            11354  current_events
## personal_fitness          10922 personal_fitness
```

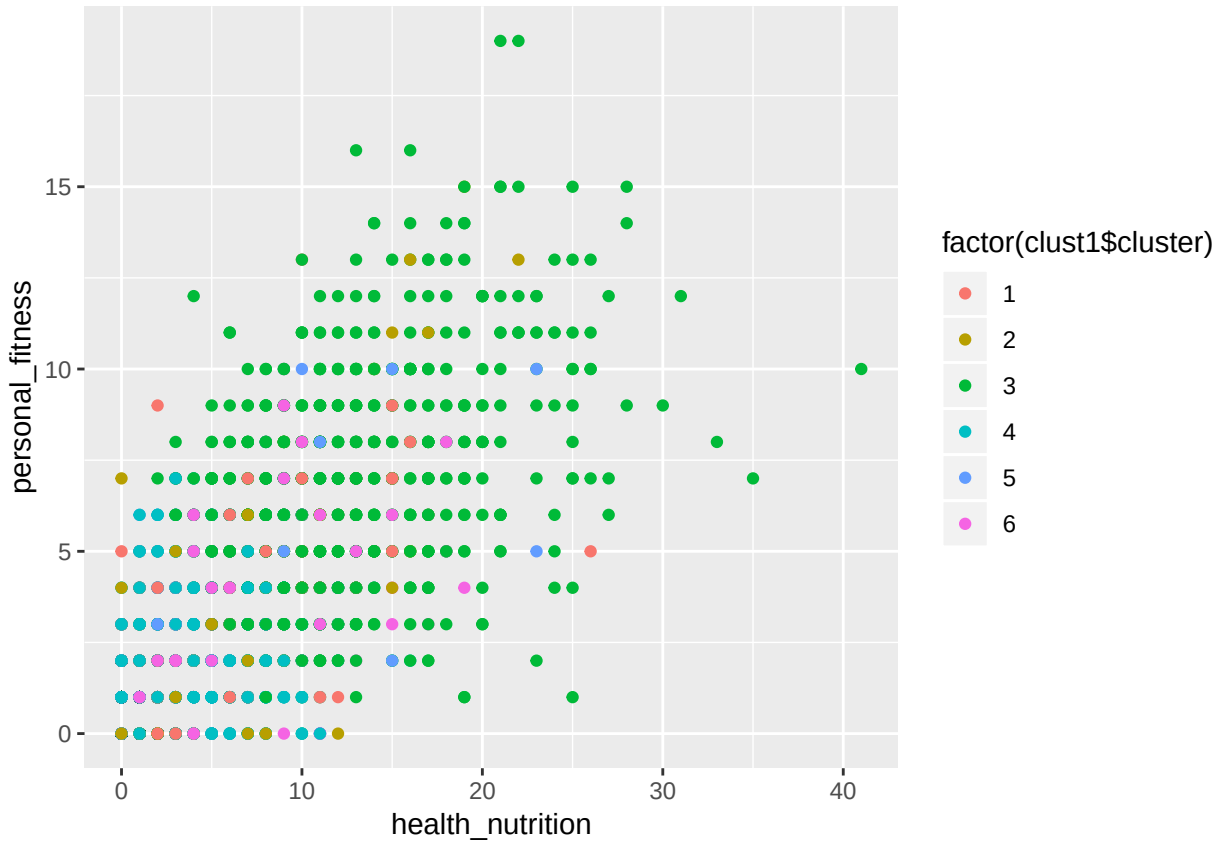


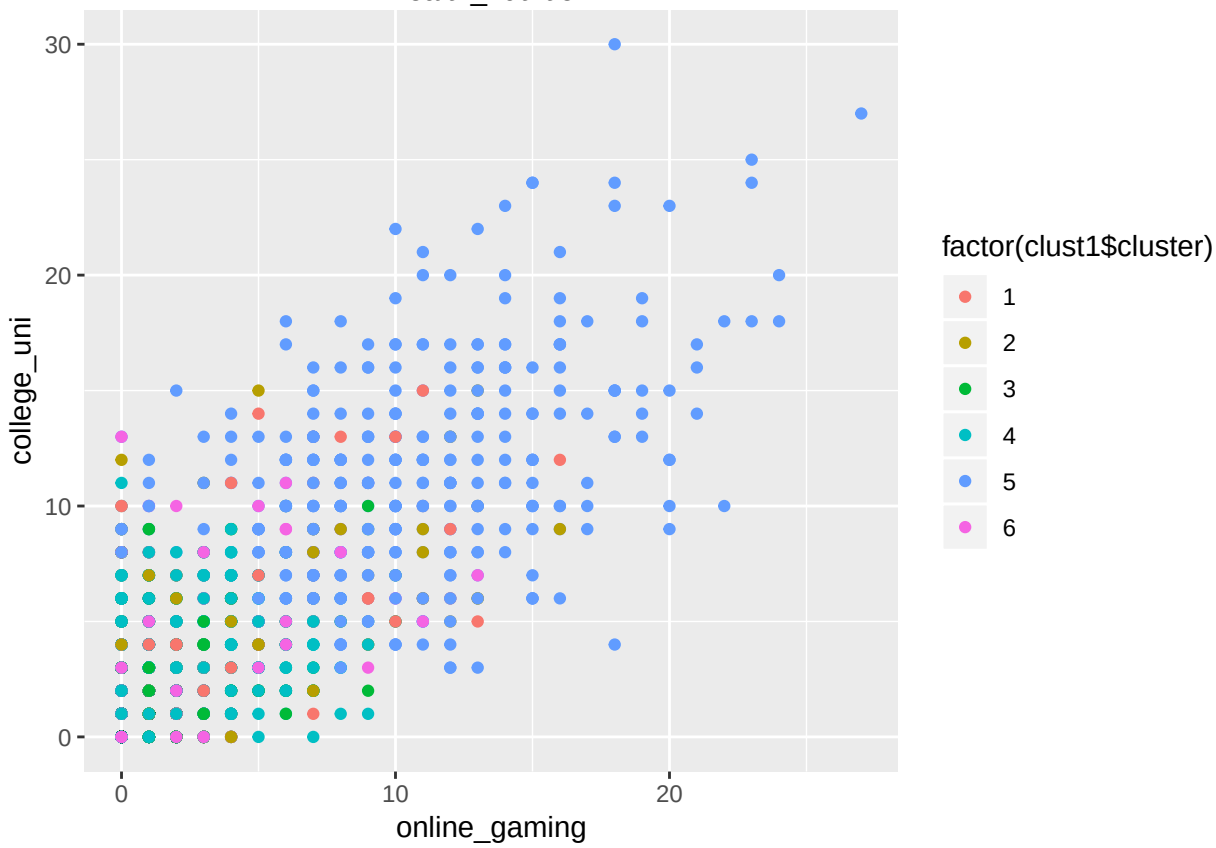

```
# Center and scale the data
X = social_clean3[,-(1:2)]
X = scale(X, center=TRUE, scale=TRUE)

# Extract the centers and scales from the rescaled data (which are named attributes)
mu = attr(X,"scaled:center")
sigma = attr(X,"scaled:scale")

# Run k-means with 6 clusters and 25 starts
clust1 = kmeans(X, 6, nstart=25)
```

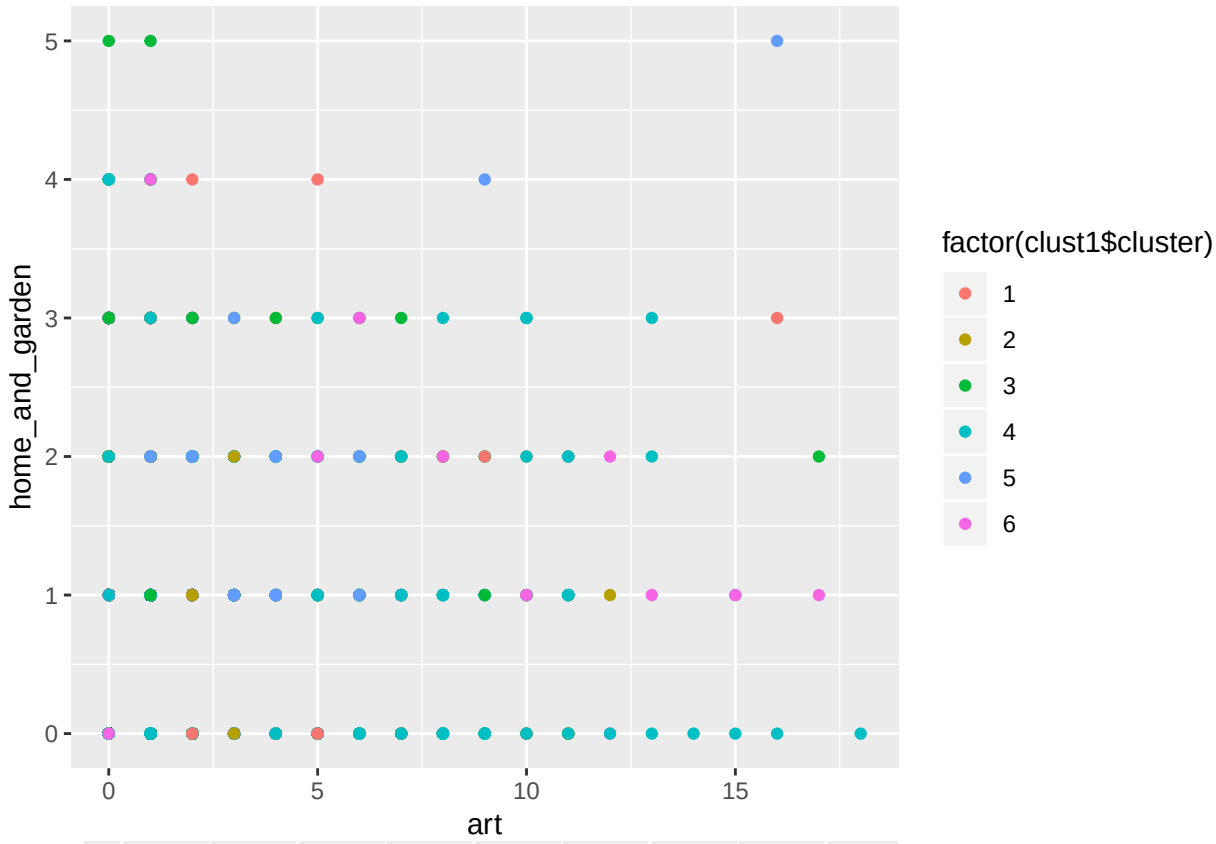
Cluster 1 = health and personal fitness Cluster 2 = college and online gaminng Cluster 3 = too mixed Cluster 4 = photo sharing Cluster 5 = Politics and travel Cluster 6 = too mixed

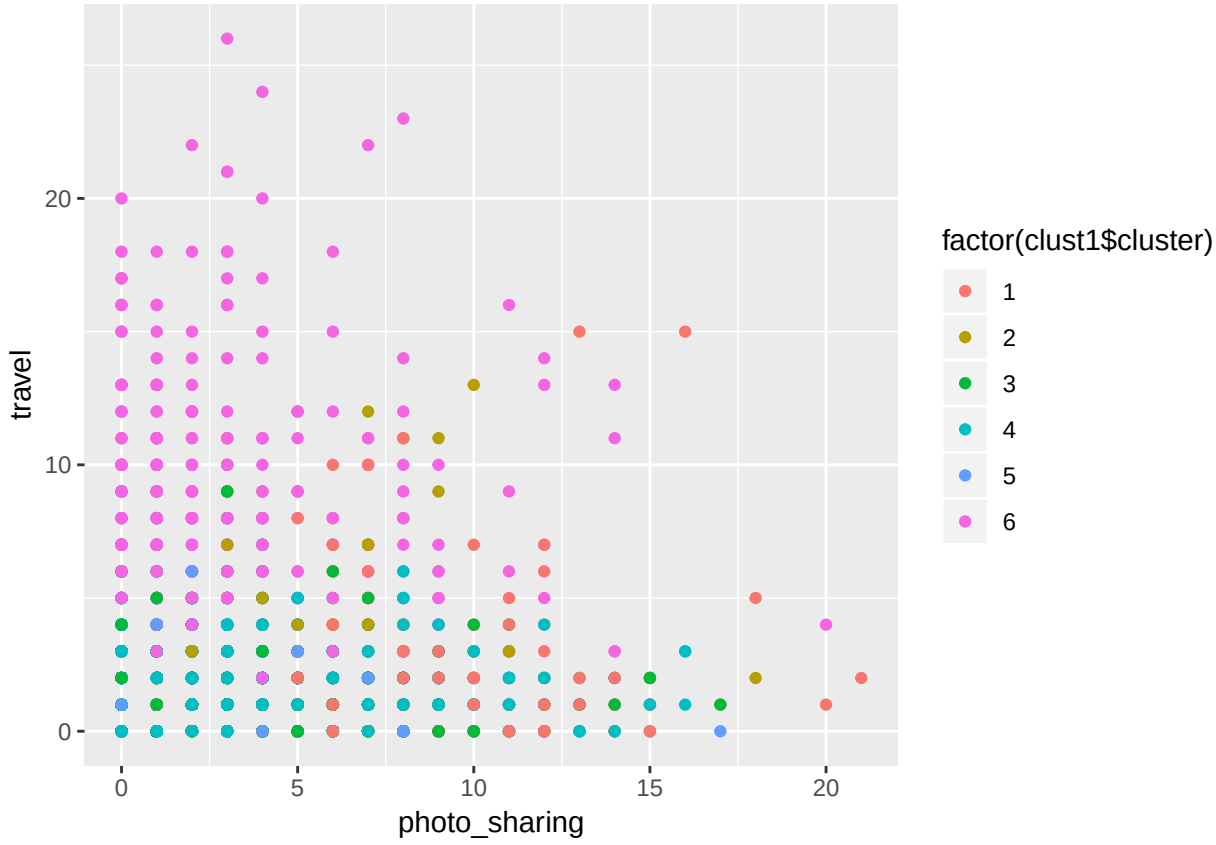
```
# What are the clusters?
```

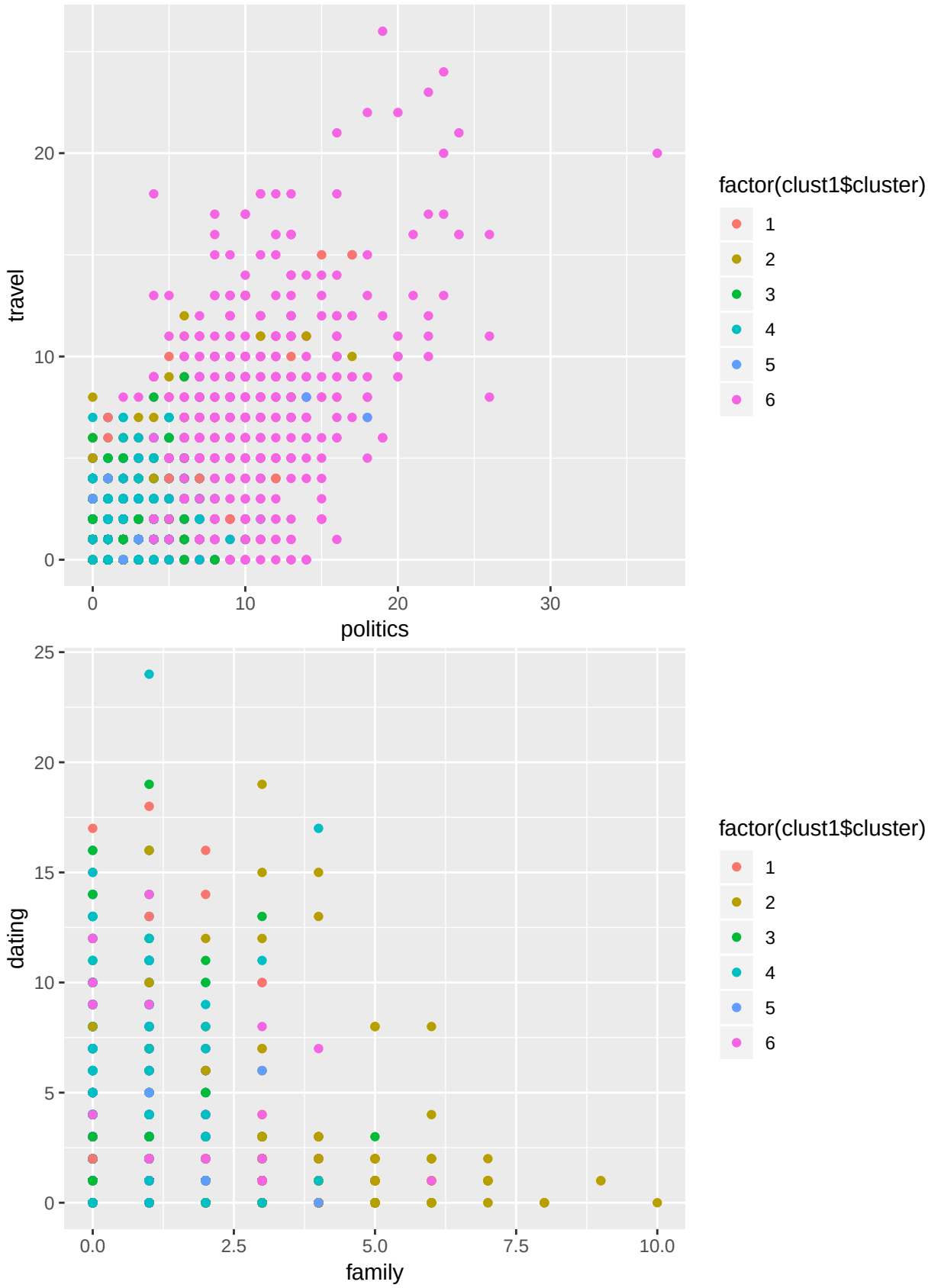
```
# Which comments are in which clusters?
#which(clust1$cluster == 1)
#which(clust1$cluster == 2)
#which(clust1$cluster == 3)
#which(clust1$cluster == 4)
#which(clust1$cluster == 5)
```

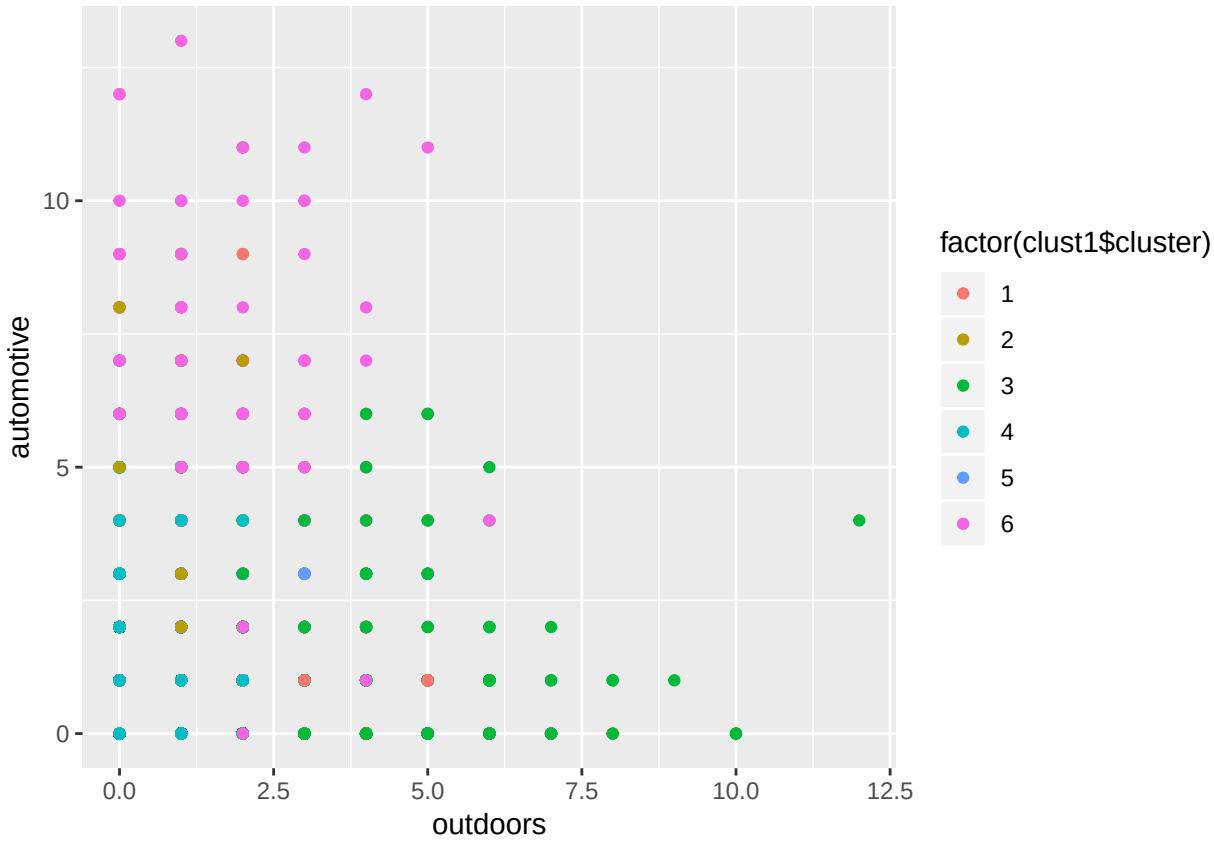

personal_fitness
health_nutrition
0
5
10
15
0
10
20
30
40
factor(clust1$cluster)
1
2
3
4
5
6


college_uni
online_gaming
0
10
20
30
0
10
20
factor(clust1$cluster)
1
2
3
4
5
6


home_and_garden
art
factor(clust1$cluster)
1
2
3
4
5
6


travel
photo_sharing
factor(clust1$cluster)
1
2
3
4
5
6


travel
politics
0
10
20
30
0
10
20
factor(clust1$cluster)
1
2
3
4
5
6
dating
family
0.0
2.5
5.0
7.5
10.0
0
5
10
15
20
25
factor(clust1$cluster)
1
2
3
4
5
6



**

k means **

```
# Using kmeans++ initialization
clust2 = kmeanspp(X, k=6, nstart=25)
```

```
# Which comments are in which clusters?
#which(clust2$cluster == 1)
#which(clust2$cluster == 2)
#which(clust2$cluster == 3)

# Compare versus within-cluster average distances from the first run
clust1$withinss
```

```
## [1] 20077.01 27258.08 27567.72 77776.11 14475.05 27815.17
```

```
clust2$withinss
```

```
## [1] 77776.11 14475.05 27815.17 27567.72 27258.08 20077.01
```

```
sum(clust1$withinss)
```

```
## [1] 194969.1
```

```
sum(clust2$withinss)
```

```
## [1] 194969.1
```

```
clust1$tot.withinss
```

```
## [1] 194969.1
```

```
clust2$tot.withinss
```

```
## [1] 194969.1
```

```
clust1$betweenss
```

```
## [1] 65955.87
```

```
clust2$betweenss
```

```
## [1] 65955.87
```

Below are the groupings I would have assumed without doing a statistical analysis. I wanted to force them into new columsn and see what the data looked like afterwards to see if this would manufacture any insights.

Arts Grouping: craft, photo_sharing, fashion, art, crafts, beauty, tv_film, music Hobbies: sports_fandom, sports_playing, online_gaming, computers, automotive, shopping Home: cooking, food, parenting, home_and_garden, travel, outdoors, dating, family, religion Health: health_nutrition, personal_fitness Business and Education: politics, small_business, eco, college_uni, current_events, news, business, school Random: chatter, spam, adult, uncategorized