STA 380, Part 2: Exercises 1

Mark Babbe, Camryn Callaway, Siqi Chen, Jiahao Ye, Zhiyi Yang

Aug 11, 2017

Contents


```
set.seed(1)
library(pander)
```

Probability practice

Part A.

First, we have the prior distribution,

$$P(RC) = 0.3$$

$$P(TC) = 1 - P(RC) = 0.7$$

and conditional distribution on random clicker,

$$P(Yes|RC) = 0.5$$

Now, given in a trial period,

$$P(Yes) = 0.65$$
$$P(No) = 0.35$$

We know,

$$P(Yes \cap RC) = P(Yes|RC) \times P(RC) = 0.5 \times 0.3 = 0.15$$

Then,

$$P(Yes \cap TC) = P(Yes) - P(Yes \cap RC) = 0.65 - 0.15 = 0.5$$

Hence,

$$P(Yes|TC) = \frac{P(Yes \cap TC)}{P(TC)} = \frac{0.5}{0.7} = 0.7143$$

Therefore, **71.43%** of the truthful clickers answered yes.

Part B.

First, we have the prior distribution,

$$P(Disease) = 0.000025$$
$$P(Non - Disease) = 1 - P(Disease) = 0.999975$$

Also, we have the conditional distribution on disease,

$$P(Positive|Disease) = 0.993 \quad P(Negative|Disease) = 0.007$$
$$P(Positive|Non - Disease) = 0.0001 \quad P(Negative|Non - Disease) = 0.9999$$

Or, in a contingency table

```
Disease = c(0.993,0.007)
Healthy = c(0.0001,0.9999)
table = cbind(Disease,Healthy)
df = data.frame(table, row.names = c('Positive','Negative'))
pander(df)
```

	Disease	Healthy
Positive	0.993	1e-04
Negative	0.007	0.9999

We can then calculate the marginal distribution of test result,

$$\begin{aligned} P(Positive) &= P(Positive|Disease) \times P(Disease) + P(Positive|Non - Disease) \times P(Non - Disease) \\ &= 0.993 \times 0.000025 + 0.0001 \times 0.999975 \\ &= 0.0001248225 \\ P(Negative) &= 1 - P(Positive) = 0.999871775 \end{aligned}$$

Now, we can calculate the conditional distribution on test result,

$$
\begin{aligned}
P(Disease|Positive) &= \frac{P(Positive|Disease) \times P(Disease)}{P(Positive)} \\
&= \frac{0.993 \times 0.000025}{0.0001248225} \\
&= 0.19888241302649762663 \\
P(Non-Disease|Positive) &= 1 - P(Disease|Positive) \\
&= 0.80111758697350237337
\end{aligned}
$$

$$
\begin{aligned}
P(Disease|Negative) &= \frac{P(Negative|Disease) \times P(Disease)}{P(Negative)} \\
&= \frac{0.007 \times 0.000025}{0.999871775} \\
&= 0.0000001750224423 \\
P(Non-Disease|Negative) &= 1 - P(Disease|Negative) \\
&= 0.9999998249775577
\end{aligned}
$$

Or, in a contingency table

```
Positive = c(0.1989,0.8011)
Negative = c(0,1)
table = cbind(Positive,Negative)
df = data.frame(table, row.names = c('Disease','Healthy'))
pander(df)
```

	Positive	Negative
Disease	0.1989	0
Healthy	0.8011	1

From the table above we can see, the problem with the testing policy is that given a positive test result, the probability of having the disease is **19.89%**, which is much lower than the probability of not having the disease 80.11%. Hence, a positive test results are not very effective in implying whether someone has the disease.

Exploratory analysis: green buildings

```
library(mosaic)
library(foreach)
library(ggplot2)
library(plyr)
library(car)
library(plotly)
library(gridExtra)
library(grid)
```

```
green = read.csv('data/greenbuildings.csv', header=TRUE)
```

Data Cleaning and Pre-processing

First, we need to clean the data, create columns with factor levels for categorical variables and drop the columns with dummies, so that we can have better interpretations from plots in next steps.

```
green_dup = green

dummies_green_certificate = model.matrix(~LEED + Energystar - 1, data=green)
dummies_class = model.matrix(~class_a + class_b - 1, data=green)
factor_green_certificate = factor(dummies_green_certificate %*% 1:ncol(dummies_green_certificate),
                                  label = c('None','LEED','Energystar','Both'))

factor_class = factor(dummies_class %*% 1:ncol(dummies_class),
                      label = c('c','a','b'))

green_dup$green_certificate = factor_green_certificate
green_dup$class = factor_class
green_dup$cluster = factor(green$cluster)
green_dup$green_rating = factor(green$green_rating, label = c('No','Yes'))
green_dup$renovated = factor(green$renovated, label = c('No','Yes'))
green_dup$net = factor(green$net, label = c('No','Yes'))
green_dup$amenities = factor(green$amenities, label = c('No','Yes'))

green_clean = subset(green_dup, select = -c(LEED, Energystar, class_a, class_b))
```

Then, we create following subsets of the `green_clean`:

- `non_green`: non-green buildings
- `green_only`: green buildings
- `green_LEED`: green buildings with certification from LEED only
- `green_Energystar`: green buildings with certification from Energystar only
- `green_both`: green buildings with certification from both LEED and Energystar

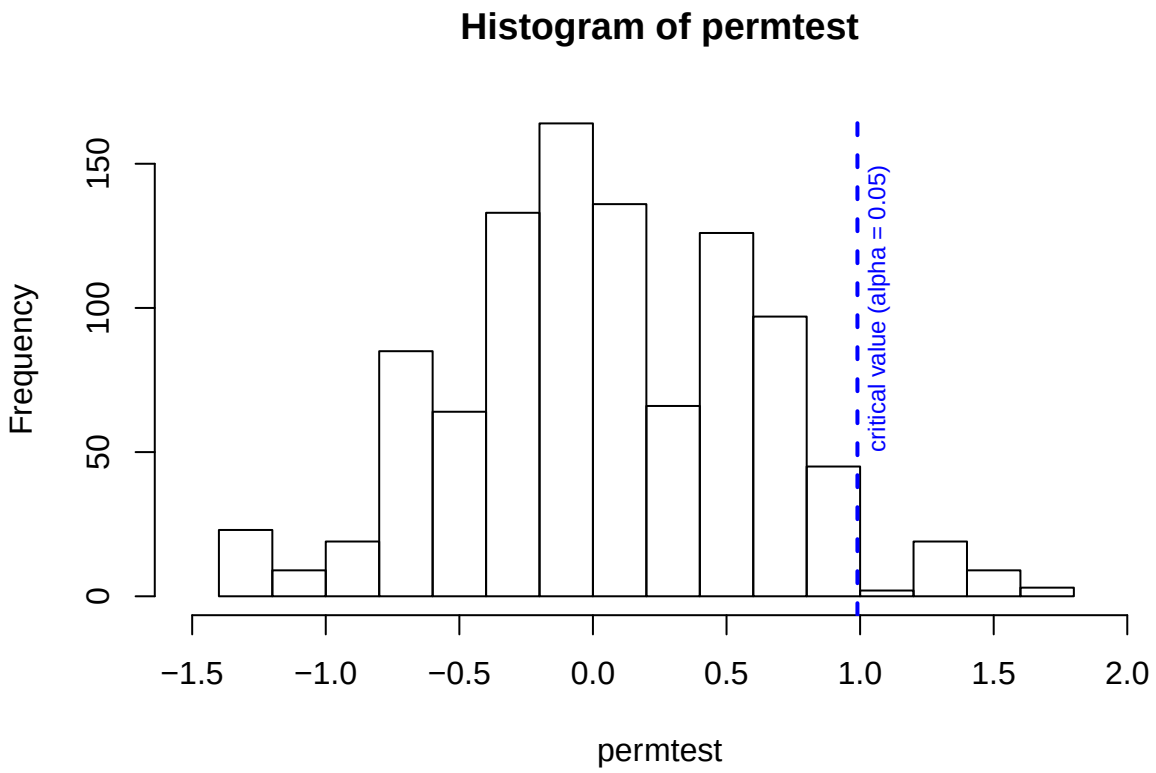
```
non_green = subset(green_clean, green_rating=='No')
green_only = subset(green_clean, green_rating=='Yes')

green_LEED = subset(green_clean, green_certificate=='LEED')
green_Energystar = subset(green_clean, green_certificate=='Energystar')
green_Both = subset(green_clean, green_certificate=='Both')
```

Permutation Test

Before exploring the relationships of the variables, we run a permutation test, to examine whether the difference in median rent for green and non-green buildings as stated by the stats guru, are significant or due to random chance.

```
permtest = foreach(i = 1:1000, .combine='c') %do% {
  t1_shuffle = data.frame(green_clean$Rent, shuffle(green_clean$green_rating))
  green_shuffle = subset(t1_shuffle,shuffle.green_clean.green_rating. == 'Yes')
  non_green_shuffle = subset(t1_shuffle,shuffle.green_clean.green_rating. == 'No')
  median(green_shuffle$green_clean.Rent) - median(non_green_shuffle$green_clean.Rent)
```

```
}
hist(permtest, xlim = c(-1.5,2))
myinterval = quantile(permtest, probs=0.95)
abline(v = myinterval, lwd = 2, lty = 2, col='blue')
text(1,50,'critical value (alpha = 0.05)',pos = 4, col = 'blue', srt = 90, cex = 0.75)
```




```
pdata(permtest, 2.6, lower=FALSE)
```

```
## [1] 0
```

The result above indicates that the $2.6/sqft.yr difference in rent is *significant*, with a p-value close to zero. So our next step is to explore the potential relationships between predictors and rent.

Green vs Non-Green (at a glance)

Histogram for Green vs Non-Green on Rent

```
#plot hist for green vs non-green
m_green = median(green_only$Rent)
m_non_green = median(non_green$Rent)

mybreaks = seq(0, 250, by=5)
hist(non_green$Rent,
     breaks = mybreaks,
     xlab="The rent charged to tenants in the building ($/sqft.yr)",
     main="", border="darkgrey",
     col="grey", axes=FALSE, ylim=c(0, 1500))
axis(2,at=seq(0,1500,by=250), las=1,tick=TRUE)
axis(1,at=seq(0,250,by=25),pos=0)
```

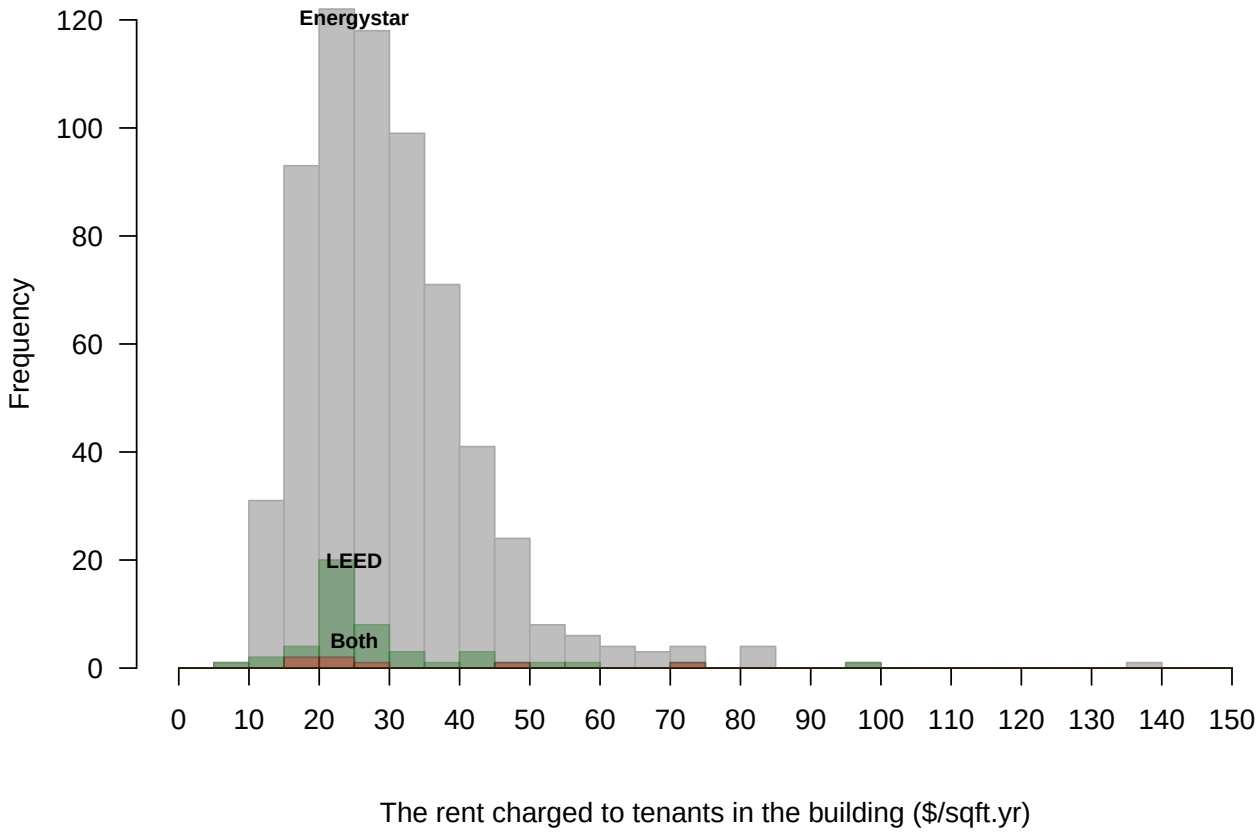
```
hist(green_only$Rent, breaks = mybreaks, add=TRUE, border=rgb(0,100,0,100,maxColorValue=255),
     col= rgb(0,100,0,80,maxColorValue=255))

text(30, 225, "Green Houses",font=2, cex = 0.75)
text(30, 1510, "Non-Green Houses",font=2, cex = 0.75)
```

The histogram above shows the annual rent ($/sqrt) for green buildings and non-green buildings. We can see that the distribution for green buildings has much lower kurtosis.

Histogram for Different Green Cerfiticates on `Rent`

```
#plot hist for LEED vs Energystar
m_green_LEED = median(green_LEED$Rent)
m_green_Energystar = median(green_Energystar$Rent)
m_green_Both = median(green_Both$Rent)

mybreaks = seq(0, 150, by=5)
hist(green_Energystar$Rent,
     breaks = mybreaks,
     xlab="The rent charged to tenants in the building ($/sqft.yr)",
     main="", border="darkgrey",
     col="grey", axes=FALSE, ylim=c(0, 120))
axis(2,at=seq(0,120,by=20), las=1,tick=TRUE)
axis(1,at=seq(0,150,by=10),pos=0)
hist(green_LEED$Rent, breaks = mybreaks, add=TRUE, border=rgb(0,100,0,100,maxColorValue=255),
     col= rgb(0,100,0,80,maxColorValue=255))
```

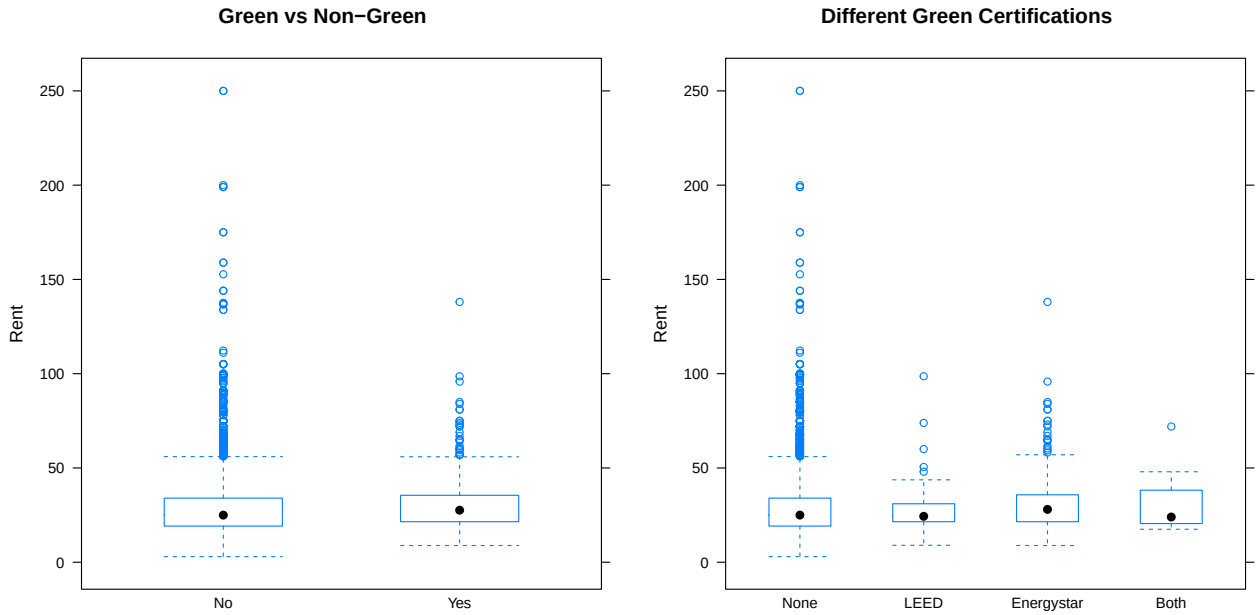
```
hist(green_Both$Rent, breaks = mybreaks, add=TRUE, border=rgb(100,0,0,100,maxColorValue=255),
     col= rgb(255,0,0,80,maxColorValue=255))

text(25, 5, "Both", font=2, cex = 0.75)
text(25, 20, "LEED", font=2, cex = 0.75)
text(25, 120, "Energystar", font=2, cex = 0.75)
```



The histogram above shows the annual rent ($/sqrt) for green buildings with certifications from LEED, Energystar or Both. We can see that the ranges of the distribution for three categories are significantly different.

Boxplots for Green vs Non-Green on `Rent`

```
plot_green = bwplot(Rent ~ green_rating, data = green_clean, main = 'Green vs Non-Green')
plot_certificate = bwplot(Rent ~ green_certificate, data = green_clean,
                          main = 'Different Green Certifications')
grid.arrange(plot_green,plot_certificate, ncol = 2)
```




```
favstats(Rent ~ green_rating, data = green_clean)
```

```
##   green_rating  min    Q1 median   Q3    max     mean       sd    n
## 1           No 2.98 19.18   25.0 34.0 250.00 28.26678 15.25413 7209
## 2          Yes 8.87 21.50   27.6 35.5 138.07 30.01603 12.95008  685
##   missing
## 1       0
## 2       0
```

```
favstats(Rent ~ green_certificate, data = green_clean)
```

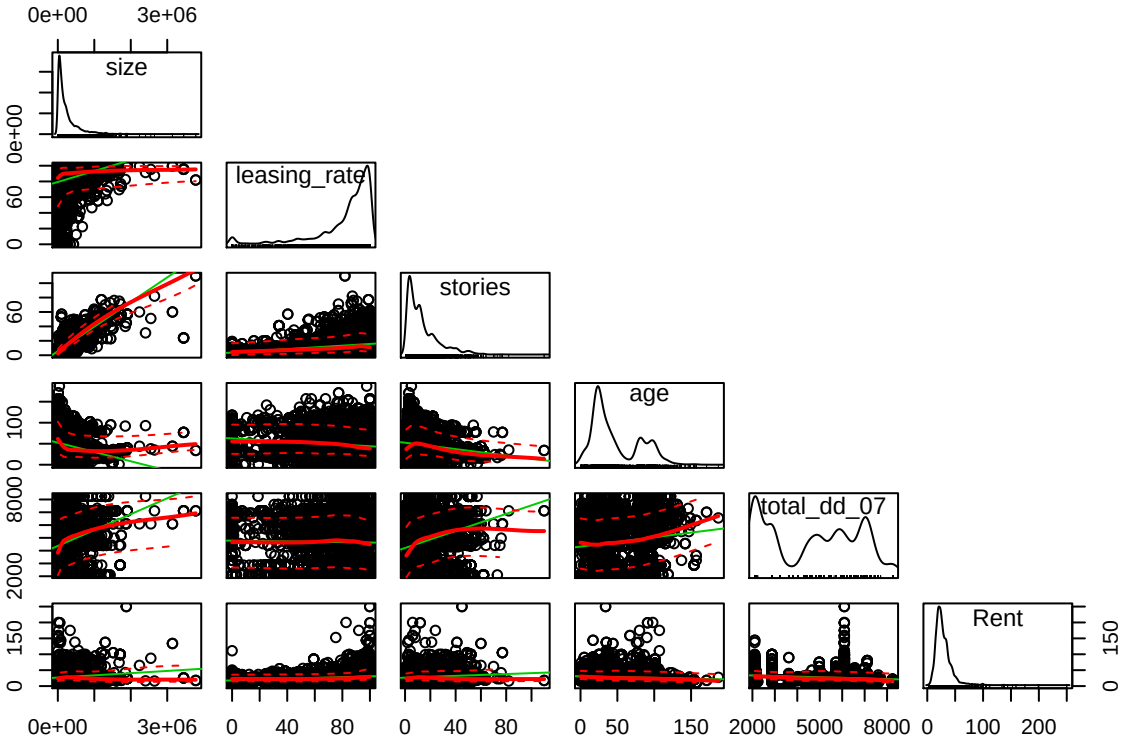
```
##   green_certificate   min    Q1 median     Q3    max     mean       sd
## 1              None  2.98 19.18  25.00 34.000 250.00 28.26678 15.25413
## 2              LEED  9.00 21.50  24.36 31.020  98.65 29.21043 15.95222
## 3        Energystar  8.87 21.50  28.03 35.775 138.07 30.04304 12.63093
## 4              Both 17.50 20.50  24.00 38.215  72.00 32.99000 20.00310
##      n missing
## 1 7209       0
## 2   47       0
## 3  631       0
## 4    7       0
```

The boxplots with summary statistics above shows that the difference in median rent has different directions once we split green buildings into sub-groups based on certifications. We can see that *buildings with LEED only or both certifications has lower median rent than non-green buildings*, which would potentially result in a loss for the construction.

Relationships between `Rent` and other variables

Scatter Plot for Numerical Variables

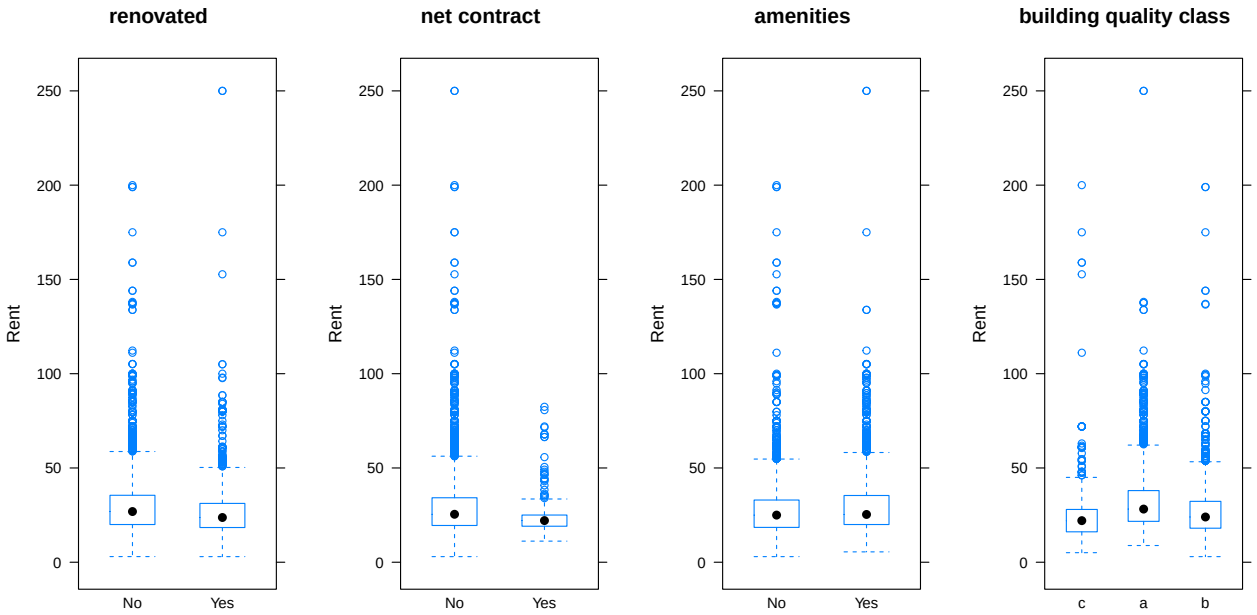
```
scatterplotMatrix(~ size + leasing_rate + stories + age + total_dd_07 + Rent,
                  data = green_clean, upper.panel = NULL)
```



From the pairwise plots above we can see, there are potential relationships between numerical variables with rent, which very likely are factors in the relationship between green buildings and rent.

Box Plot for Categorical Variables

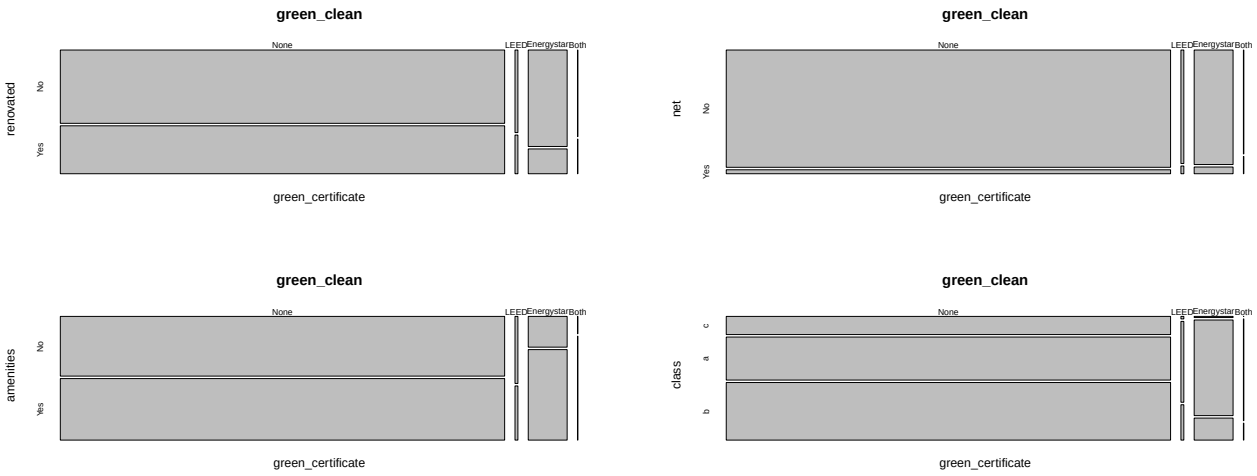
```
plot1 = bwplot(Rent ~ renovated, data = green_clean, main = 'renovated')
plot2 = bwplot(Rent ~ net, data = green_clean, main = 'net contract')
plot3 = bwplot(Rent ~ amenities, data = green_clean, main = 'amenities')
plot4 = bwplot(Rent ~ class, data = green_clean, main = 'building quality class')

grid.arrange(plot1,plot2,plot3,plot4, ncol = 4)
```



From the pairwise plots above we can see, there are potential relationships between categorical variables with rent as well. Again, such relationships are very likely factors in the relationship between green buildings and rent.

Relationship between `green_certificate` and other categorical variables

```
par(mfrow = c(2,2))
mosaicplot(green_certificate ~ renovated, data = green_clean)
mosaicplot(green_certificate ~ net, data = green_clean)
mosaicplot(green_certificate ~ amenities, data = green_clean)
mosaicplot(green_certificate ~ class, data = green_clean)
```

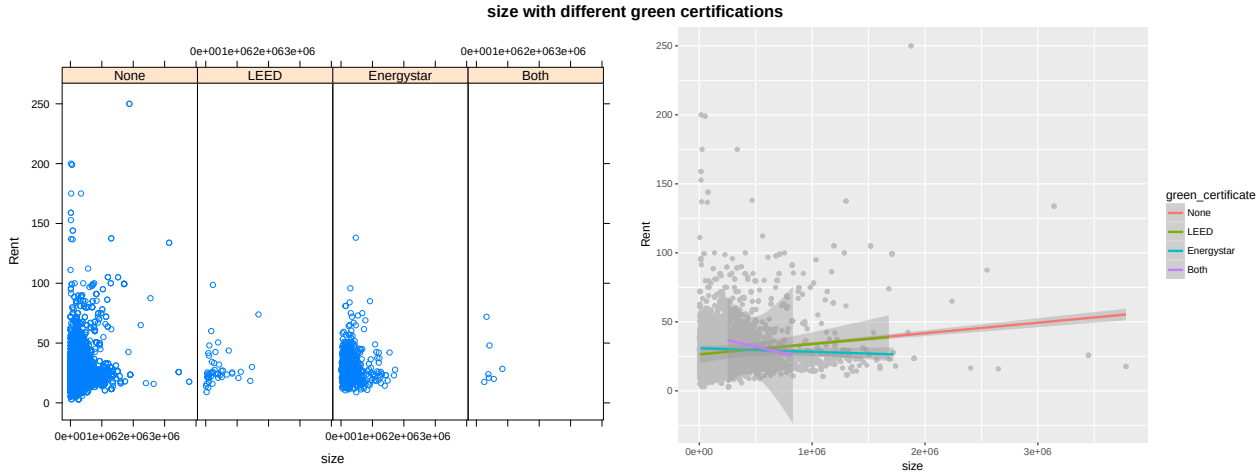


As shown in the mosaicplots between different green certifications and categorical variables, effects of categorical variables on rent are further shown. For example, the 4th plot shows that green buildings are less likely to be built with high qualities (a or b), which can be an important factor for higher rent.

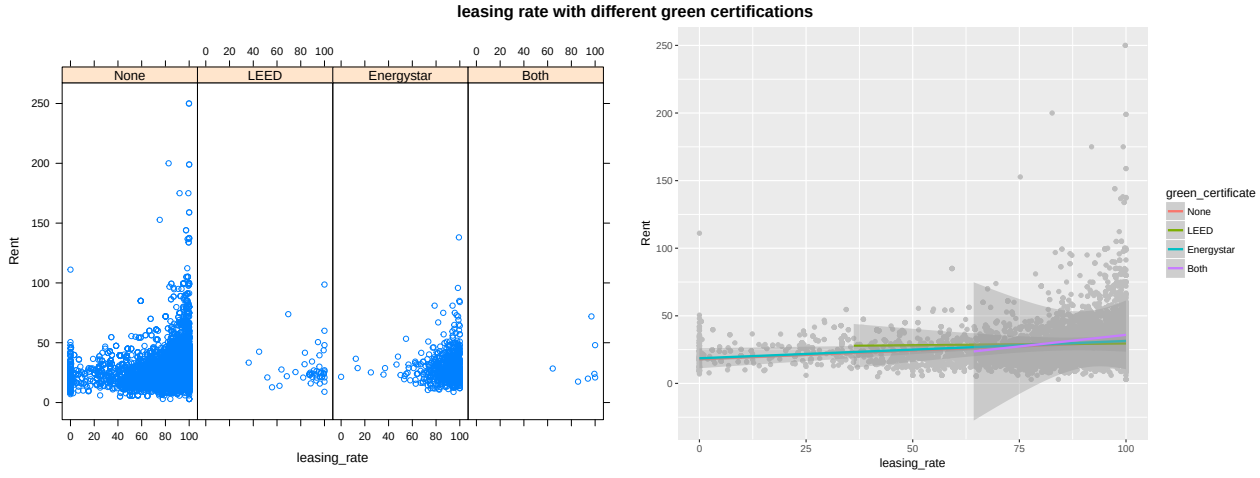
Lattice and Interaction Plot

Numerical Variables with Different Green Certifications

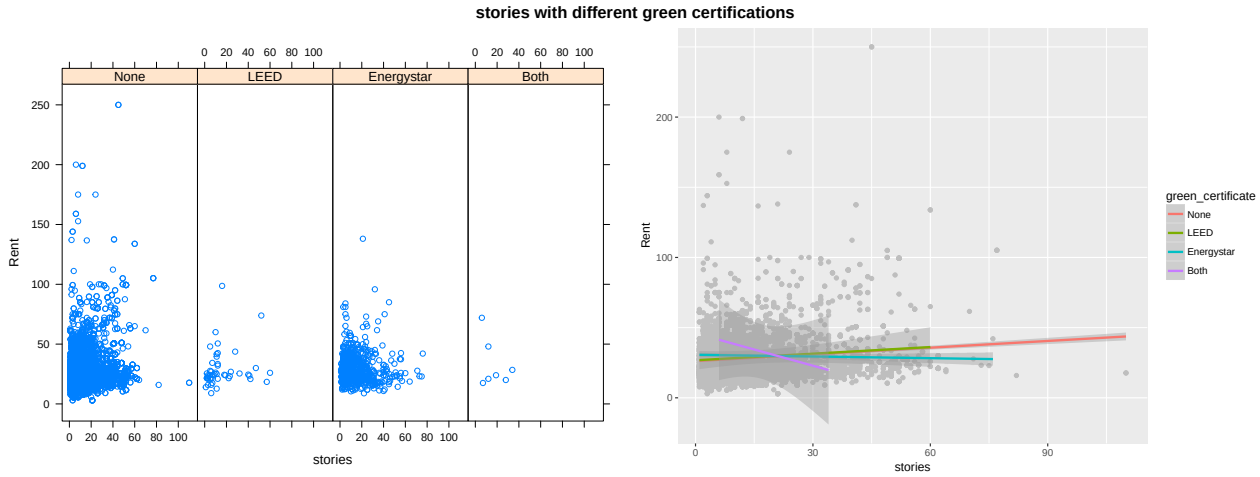
```
plot1_l = xyplot(Rent ~ size | green_certificate, data = green_clean)
plot1_i = ggplot(green_clean) +
  aes(x = size, y = Rent, color = green_certificate) +
  geom_point(color = "grey") +
  geom_smooth(method = "lm")

grid.arrange(plot1_l,plot1_i,ncol = 2,
             top=textGrob("size with different green certifications",
                          gp=gpar(fontsize=15,fontface = 'bold')))
```




```
plot2_l = xyplot(Rent ~ leasing_rate | green_certificate, data = green_clean)
plot2_i = ggplot(green_clean) +
  aes(x = leasing_rate, y = Rent, color = green_certificate) +
  geom_point(color = "grey") +
  geom_smooth(method = "lm")

grid.arrange(plot2_l,plot2_i,ncol = 2,
             top=textGrob("leasing rate with different green certifications",
                          gp=gpar(fontsize=15,fontface = 'bold')))
```



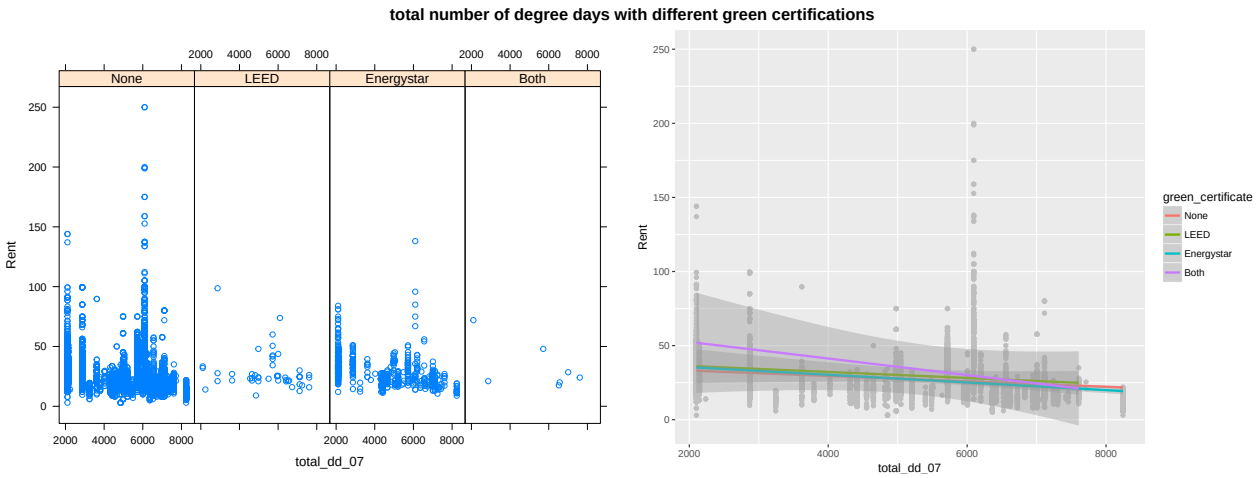
```
plot3_l = xyplot(Rent ~ stories | green_certificate, data = green_clean)
plot3_i = ggplot(green_clean) +
  aes(x = stories, y = Rent, color = green_certificate) +
  geom_point(color = "grey") +
  geom_smooth(method = "lm")

grid.arrange(plot3_l,plot3_i,ncol = 2,
             top=textGrob( 'stories with different green certifications',
                           gp=gpar(fontsize=15,fontface = 'bold')))
```




```
plot4_l = xyplot(Rent ~ age | green_certificate, data = green_clean)
plot4_i = ggplot(green_clean) +
  aes(x = age, y = Rent, color = green_certificate) +
  geom_point(color = "grey") +
  geom_smooth(method = "lm")

grid.arrange(plot4_l,plot4_i,ncol = 2,
             top=textGrob( 'building age with different green certifications',
                           gp=gpar(fontsize=15,fontface = 'bold')))
```


```
plot1_l = xyplot(Rent ~ total_dd_07 | green_certificate, data = green_clean)
plot1_i = ggplot(green_clean) +
  aes(x = total_dd_07, y = Rent, color = green_certificate) +
```

```
  geom_point(color = "grey") +
  geom_smooth(method = "lm")

grid.arrange(plot1_l,plot1_i,ncol = 2,
             top=textGrob("total number of degree days with different green certifications",
                          gp=gpar(fontsize=15,fontface = 'bold')))
```

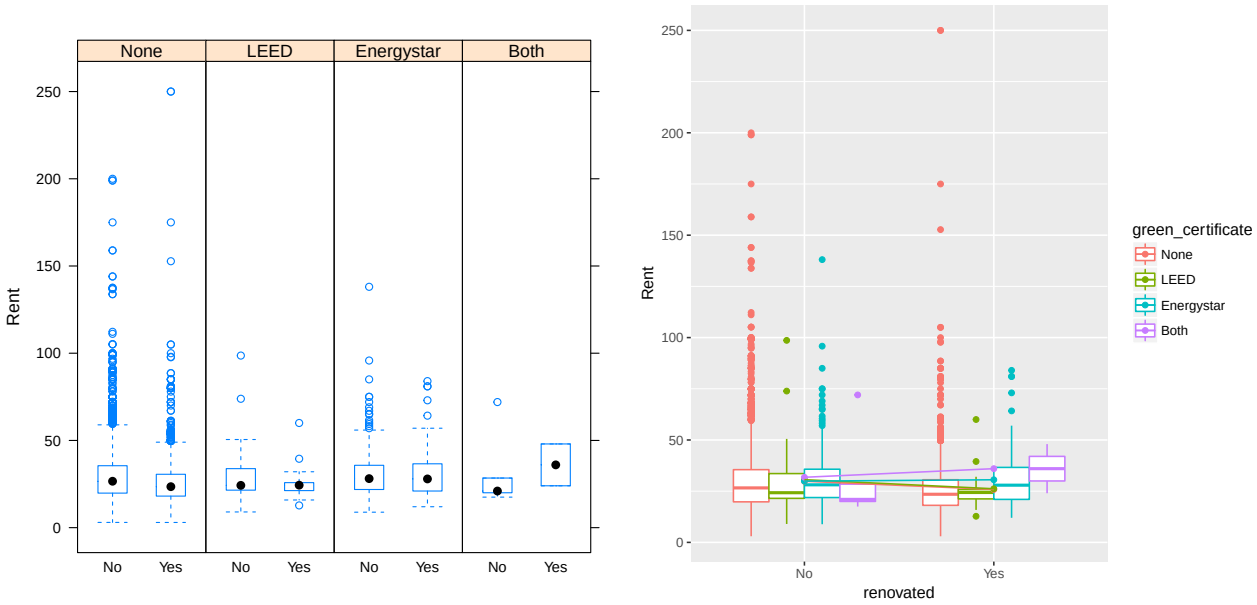


The plots above shows the pairwise interactions with numerical variables and buildings green certifications. Plots on the right hand side indicate significant interactions between them.

Categorical Variables with Different Green Certifications

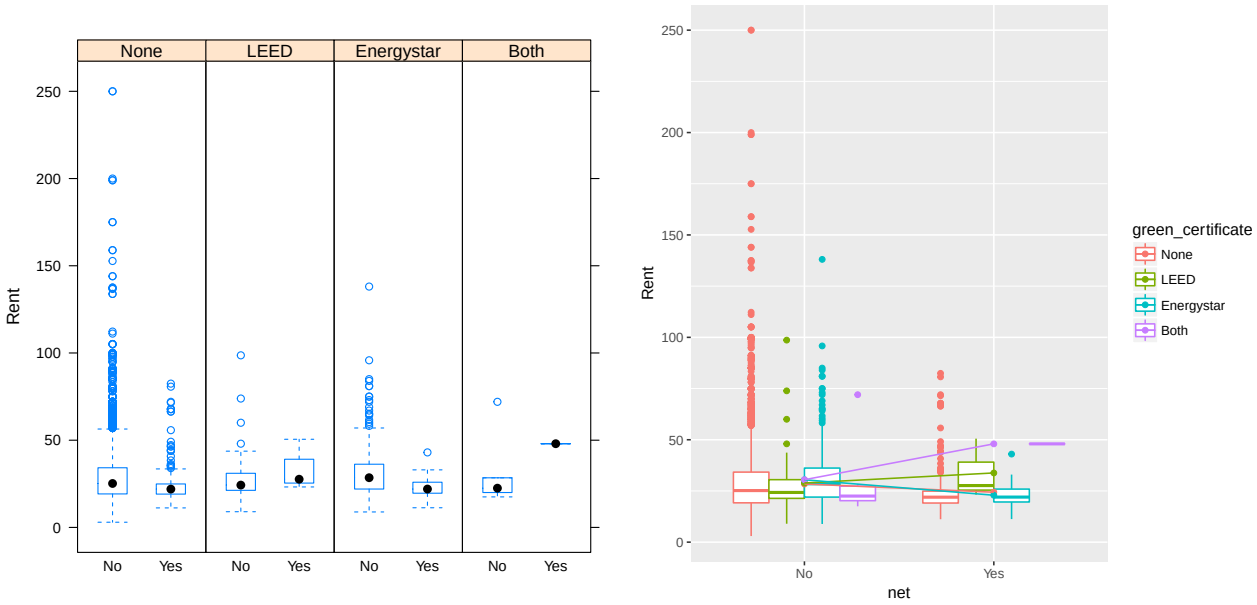
```
plot5_l = bwplot(Rent ~ renovated | green_certificate, data = green_clean)
green_ren_Int <- ddply(green_clean,.(renovated,green_certificate),summarise, val = mean(Rent))
plot5_i = ggplot(green_clean, aes(x = renovated, y = Rent, colour = green_certificate)) +
  geom_boxplot() +
  geom_point(data = green_ren_Int, aes(y = val)) +
  geom_line(data = green_ren_Int, aes(y = val, group = green_certificate))

grid.arrange(plot5_l,plot5_i, ncol=2,
             top=textGrob( 'renovated building with different green certifications',
                          gp=gpar(fontsize=15,fontface = 'bold')))
```

renovated building with different green certifications



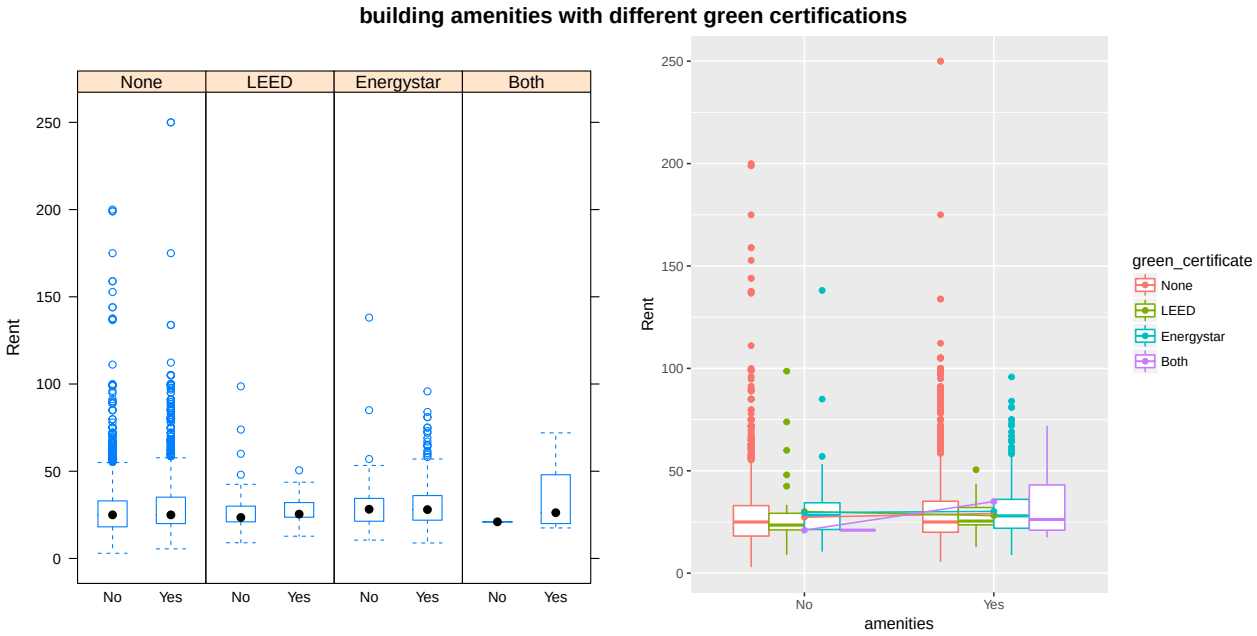

```
plot6_l = bwplot(Rent ~ net | green_certificate, data = green_clean)
green_net_Int <- ddply(green_clean,.(net,green_certificate),summarise, val = mean(Rent))
plot6_i = ggplot(green_clean, aes(x = net, y = Rent, colour = green_certificate)) +
  geom_boxplot() +
  geom_point(data = green_net_Int, aes(y = val)) +
  geom_line(data = green_net_Int, aes(y = val, group = green_certificate))

grid.arrange(plot6_l,plot6_i,ncol = 2,
             top=textGrob( 'net contract building with different green certifications',
                           gp=gpar(fontsize=15,fontface = 'bold')))
```

net contract building with different green certifications




```
plot7_l = bwplot(Rent ~ amenities | green_certificate, data = green_clean)
green_amen_Int <- ddply(green_clean,.(amenities,green_certificate),summarise, val = mean(Rent))
plot7_i = ggplot(green_clean, aes(x = amenities, y = Rent, colour = green_certificate)) +
```

```
  geom_boxplot() +
  geom_point(data = green_amen_Int, aes(y = val)) +
  geom_line(data = green_amen_Int, aes(y = val, group = green_certificate))

grid.arrange(plot7_l,plot7_i,ncol = 2,
             top=textGrob( 'building amenities with different green certifications',
                           gp=gpar(fontsize=15,fontface = 'bold')))
```




```
plot8_l = bwplot(Rent ~ class | green_certificate, data = green_clean)
green_class_Int <- ddply(green_clean,.(class,green_certificate),summarise, val = mean(Rent))
plot8_i = ggplot(green_clean, aes(x = class, y = Rent, colour = green_certificate)) +
  geom_boxplot() +
  geom_point(data = green_class_Int, aes(y = val)) +
  geom_line(data = green_class_Int, aes(y = val, group = green_certificate))

grid.arrange(plot8_l,plot8_i,ncol = 2,
             top=textGrob( 'building quality class with different green certifications',
                           gp=gpar(fontsize=15,fontface = 'bold')))
```

The plots above shows the pairwise interactions with categorical variables and buildings green certifications. Again, plots on the right hand side indicate significant interactions between them.

Linear Regression

To quantify the potential effects of other predictors in rent as shown above, we run three multiple linear regression model to show that by taking into account of other variables (both numerical and categorical), the effect of green buildings on rent may be reduced, possibly even into a negative direction.

Model 1

```
lm.fit1 = lm(Rent ~ green_rating + size + leasing_rate + stories + age + total_dd_07 +
               class + amenities + renovated + net, data = green_clean)
summary(lm.fit1)
```

```
##
## Call:
## lm(formula = Rent ~ green_rating + size + leasing_rate + stories +
##     age + total_dd_07 + class + amenities + renovated + net,
##     data = green_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.945  -7.813  -2.361   4.206 212.499
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.471e+01  8.773e-01  28.164  < 2e-16 ***
## green_ratingYes -2.118e+00  5.816e-01  -3.641 0.000273 ***
## size             6.441e-06  9.565e-07   6.734 1.76e-11 ***
## leasing_rate     9.219e-02  7.664e-03  12.029  < 2e-16 ***
## stories         -6.120e-03  2.359e-02  -0.259 0.795339
## age              5.095e-02  6.571e-03   7.754 1.00e-14 ***
## total_dd_07     -2.006e-03  8.260e-05 -24.290  < 2e-16 ***
```

```
## classa           7.338e+00  6.329e-01  11.593  < 2e-16 ***
## classb           2.432e+00  5.004e-01   4.860 1.20e-06 ***
## amenitiesYes    -9.198e-01  3.619e-01  -2.541 0.011060 *
## renovatedYes    -4.146e+00  3.693e-01 -11.228  < 2e-16 ***
## netYes          -5.672e+00  8.640e-01  -6.565 5.54e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.89 on 7882 degrees of freedom
## Multiple R-squared:  0.152,  Adjusted R-squared:  0.1508
## F-statistic: 128.4 on 11 and 7882 DF,  p-value: < 2.2e-16
```

In the first model, we fit the `Rent` in a linear regressin using all the numerical and categorical variables discussed above, and also `green_rating`. The result shows that most of the coefficients on the predictors are significant, and after taking account for effects from other variables, rent for green buildings on average is \$2.12/sqft.yr lower than rent for non-green buildings.

Model 2

```
lm.fit2 = lm(Rent ~ green_certificate + size + leasing_rate + stories + age + total_dd_07 +
               class + amenities + renovated + net, data = green_clean)
summary(lm.fit2)
```

```
##
## Call:
## lm(formula = Rent ~ green_certificate + size + leasing_rate +
##     stories + age + total_dd_07 + class + amenities + renovated +
##     net, data = green_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.857  -7.790  -2.338   4.224 212.548
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  2.473e+01  8.775e-01  28.188  < 2e-16 ***
## green_certificateLEED       -1.173e-01  2.041e+00  -0.057 0.954166
## green_certificateEnergystar -2.326e+00  6.034e-01  -3.855 0.000117 ***
## green_certificateBoth        2.726e+00  5.259e+00   0.518 0.604284
## size                         6.400e-06  9.570e-07   6.688 2.42e-11 ***
## leasing_rate                 9.217e-02  7.664e-03  12.026  < 2e-16 ***
## stories                     -5.426e-03  2.360e-02  -0.230 0.818165
## age                          5.091e-02  6.572e-03   7.748 1.05e-14 ***
## total_dd_07                 -2.012e-03  8.272e-05 -24.324  < 2e-16 ***
## classa                       7.333e+00  6.329e-01  11.586  < 2e-16 ***
## classb                       2.428e+00  5.005e-01   4.852 1.25e-06 ***
## amenitiesYes                -9.011e-01  3.624e-01  -2.486 0.012930 *
## renovatedYes                -4.152e+00  3.693e-01 -11.243  < 2e-16 ***
## netYes                      -5.683e+00  8.640e-01  -6.577 5.10e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.89 on 7880 degrees of freedom
## Multiple R-squared:  0.1522, Adjusted R-squared:  0.1508
## F-statistic: 108.8 on 13 and 7880 DF,  p-value: < 2.2e-16
```

In the second model, we fit the `Rent` in a linear regressin using all the numerical and categorical variables discussed above, and also `green_classification`. The only difference between this model and model 1 is that now we further classify green buildings by their certifications. The result shows that after taking account for effects from other variables and further classifying on green certifications, rent for green buildings with Energystar certifications on average is $2.33/sqft.yr lower than rent for non-green buildings, with a significant relationship.

Model 3

```
lm.fit3 = lm(Rent ~ green_certificate * size + green_certificate * leasing_rate + green_certificate * s
               green_certificate * age + green_certificate * total_dd_07 + green_certificate * class + a
summary(lm.fit3)
```

```
##
## Call:
## lm(formula = Rent ~ green_certificate * size + green_certificate *
##     leasing_rate + green_certificate * stories + green_certificate *
##     age + green_certificate * total_dd_07 + green_certificate *
##     class + amenities + renovated + net, data = green_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.707  -7.837  -2.334   4.279 211.956
##
## Coefficients: (1 not defined because of singularities)
##                                          Estimate Std. Error t value
## (Intercept)                             2.463e+01  8.927e-01  27.590
## green_certificateLEED                  -6.723e+00  1.666e+01  -0.404
## green_certificateEnergystar            -7.254e-01  7.602e+00  -0.095
## green_certificateBoth                   3.219e+03  1.318e+03   2.443
## size                                    6.617e-06  9.749e-07   6.788
## leasing_rate                            9.117e-02  7.783e-03  11.713
## stories                                 6.357e-03  2.445e-02   0.260
## age                                     4.894e-02  6.635e-03   7.376
## total_dd_07                            -1.952e-03  8.643e-05 -22.585
## classa                                  7.076e+00  6.462e-01  10.950
## classb                                  2.359e+00  5.030e-01   4.691
## amenitiesYes                           -9.918e-01  3.630e-01  -2.732
## renovatedYes                           -4.245e+00  3.698e-01 -11.480
## netYes                                 -5.554e+00  8.666e-01  -6.409
## green_certificateLEED:size              4.397e-06  1.466e-05   0.300
## green_certificateEnergystar:size       -4.513e-06  5.397e-06  -0.836
## green_certificateBoth:size             -4.137e-03  1.732e-03  -2.388
## green_certificateLEED:leasing_rate     -7.474e-02  1.386e-01  -0.539
## green_certificateEnergystar:leasing_rate  7.161e-03  4.767e-02   0.150
## green_certificateBoth:leasing_rate     -2.223e+01  9.113e+00  -2.440
## green_certificateLEED:stories          -1.093e-01  3.701e-01  -0.295
## green_certificateEnergystar:stories    -5.655e-02  1.155e-01  -0.490
## green_certificateBoth:stories           6.850e+01  2.897e+01   2.365
## green_certificateLEED:age              -1.211e-02  7.389e-02  -0.164
## green_certificateEnergystar:age         1.233e-01  5.215e-02   2.364
## green_certificateBoth:age               6.561e+00  2.592e+00   2.531
## green_certificateLEED:total_dd_07      -1.030e-03  1.468e-03  -0.702
## green_certificateEnergystar:total_dd_07 -6.955e-04  3.075e-04  -2.262
## green_certificateBoth:total_dd_07      -1.459e-01  5.932e-02  -2.460
```

```
## green_certificateLEED:classa              1.840e+01  1.556e+01   1.182
## green_certificateEnergystar:classa        5.603e-01  5.952e+00   0.094
## green_certificateBoth:classa              1.639e+02  7.324e+01   2.238
## green_certificateLEED:classb              2.223e+01  1.558e+01   1.426
## green_certificateEnergystar:classb       -1.837e+00  5.968e+00  -0.308
## green_certificateBoth:classb                     NA         NA      NA
##                                          Pr(>|t|)
## (Intercept)                               < 2e-16 ***
## green_certificateLEED                      0.6865
## green_certificateEnergystar                0.9240
## green_certificateBoth                      0.0146 *
## size                                     1.22e-11 ***
## leasing_rate                              < 2e-16 ***
## stories                                    0.7949
## age                                      1.80e-13 ***
## total_dd_07                               < 2e-16 ***
## classa                                    < 2e-16 ***
## classb                                   2.77e-06 ***
## amenitiesYes                               0.0063 **
## renovatedYes                              < 2e-16 ***
## netYes                                   1.55e-10 ***
## green_certificateLEED:size                 0.7642
## green_certificateEnergystar:size           0.4031
## green_certificateBoth:size                 0.0169 *
## green_certificateLEED:leasing_rate         0.5897
## green_certificateEnergystar:leasing_rate   0.8806
## green_certificateBoth:leasing_rate         0.0147 *
## green_certificateLEED:stories              0.7677
## green_certificateEnergystar:stories        0.6244
## green_certificateBoth:stories              0.0181 *
## green_certificateLEED:age                  0.8699
## green_certificateEnergystar:age            0.0181 *
## green_certificateBoth:age                  0.0114 *
## green_certificateLEED:total_dd_07          0.4828
## green_certificateEnergystar:total_dd_07    0.0238 *
## green_certificateBoth:total_dd_07          0.0139 *
## green_certificateLEED:classa               0.2371
## green_certificateEnergystar:classa         0.9250
## green_certificateBoth:classa               0.0252 *
## green_certificateLEED:classb               0.1538
## green_certificateEnergystar:classb         0.7582
## green_certificateBoth:classb                   NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.88 on 7860 degrees of freedom
## Multiple R-squared:  0.1562, Adjusted R-squared:  0.1526
## F-statistic: 44.08 on 33 and 7860 DF,  p-value: < 2.2e-16
```

In the third model, we fit the `Rent` in a linear regressin using all the numerical and categorical variables discussed above, including their interaction with `green_certificate`. The result further proves that the relationships between rent and green buildings are correlated with the effect from other predictors. Further more, different green certifications can have very different effects on rent.

Summary

In a nutshell, the evaluation from the stats guru is too "simple". First we perform a permutation test, showing that the difference in median rent for green and non-green buildings is not by random chance. Then we further examine possible confounding effect on rent from other predictors using histogram, box plots, mosaic plots, lattic plots and interaction plots. Lastly we perform linear regression models to quantify the effect on the relationship with rent from other predictors. We found that most of the relationships between other variables and rent are significant, which have shrinken the relationship of green buildings. Moreover, such relationship is very different based on different green certifications. Thus, for more accurate predictions on the profitability of this project, we need to take into accounts of other predictors, and the certification type of the green building.

Bootstrapping

```
library(mosaic)
library(foreach)
library(quantmod)
```

Market Price Data from Yahoo

```
mystocks = c('SPY','TLT', 'LQD', 'EEM', 'VNQ')
getSymbols(mystocks,src="yahoo")
```

```
## Warning: LQD contains missing values. Some functions will not work if
## objects contain missing values in the middle of the series. Consider using
## na.omit(), na.approx(), na.fill(), etc to remove or replace them.
```

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

Daily Returns for Each ETF

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

all_returns = cbind(ClCl(SPYa),ClCl(TLTa),ClCl(LQDa),ClCl(EEMa),ClCl(VNQa))
myreturns = data.frame(na.omit(all_returns))
```

Risk in Returns for Each ETF

```
summary(myreturns)
```

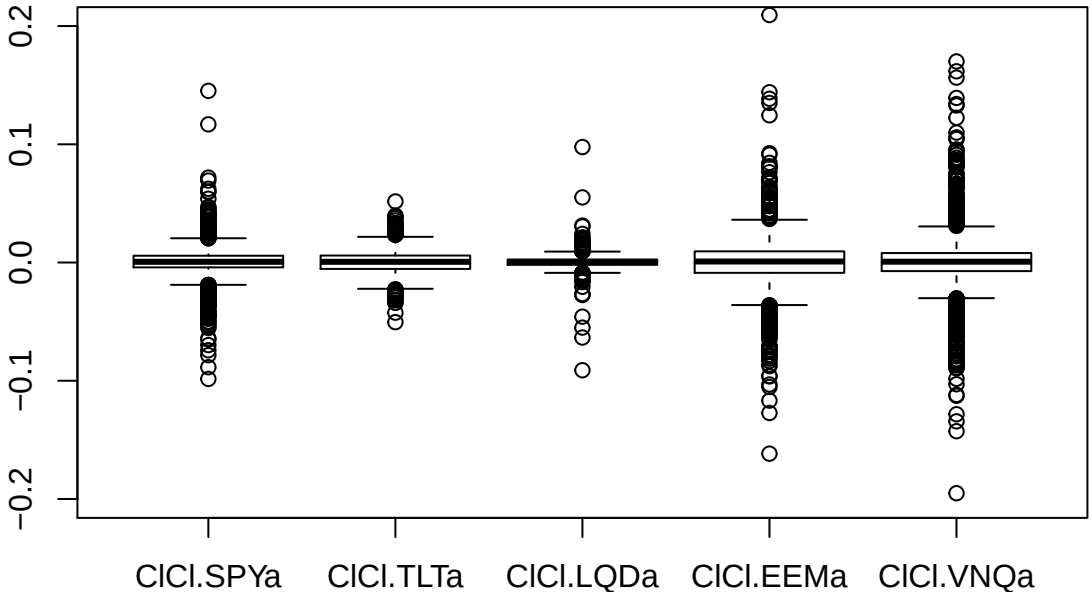
```
##    ClCl.SPYa            ClCl.TLTa            ClCl.LQDa
##  Min.   :-0.0984477   Min.   :-0.0504495   Min.   :-0.0911111
##  1st Qu.:-0.0041093   1st Qu.:-0.0054183   1st Qu.:-0.0019680
##  Median : 0.0006062   Median : 0.0005947   Median : 0.0004316
##  Mean   : 0.0003614   Mean   : 0.0003069   Mean   : 0.0002290
```

```
##  3rd Qu.: 0.0057484   3rd Qu.: 0.0059098   3rd Qu.: 0.0025554
##  Max.   : 0.1451977   Max.   : 0.0516616   Max.   : 0.0976772
##    ClCl.EEMa            ClCl.VNQa
##  Min.   :-0.1616620   Min.   :-0.1951372
##  1st Qu.:-0.0087326   1st Qu.:-0.0072667
##  Median : 0.0008534   Median : 0.0006913
##  Mean   : 0.0010627   Mean   : 0.0004338
##  3rd Qu.: 0.0094581   3rd Qu.: 0.0080174
##  Max.   : 1.8891250   Max.   : 0.1700654
```

```
apply(myreturns, 2, sd)
```

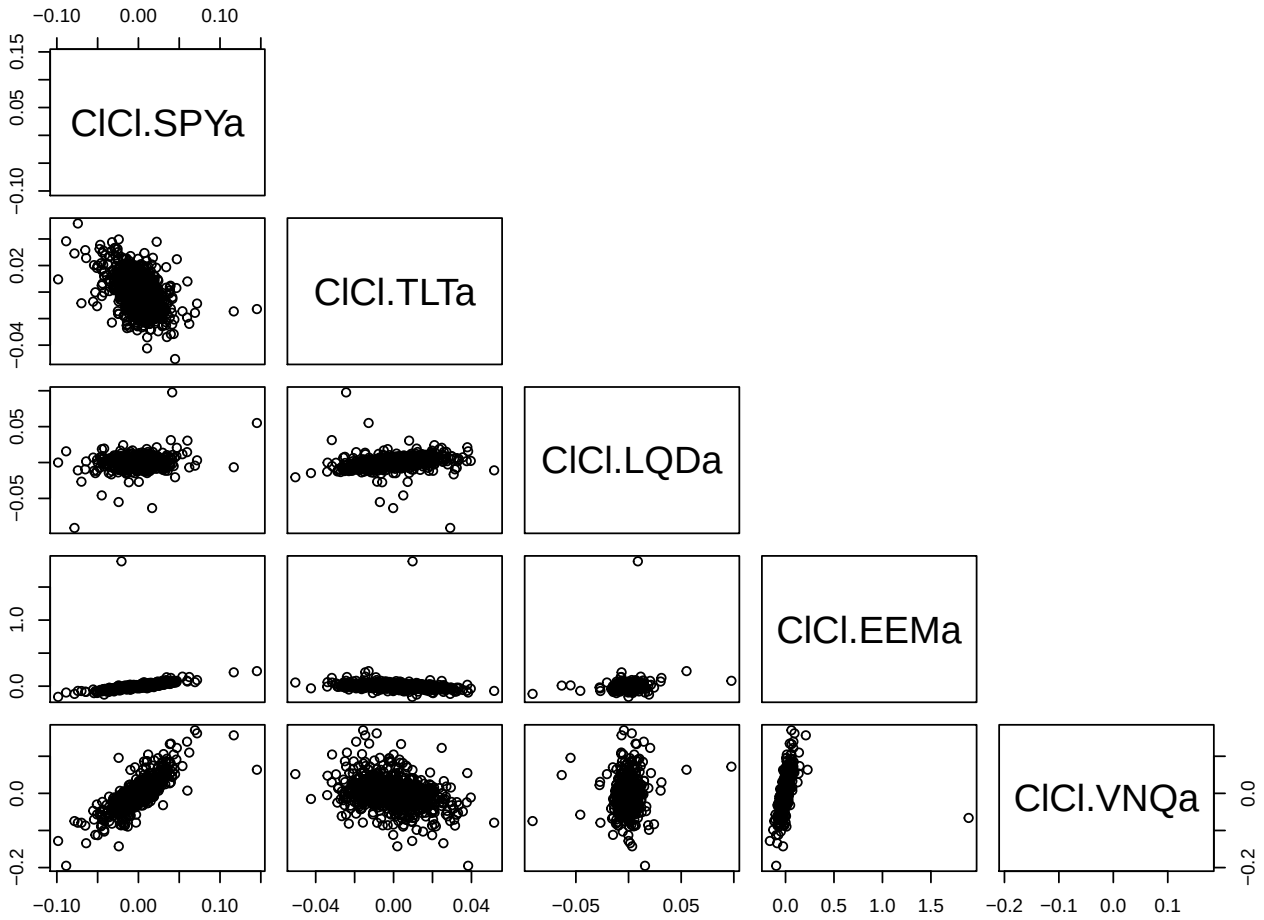
```
##   ClCl.SPYa   ClCl.TLTa   ClCl.LQDa   ClCl.EEMa   ClCl.VNQa
## 0.012790205 0.009403909 0.005406605 0.041974329 0.021997783
```

```
boxplot(myreturns, ylim=c(-0.2,0.2))
```



From the summary statistics and box plots above we can see mean, median daily returns for each of the five ETF. We can also see the quantiles and standard deviation of each ETF. We consider standard deviation of the sample daily return as a good indicator for risk since it quantifies the volatility of the return, which ultimately indicating the potential risk accosiated with each ETF. Out of the five asset classes, LQD appears to have lowest expected daily return with smallest volitility, which can be viewed as a safe alternative. On the other hand, EEM appears to have highest expected daily return along with largest volitility, which can be viewed as an aggressive alternative.

```
pairs(myreturns, upper.panel = NULL)
```




```
cor(myreturns)
```

```
##             ClCl.SPYa  ClCl.TLTa  ClCl.LQDa   ClCl.EEMa   ClCl.VNQa
## ClCl.SPYa  1.0000000 -0.4434811 0.10255294  0.40329628  0.77671377
## ClCl.TLTa -0.4434811  1.0000000 0.42152831 -0.16861612 -0.26520718
## ClCl.LQDa  0.1025529  0.4215283 1.00000000  0.08800942  0.06703847
## ClCl.EEMa  0.4032963 -0.1686161 0.08800942  1.00000000  0.29131651
## ClCl.VNQa  0.7767138 -0.2652072 0.06703847  0.29131651  1.00000000
```

Since these five asset classes are not indipendent, the risk for a portfolio containing these five ETFs can either be diversified or increased by combining negatively or possitively correlated assets. The table and plot above shows the relationship between them.

Portfolio Weights

Weights Combinations of 5 ETFs

To better determine the weights for safe and aggressive portfolio, we consider all possible combinations of weights from 0 to 1 (smallest unit is 0.1).

```
combs = data.frame(matrix(0,1002,5))
count = 1
for (i in seq(0,1,by=0.1)){
  for (j in seq(0,1-i,by=0.1)){
    for(k in seq(0,1-i-j,by=0.1)){
```

```
      for(l in seq(0,1-i-j-k, by=0.1)){
        m = 1 - i - j - k -l
        combs[count,] = c(i,j,k,l,m)
        count = count + 1
      }
    }
  }
}
```

Safe Portfolio

First let's look at possible combinations for safe portfolio. We filter out all the portfolios that has more than 2 zero-weight ETFs, and for all of the rest portfolios, we compute the mean and standard deviation for each portfolio.

```
combs_safe = combs[rowSums(combs==0)<=2,]
n_safe = nrow(combs_safe)
m_safe = rep(0,n_safe)
sd_safe = rep(0,n_safe)
low_safe = rep(0,n_safe)
up_safe = rep(0,n_safe)
for (i in 1:n_safe){
  comb = data.matrix(myreturns) %*% diag(combs_safe[i,])
  m_safe[i] = mean(comb[,1] + comb[,2] + comb[,3] + comb[,4] + comb[,5])
  sd_safe[i] = sd(comb[,1] + comb[,2] + comb[,3] + comb[,4] + comb[,5])
}
safe_stat = data.frame(cbind(m_safe,sd_safe))
head(safe_stat[order(safe_stat$sd_safe),])
```

```
##           m_safe     sd_safe
## 550 0.0002710469 0.004814830
## 571 0.0002788402 0.004822594
## 361 0.0002578018 0.004935254
## 702 0.0002920852 0.004960177
## 522 0.0002632535 0.004960220
## 325 0.0002500085 0.004972450
```

The table above shows the mean and standard deviation in daily returns for each of the portfolio, ordering by standard deviation in ascending order. The first portfolio has the lowest standard deviation, and slightly lower mean than the second portfolio. But notice that here the difference in standard deviation is much larger than the difference in mean, hence the first portfolio can be a good choice for the safe portfolio.

```
best_safe = which.min(sd_safe)
best_comb_safe = combs_safe[best_safe,]
best_comb_safe
```

```
##      X1  X2  X3 X4 X5
## 615 0.2 0.2 0.6  0  0
```

The weights for the safe portfolio is 20% SPY, 20% TLT, 60% LQD, which is also intuitively safe when we look at the volatility of each ETF.

Aggressive Portfolio

Now let's look at possible combinations for aggressive portfolio. We filter out all the portfolios that has more than 3 zero-weight ETFs, and for all of the rest portfolios, we compute the mean and standard deviation for each portfolio.

```
combs_agg = combs[rowSums(combs==0)<=3,]
n_agg = nrow(combs_agg)
m_agg = rep(0,n_agg)
sd_agg = rep(0,n_agg)
low_agg = rep(0,n_agg)
up_agg = rep(0,n_agg)
for (i in 1:n_agg){
  comb = data.matrix(myreturns) %*% diag(combs_agg[i,])
  m_agg[i] = mean(comb[,1] + comb[,2] + comb[,3] + comb[,4] + comb[,5])
  sd_agg[i] = sd(comb[,1] + comb[,2] + comb[,3] + comb[,4] + comb[,5])
}

agg_stat = data.frame(cbind(m_agg,sd_agg,low_agg,up_agg))
head(agg_stat[order(agg_stat$sd_agg,decreasing = TRUE),])
```

```
##            m_agg     sd_agg low_agg up_agg
## 9   0.0009998328 0.03847532       0      0
## 292 0.0009925950 0.03831060       0      0
## 19  0.0009793500 0.03782831       0      0
## 73  0.0009871433 0.03762975       0      0
## 8   0.0009369400 0.03511426       0      0
## 291 0.0009297023 0.03487271       0      0
```

The table above shows the mean and standard deviation in daily returns for each of the portfolio, ordering by standard deviation in descending order. The first portfolio has the highest standard deviation, and higher mean than the other portfolios with high standard deviation, hence the first portfolio can be a good choice for the aggressive portfolio.

```
best_agg = which.max(sd_agg)
best_comb_agg = combs_agg[best_agg,]
best_comb_agg
```

```
##    X1 X2 X3  X4  X5
## 10  0  0  0 0.9 0.1
```

The weights for the safe portfolio is 90% SPY, 10% LQD. Again, the result is fairly intuitive as we discussed before that SPY is very volatile yet with very high expected return.

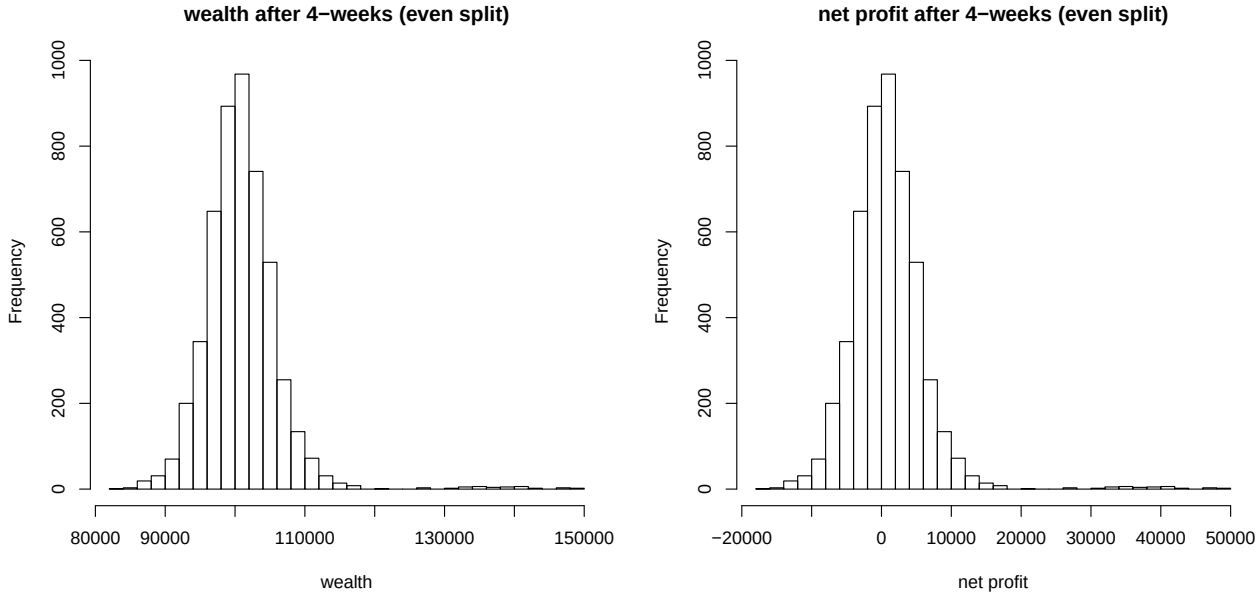
Bootstrap Resampling

Now, let's run bootstrap resampling for even split, safe and aggressive portfolio. For each portfolio, we run 5000 simulations.

Even Split

```
sim_even = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = c(0.2,0.2,0.2,0.2,0.2)
```

```
  holdings = weights * totalwealth
  n_days = 20
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
    holdings = weights * totalwealth
  }
  wealthtracker
}

par(mfrow = c(1,2))
hist(sim_even[,n_days], 25, main = 'wealth after 4-weeks (even split)', xlab = 'wealth')
hist(sim_even[,n_days]- 100000, 25, main = 'net profit after 4-weeks (even split)',
     xlab = 'net profit')
```

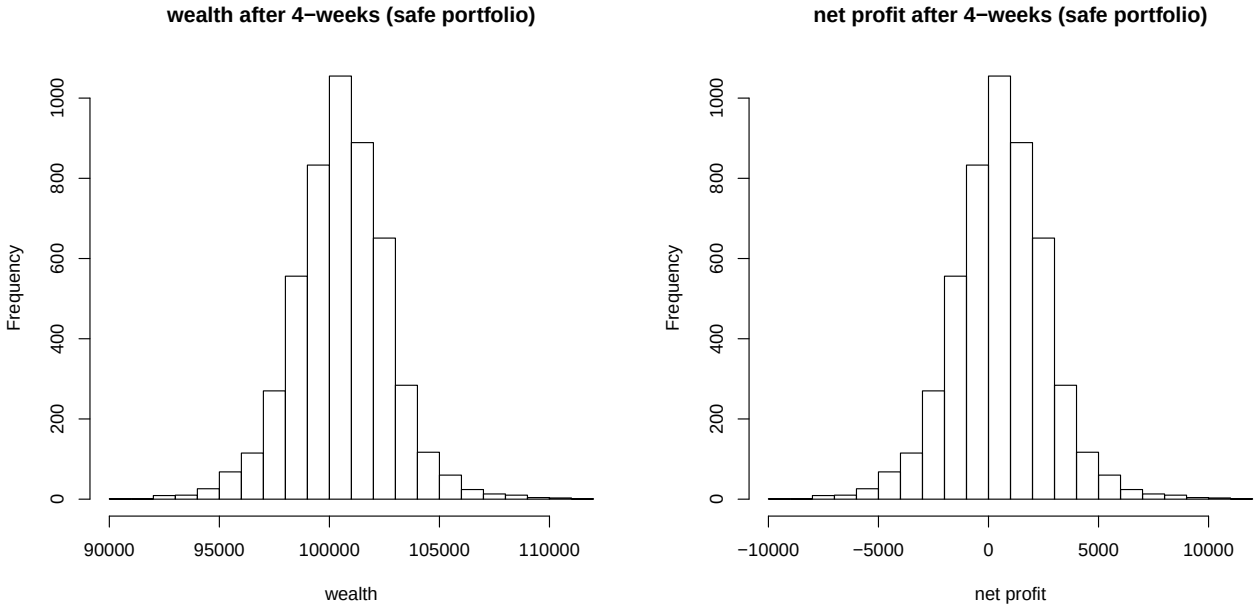



```
var_even5 = round(quantile(sim_even[,n_days], 0.05) - 100000, digits = 2)
var_even95 = round(quantile(sim_even[,n_days], 0.95) - 100000, digits = 2)
mean_even = round(mean(sim_even[,n_days]) - 100000, digits = 2)
sd_even = round(sd(sim_even[,n_days]), digits = 2)
```

Safe Portfolio

```
sim_safe = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = best_comb_safe
  holdings = weights * totalwealth
  n_days = 20
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
```

```
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
    holdings = weights * totalwealth
  }
  wealthtracker
}

par(mfrow = c(1,2))
hist(sim_safe[,n_days], 25, main = 'wealth after 4-weeks (safe portfolio)', xlab = 'wealth')
hist(sim_safe[,n_days]- 100000, 25, main = 'net profit after 4-weeks (safe portfolio)',
     xlab = 'net profit')
```

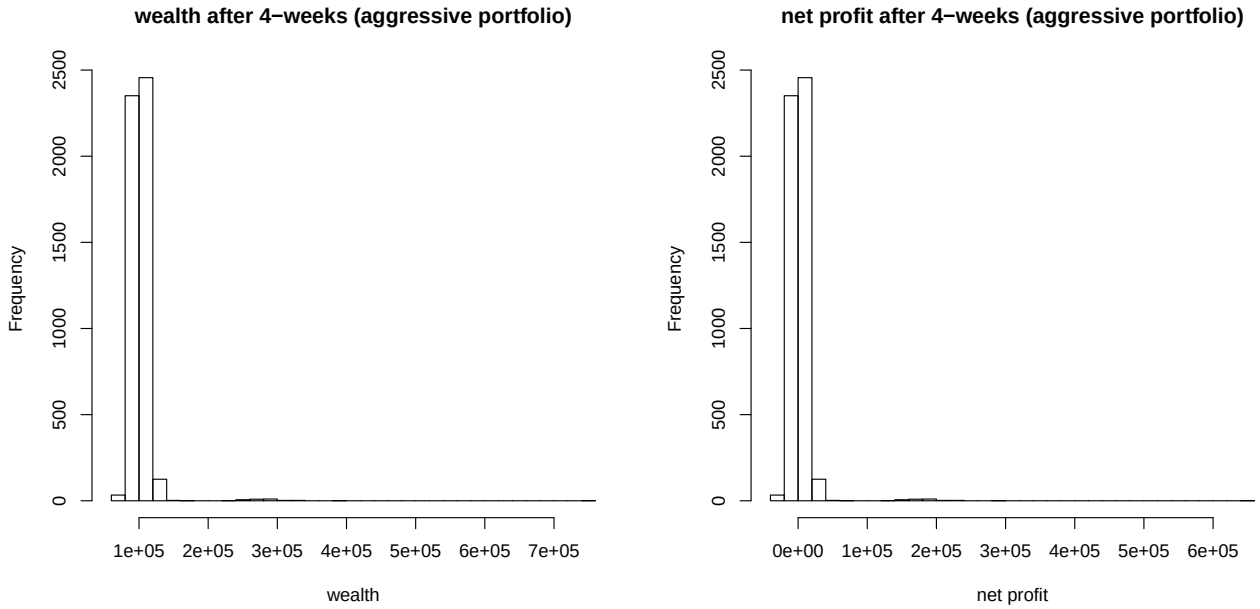



```
var_safe5 = round(quantile(sim_safe[,n_days], 0.05) - 100000, digits = 2)
var_safe95 = round(quantile(sim_safe[,n_days], 0.95) - 100000, digits = 2)
mean_safe = round(mean(sim_safe[,n_days]) - 100000, digits = 2)
sd_safe = round(sd(sim_safe[,n_days]),digits = 2)
```

Aggressive Portfolio

```
sim_agg = foreach(i=1:5000, .combine='rbind') %do% {
  totalwealth = 100000
  weights = best_comb_agg
  holdings = weights * totalwealth
  n_days = 20
  wealthtracker = rep(0, n_days)
  for(today in 1:n_days) {
    return.today = resample(myreturns, 1, orig.ids=FALSE)
    holdings = holdings + holdings*return.today
    totalwealth = sum(holdings)
    wealthtracker[today] = totalwealth
    holdings = weights * totalwealth
  }
```

```
  wealthtracker
}

par(mfrow = c(1,2))
hist(sim_agg[,n_days], 25, main = 'wealth after 4-weeks (aggressive portfolio)', xlab = 'wealth')
hist(sim_agg[,n_days]- 100000, 25, main = 'net profit after 4-weeks (aggressive portfolio)',
     xlab = 'net profit')
```




```
var_agg5 = round(quantile(sim_agg[,n_days], 0.05) - 100000, digits = 2)
var_agg95 = round(quantile(sim_agg[,n_days], 0.95) - 100000, digits = 2)
mean_agg = round(mean(sim_agg[,n_days]) - 100000, digits = 2)
sd_agg = round(sd(sim_agg[,n_days]), digits = 2)
```

Result

The table below demonstrate the result of return in 4 trading weeks for each portfolio. Mean, stanfard deviation, 5% VaR and 95% VaR are reported.

```
even_result = c('Even Split', '20%, 20%, 20%, 20%, 20%', mean_even, sd_even, var_even5, var_even95)
safe_result = c('Safe', '20%, 20%, 60%, 0%, 0%', mean_safe, sd_safe, var_safe5, var_safe95)
agg_result = c('Aggressive', '0%, 0%, 0%, 90%, 1`0%', mean_agg, sd_agg, var_agg5, var_agg95)
result = data.frame(rbind(even_result, safe_result, agg_result))
colnames(result) = c('Portfolio','% in SPY, TLT, LQD, EEM, VNQ',
                     'Mean($)', 'Std Dev($)', 'VaR at 5%($)', 'VaR at 95%($)')
rownames(result) = NULL
pander(result, split.cells = c(8,25,8,8,8,8))
```

Portfolio	% in SPY, TLT, LQD, EEM, VNQ	Mean($)	Std Dev($)	VaR at 5%($)	VaR at 95%($)
Even Split	20%, 20%, 20%, 20%, 20%	924.4	5510.5	-6501.66	8647.83
Safe	20%, 20%, 60%, 0%, 0%	563.35	2146.71	-2882.79	3918.93
Aggressive	0%, 0%, 0%, 90%, 1‘0%	2196.15	19305.95	-12975.72	17309.62

Market segmentation

```
library(pander)
library(ggplot2)
library(LICORS)
library(foreach)
library(mosaic)
library(gridExtra)
library(wordcloud)
```

Data Cleaning and Pre-processing

First, we need to clean the dataset. As mentioned in the description, most of the tweets with `spam` and `adult` info are filtered. So to reduce the bias, it would better to take these two categories out. Also, we take out `uncategorized` and `chatter` since they are usually used as the categories when annotator can't figure out what categories to put. Lastly, we take out `X` make it as the index for the dataframe.

```
social_raw = read.csv('data/social_marketing.csv')
rownames(social_raw) = social_raw$X
social_clean = subset(social_raw, select = -c(X,uncategorized, spam, adult, chatter))
social_scaled = scale(social_clean, center=TRUE, scale=TRUE)
```

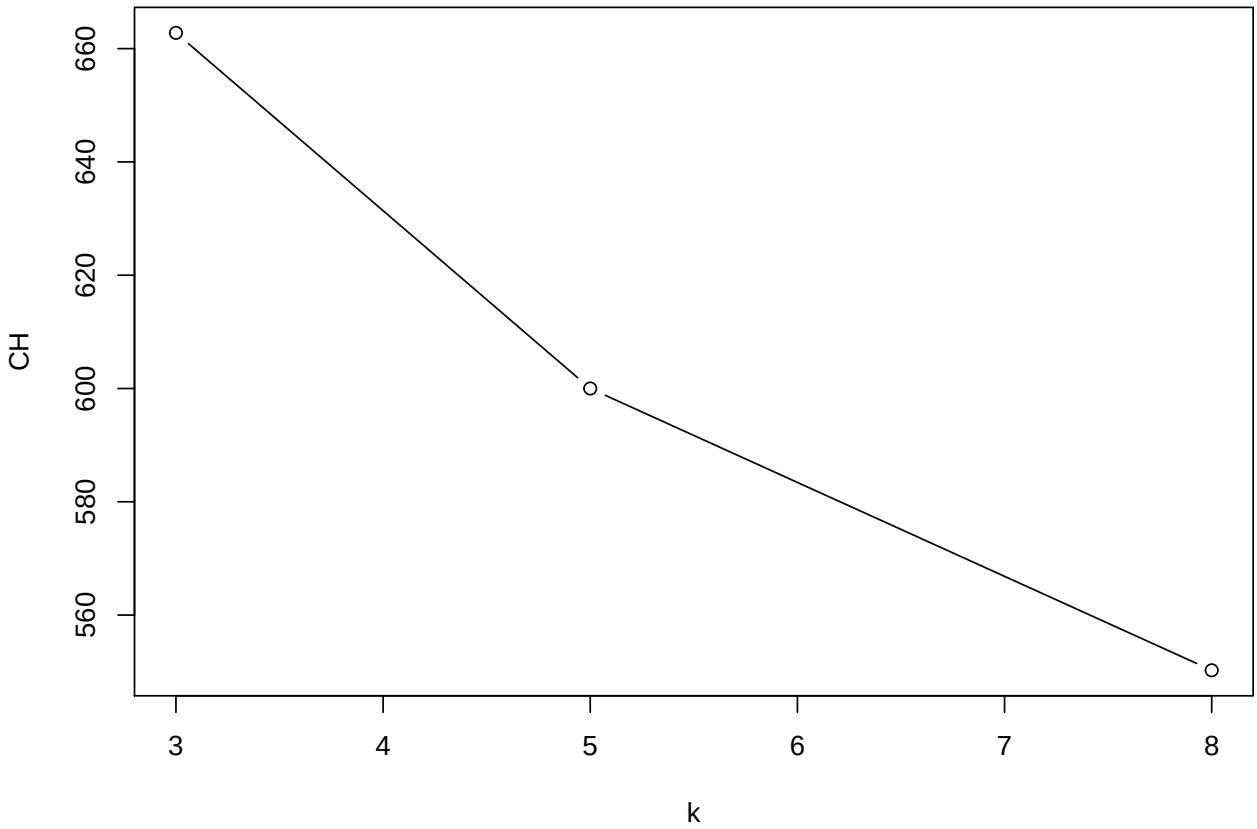
K-means Clustering

We first try out clustering using K-means. The drawback for K-means is that we need to pre-specify the number of clusters (K) before we run the model. So we pick K = 3, 5, 8 and compare the clustering results.

Random initial cluster center

The first method we try is to randomly pick a starting point for K-means, and find the local optimal.

```
kmeans_social3 = kmeans(social_scaled, 3, nstart = 25)
kmeans_social5 = kmeans(social_scaled, 5, nstart = 25)
kmeans_social8 = kmeans(social_scaled, 8, nstart = 25)
CH3 = kmeans_social3$betweenss/kmeans_social3$tot.withinss * (7882 - 3)/(3-1)
CH5 = kmeans_social5$betweenss/kmeans_social5$tot.withinss * (7882 - 5)/(5-1)
CH8 = kmeans_social8$betweenss/kmeans_social8$tot.withinss * (7882 - 8)/(8-1)

k = c(3,5,8)
CH = c(CH3,CH5,CH8)
plot(CH ~ k, type = 'b')
```



We first compute the CH-score for each of the clustering results, it turns out that the score is highest when K = 3. However, since this question is more focused on interpreting the groups(clusters) features and potential user groups, so we decide to look at the top ten categories for each cluster under each K clustering.

```
print(apply(kmeans_social3$centers,1,function(x) colnames(social_scaled)
            [order(x, decreasing=TRUE)[1:10]]))
```

```
##       1                 2                 3
##  [1,] "current_events"  "religion"        "cooking"
##  [2,] "online_gaming"   "parenting"       "fashion"
##  [3,] "automotive"      "sports_fandom"   "health_nutrition"
##  [4,] "news"            "food"            "personal_fitness"
##  [5,] "dating"          "school"          "outdoors"
##  [6,] "home_and_garden" "family"          "photo_sharing"
##  [7,] "tv_film"         "crafts"          "beauty"
##  [8,] "college_uni"     "beauty"          "music"
##  [9,] "politics"        "home_and_garden" "politics"
## [10,] "art"             "eco"             "computers"
```

```
kmeans_social3$size
```

```
## [1] 5046  822 2014
```

With 3 clusters, we can see some pattern in each group, however, it's not easy to tell what each group might be demographically.

```
print(apply(kmeans_social5$centers,1,function(x) colnames(social_scaled)
            [order(x, decreasing=TRUE)[1:10]]))
```

```
##       1                2                3                4
```

```
##  [1,] "cooking"        "college_uni"    "politics"       "religion"
##  [2,] "fashion"        "online_gaming"  "news"           "parenting"
##  [3,] "beauty"         "tv_film"        "travel"         "sports_fandom"
##  [4,] "photo_sharing"  "art"            "computers"      "food"
##  [5,] "music"          "current_events" "automotive"     "school"
##  [6,] "shopping"       "shopping"       "business"       "family"
##  [7,] "small_business" "small_business" "small_business" "crafts"
##  [8,] "business"       "sports_playing" "dating"         "beauty"
##  [9,] "sports_playing" "dating"         "sports_fandom"  "home_and_garden"
## [10,] "current_events" "music"          "crafts"         "eco"
##       5
##  [1,] "health_nutrition"
##  [2,] "personal_fitness"
##  [3,] "outdoors"
##  [4,] "eco"
##  [5,] "food"
##  [6,] "cooking"
##  [7,] "dating"
##  [8,] "home_and_garden"
##  [9,] "crafts"
## [10,] "music"
```

```
kmeans_social5$size
```

```
## [1]  611 4839  712  787  933
```

With 5 clusters, we can see much stronger pattern in each group, which is a good segamentation of all users.

```
print(apply(kmeans_social8$centers,1,function(x) colnames(social_scaled)
            [order(x, decreasing=TRUE)[1:10]]))
```

```
##       1                2                3
##  [1,] "religion"       "online_gaming"  "shopping"
##  [2,] "parenting"      "college_uni"    "dating"
##  [3,] "sports_fandom"  "sports_playing" "current_events"
##  [4,] "food"           "art"            "home_and_garden"
##  [5,] "school"         "family"         "small_business"
##  [6,] "family"         "small_business" "business"
##  [7,] "crafts"         "tv_film"        "photo_sharing"
##  [8,] "beauty"         "home_and_garden" "eco"
##  [9,] "eco"            "automotive"     "music"
## [10,] "home_and_garden" "crafts"        "computers"
##       4                5                6
##  [1,] "health_nutrition" "cooking"       "tv_film"
##  [2,] "personal_fitness" "fashion"       "art"
##  [3,] "outdoors"         "beauty"        "music"
##  [4,] "eco"              "photo_sharing" "small_business"
##  [5,] "food"             "music"         "crafts"
##  [6,] "cooking"          "shopping"      "business"
##  [7,] "dating"           "business"      "college_uni"
##  [8,] "home_and_garden"  "school"        "current_events"
##  [9,] "business"         "small_business" "home_and_garden"
## [10,] "crafts"           "sports_playing" "travel"
##       7               8
##  [1,] "travel"        "news"
##  [2,] "politics"      "automotive"
```

```
##  [3,] "computers"      "politics"
##  [4,] "news"           "sports_fandom"
##  [5,] "business"       "outdoors"
##  [6,] "small_business" "family"
##  [7,] "dating"         "home_and_garden"
##  [8,] "crafts"         "current_events"
##  [9,] "eco"            "parenting"
## [10,] "food"           "school"
```

```
kmeans_social8$size
```

```
## [1]  727  375 4116  853  520  462  368  461
```

With 10 clusters, the patterns are weakened comparing to 5 clusters. It seems like we are over fitting the users into too many groups.

K-means++

We then try to use K-means++ to get a better initial points for K-means model.

```
kmeansPP_social = kmeanspp(social_scaled, 5)
print(apply(kmeansPP_social$centers,1,function(x) colnames(social_scaled)[order(x, decreasing=TRUE)[1:1
```

```
##       1                 2                 3
##  [1,] "politics"        "religion"        "fashion"
##  [2,] "news"            "parenting"       "cooking"
##  [3,] "travel"          "sports_fandom"   "beauty"
##  [4,] "computers"       "food"            "college_uni"
##  [5,] "automotive"      "school"          "online_gaming"
##  [6,] "business"        "family"          "photo_sharing"
##  [7,] "small_business"  "crafts"          "sports_playing"
##  [8,] "dating"          "beauty"          "music"
##  [9,] "sports_fandom"   "eco"             "tv_film"
## [10,] "home_and_garden" "home_and_garden" "art"
##       4                 5
##  [1,] "current_events"  "health_nutrition"
##  [2,] "shopping"        "personal_fitness"
##  [3,] "dating"          "outdoors"
##  [4,] "tv_film"         "eco"
##  [5,] "home_and_garden" "food"
##  [6,] "small_business"  "cooking"
##  [7,] "art"             "dating"
##  [8,] "business"        "home_and_garden"
##  [9,] "online_gaming"   "crafts"
## [10,] "automotive"      "business"
```

```
kmeansPP_social$size
```

```
## [1]  685  761 1146 4397  893
```

It turns out that the initial points we randomly picked is a very good point, and the result from K-means and K-means++ are very similar.

K-means vs K-means++

```
sum(kmeans_social5$withinss)
```

```
## [1] 193297.4
```

```
sum(kmeansPP_social$withinss)
```

```
## [1] 193598.3
```

Again, the result from K-means and K-means++ are very close. Note in this question, we're not trying to reduce the total within group SSE to a certain amount, rather we're trying to find a set of clusters that's most interpretable and meaningful. As shown above, when K=5, the groups has very strong demographical pattern, so we decide to use it as our cluster size.

Hierarchical Clustering

We also try out clustering using hierarchical clustering models. The drawback for hiercarchical clustering is that we need to pre-specify the linkage method and the dissimilarity measure.

Euclidean

We first try to quantify dissimilarity by euclidean distance. We test on all three linkage methods, and examine the clustering result.

```
social_distance_matrix = dist(social_scaled, method='euclidean')

hier_social = hclust(social_distance_matrix, method='average')
cluster1 = cutree(hier_social, k=5)

hier_social2 = hclust(social_distance_matrix, method='complete')
cluster2 = cutree(hier_social2, k=5)

hier_social3 = hclust(social_distance_matrix, method='single')
cluster3 = cutree(hier_social3, k=5)

clust.centroid = function(i, dat, clusters) {
  ind = (clusters == i)
  colMeans(dat[ind,])
}
clust.center = function(i, center){
  row.names(center[order(center[,i], decreasing = TRUE),][1:10,])
}
```

Top ten features from each group under each linkage method are shown below.

```
center1 = sapply(unique(cluster1), clust.centroid, social_clean, cluster1)
sapply(unique(cluster1), clust.center, center1)
```

```
##       [,1]               [,2]             [,3]
##  [1,] "photo_sharing"    "cooking"        "politics"
##  [2,] "health_nutrition" "photo_sharing" "travel"
##  [3,] "cooking"          "fashion"        "computers"
##  [4,] "politics"         "beauty"         "photo_sharing"
##  [5,] "sports_fandom"    "food"           "news"
```

```
##  [6,] "travel"           "shopping"      "cooking"
##  [7,] "college_uni"      "parenting"     "shopping"
##  [8,] "current_events"   "sports_fandom" "fashion"
##  [9,] "personal_fitness" "politics"      "beauty"
## [10,] "food"             "family"        "food"
##       [,4]               [,5]
##  [1,] "art"              "politics"
##  [2,] "tv_film"          "health_nutrition"
##  [3,] "college_uni"      "travel"
##  [4,] "health_nutrition" "dating"
##  [5,] "crafts"           "news"
##  [6,] "photo_sharing"    "school"
##  [7,] "travel"           "computers"
##  [8,] "online_gaming"    "art"
##  [9,] "shopping"         "religion"
## [10,] "personal_fitness" "small_business"
```

```
summary(factor(cluster1))
```

```
##    1    2    3    4    5
## 7857    5   10    9    1
```

```
center2 = sapply(unique(cluster2), clust.centroid, social_clean, cluster2)
sapply(unique(cluster2), clust.center, center2)
```

```
##       [,1]               [,2]               [,3]
##  [1,] "health_nutrition" "art"              "sports_fandom"
##  [2,] "photo_sharing"    "tv_film"          "religion"
##  [3,] "politics"         "photo_sharing"    "food"
##  [4,] "cooking"          "travel"           "parenting"
##  [5,] "sports_fandom"    "health_nutrition" "school"
##  [6,] "college_uni"      "food"             "family"
##  [7,] "travel"           "current_events"   "health_nutrition"
##  [8,] "current_events"   "cooking"          "photo_sharing"
##  [9,] "personal_fitness" "politics"         "politics"
## [10,] "shopping"         "shopping"         "travel"
##       [,4]               [,5]
##  [1,] "cooking"          "politics"
##  [2,] "fashion"          "travel"
##  [3,] "photo_sharing"    "computers"
##  [4,] "beauty"           "news"
##  [5,] "health_nutrition" "dating"
##  [6,] "politics"         "food"
##  [7,] "shopping"         "religion"
##  [8,] "travel"           "photo_sharing"
##  [9,] "current_events"   "health_nutrition"
## [10,] "college_uni"      "parenting"
```

```
summary(factor(cluster2))
```

```
##    1    2    3    4    5
## 7438  165   39  225   15
```

```
center3 = sapply(unique(cluster3), clust.centroid, social_clean, cluster1)
sapply(unique(cluster3), clust.center, center3)
```

```
##       [,1]               [,2]            [,3]
```

```
##  [1,] "photo_sharing"    "cooking"       "politics"
##  [2,] "health_nutrition" "photo_sharing" "travel"
##  [3,] "cooking"          "fashion"       "computers"
##  [4,] "politics"         "beauty"        "photo_sharing"
##  [5,] "sports_fandom"    "food"          "news"
##  [6,] "travel"           "shopping"      "cooking"
##  [7,] "college_uni"      "parenting"     "shopping"
##  [8,] "current_events"   "sports_fandom" "fashion"
##  [9,] "personal_fitness" "politics"      "beauty"
## [10,] "food"             "family"        "food"
##       [,4]               [,5]
##  [1,] "art"              "politics"
##  [2,] "tv_film"          "health_nutrition"
##  [3,] "college_uni"      "travel"
##  [4,] "health_nutrition" "dating"
##  [5,] "crafts"           "news"
##  [6,] "photo_sharing"    "school"
##  [7,] "travel"           "computers"
##  [8,] "online_gaming"    "art"
##  [9,] "shopping"         "religion"
## [10,] "personal_fitness" "small_business"
```

```
summary(factor(cluster3))
```

```
##    1    2    3    4    5
## 7878    1    1    1    1
```

We can see that all three clustering are not having clear patterns in each group. Among three linkage method, when we use `complete`, the result are more interpretable.

Correlation-based distance

We then try to quantify dissimilarity by correlation. We expect this measure to yield a better result since some user tends to post tweet more often than other users. So the similarity should be quantify in the sense of relative frequency for each user instead of absolute frequency.

```
social_cor = cor(t(social_scaled), method = "pearson")
social_dis = as.dist(1 - social_cor)
hier_social.cor = hclust(social_dis, method='ward.D2')
cluster.cor = cutree(hier_social.cor, k=5)

center4 = sapply(unique(cluster.cor), clust.centroid, social_clean, cluster.cor)
sapply(unique(cluster.cor), clust.center, center4)
```

```
##       [,1]               [,2]               [,3]
##  [1,] "health_nutrition" "sports_fandom"    "photo_sharing"
##  [2,] "personal_fitness" "religion"         "college_uni"
##  [3,] "cooking"          "food"             "shopping"
##  [4,] "outdoors"         "parenting"        "online_gaming"
##  [5,] "photo_sharing"    "photo_sharing"    "current_events"
##  [6,] "food"             "school"           "tv_film"
##  [7,] "current_events"   "family"           "travel"
##  [8,] "travel"           "current_events"   "health_nutrition"
##  [9,] "shopping"         "health_nutrition" "politics"
## [10,] "politics"         "cooking"          "cooking"
```

```
##       [,4]               [,5]
##  [1,] "politics"         "cooking"
##  [2,] "news"             "photo_sharing"
##  [3,] "travel"           "fashion"
##  [4,] "automotive"       "beauty"
##  [5,] "photo_sharing"    "health_nutrition"
##  [6,] "sports_fandom"    "current_events"
##  [7,] "computers"        "shopping"
##  [8,] "health_nutrition" "travel"
##  [9,] "current_events"   "college_uni"
## [10,] "cooking"          "personal_fitness"
```

```
summary(factor(cluster.cor))
```

```
##    1    2    3    4    5
## 1079 1131 3897 1094  681
```

We can see that using correlation as dissimilarity measure, clusters now have stronger pattern comparing to using euclidean distance. However, comparing to k-means, there still exists some ambiguity among groups. Therefore, we decide to use result from K-means clustering with 5 clusters for further analysis.

Market Segments and Insight

Market Segments

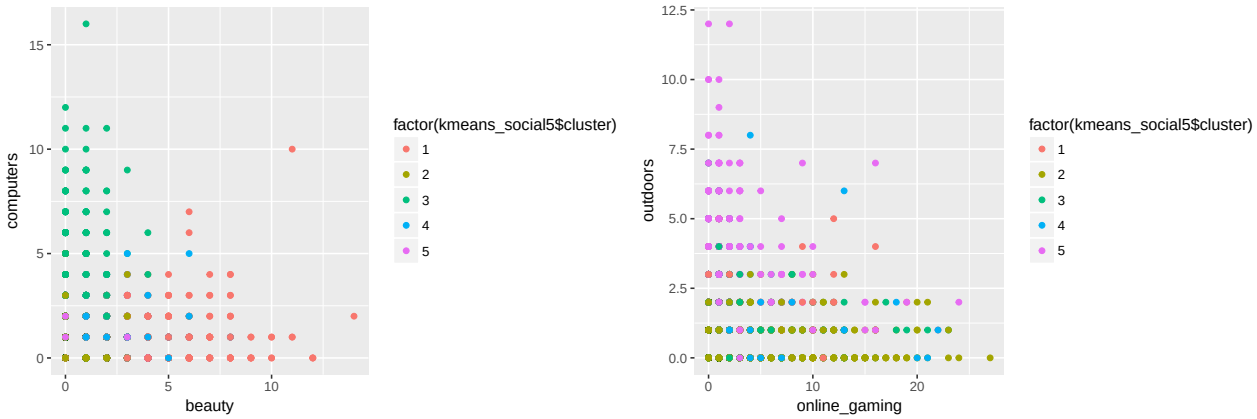
Top ten features from K-means clustering with 5 clusters are shown below.

```
center = apply(kmeans_social5$centers,1,function(x) colnames(social_scaled)
               [order(x, decreasing=TRUE)[1:10]])
groups = cbind(center[,1],center[,2],center[,3],center[,4],center[,5])
colnames(groups) = c(1,2,3,4,5)
pander(groups, split.cells = c(8,8,8,8,8))
```

1	2	3	4	5
cooking	college_uni	politics	religion	health_nutrition
fashion	online_gaming	news	parenting	personal_fitness
beauty	tv_film	travel	sports_fandom	outdoors
photo_sharing	art	computers	food	eco
music	current_events	automotive	school	food
shopping	shopping	business	family	cooking
small_business	small_business	small_business	crafts	dating
business	sports_playing	dating	beauty	home_and_garden
sports_playing	dating	sports_fandom	home_and_garden	crafts
current_events	music	crafts	eco	music

We also make two plots with two pairs of distinct features to show the clustering result.

```
compare1 = qplot(beauty, computers, data = social_clean,
                 color = factor(kmeans_social5$cluster))
compare2 = qplot(online_gaming, outdoors, data = social_clean,
                 color = factor(kmeans_social5$cluster))

grid.arrange(compare1, compare2, ncol = 2)
```



As shown above, users in group 3 have higher interest in computers whereas people in group 1 have more interest in beauty. Also, users in group 5 have stronger interest in outdoor activities whereas people in group 2 have more interest in online gaming.

Insight with AIO Model

Lastly, we charaterize each group by activity-interest-opinion (AIO) model, and make predictions on potential demographic for each group.

```
group1 = c('entertainments, social events',
           'fashion, recreation',
           'themselves, product',
           'young to mid-age female')
group2 = c('entertainment, hobbies',
           'recreation, fashion',
           'themselves, product',
           'college students')
group3 = c('work, social events',
           'technology, recreation',
           'politics, business',
           'mid-age male')
group4 = c('community, hobbies',
           'family, community',
           'themselves, education',
           'housewife')
group5 = c('hobbies, community',
           'health, food',
           'themselves, product',
           'people with healthy lifestyle')
size = kmeans_social5$size
number = c(1,2,3,4,5)
segment = data.frame(cbind(number, size,rbind(group1, group2, group3, group4, group5)))

colnames(segment) = c('group','size', 'activity', 'interest', 'opinion', 'potential')
rownames(segment) = NULL
pander(segment, split.cells = c(5,5,15,15,15,25))
```

group	size	activity	interest	opinion	potential
1	611	entertainments, social events	fashion, recreation	themselves, product	young to mid-age female

group	size	activity	interest	opinion	potential
2	4839	entertainment, hobbies	recreation, fashion	themselves, product	college students
3	712	work, social events	technology, recreation	politics, business	mid-age male
4	787	community, hobbies	family, community	themselves, education	housewife
5	933	hobbies, community	health, food	themselves, product	people with healthy lifestyle