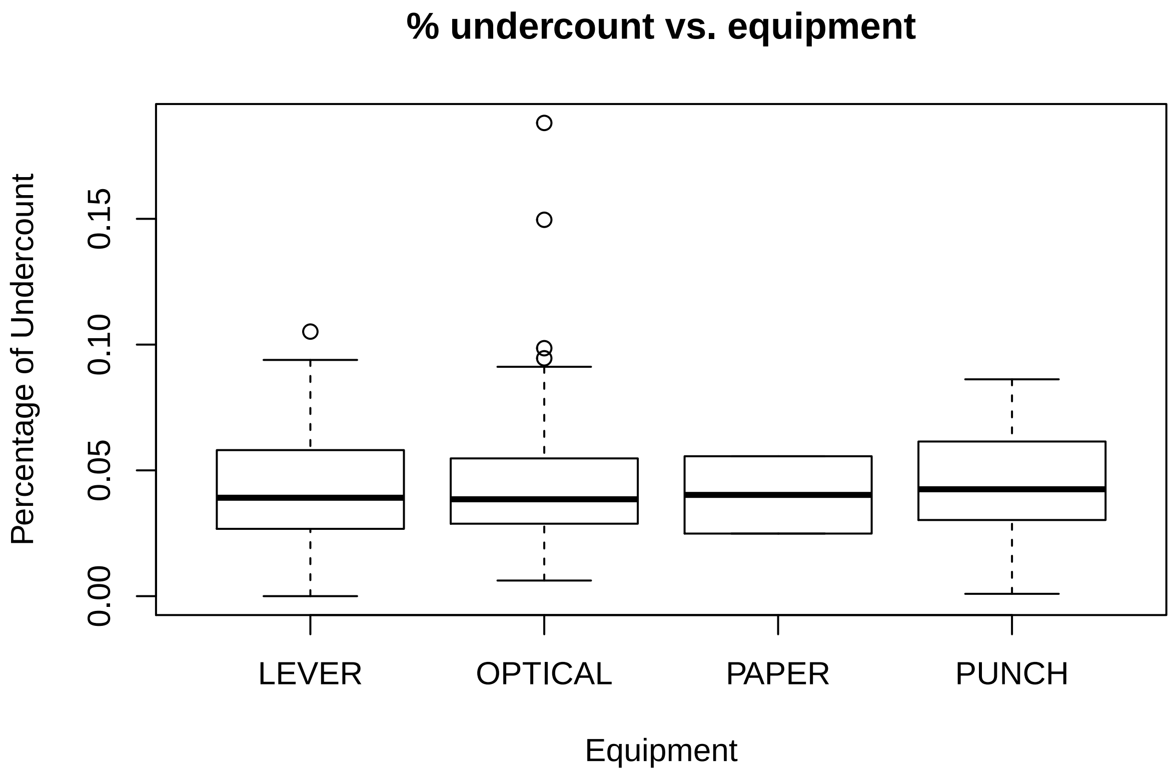
output: pdf_document

```
# import data
setwd("C:\\Users\\Daniel\\Documents\\GitHub\\STA380\\data")
georgia = read.csv("georgia2000.csv")
```

```
# calculate the percentage of undercount
georgia$percent_undercount <- (georgia$ballots - georgia$votes)/georgia$ballots
boxplot(percent_undercount~equip,data=georgia, main="% undercount vs. equipment", xlab="Equipment", ylab
```



4 outliers effect optical as median percentage of undercount are similiar across the board.

```
#names(georgia)
georgia=georgia[,c(-1,-2,-3)]

lm.georgia = lm(percent_undercount~.,data=georgia)
summary(lm.georgia)
```

```
##
## Call:
## lm(formula = percent_undercount ~ ., data = georgia)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.064135 -0.011683 -0.001676  0.010158  0.123612
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   3.119e-02  4.483e-03   6.957 1.02e-10 ***
## equipOPTICAL  1.381e-02  4.049e-03   3.412 0.000831 ***
## equipPAPER   -1.070e-02  1.592e-02  -0.672 0.502531    
## equipPUNCH    1.424e-02  6.694e-03   2.128 0.035012 *  
## poor          1.938e-02  4.785e-03   4.050 8.21e-05 ***
## urban        -7.116e-03  5.078e-03  -1.401 0.163206    
## atlanta      -6.872e-03  8.402e-03  -0.818 0.414724    
## perAA         1.079e-03  1.407e-02   0.077 0.938990    
## gore          2.701e-07  1.732e-07   1.559 0.121092    
## bush         -3.275e-07  1.983e-07  -1.652 0.100671    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02206 on 149 degrees of freedom
## Multiple R-squared:  0.2635, Adjusted R-squared:  0.219 
## F-statistic: 5.922 on 9 and 149 DF,  p-value: 4.657e-07
```

```
lm.georgia = lm(percent_undercount~.*poor,data=georgia)
summary(lm.georgia)
```

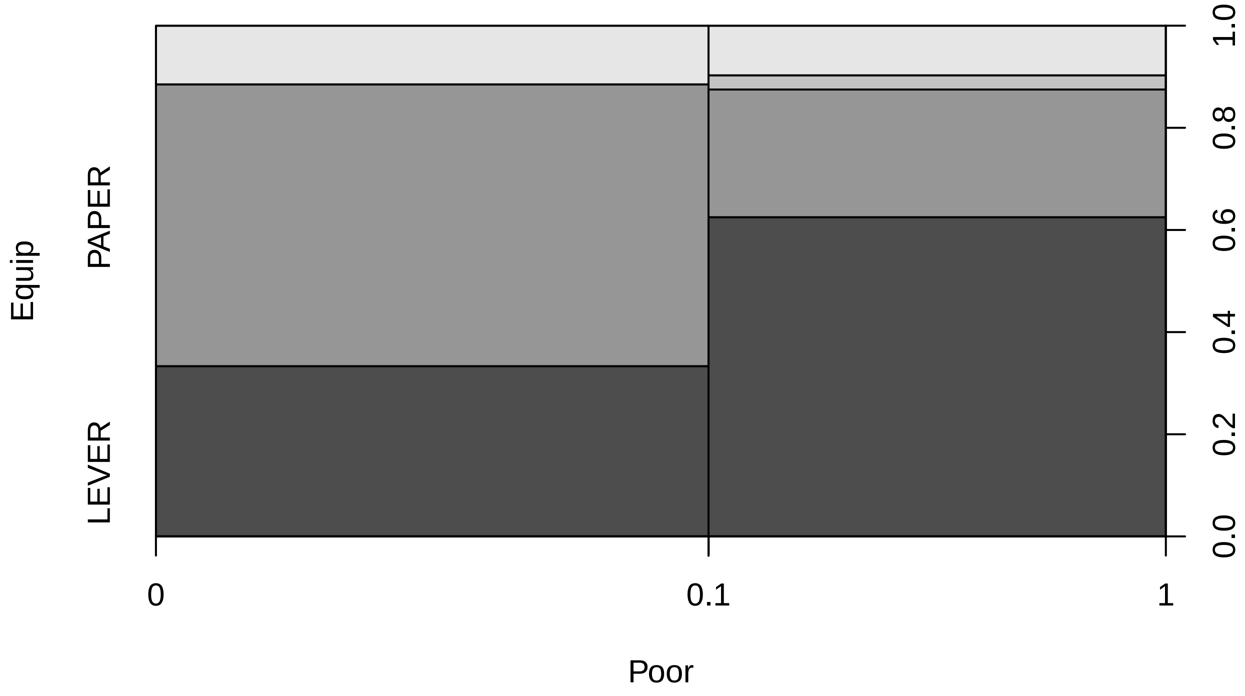
```
##
## Call:
## lm(formula = percent_undercount ~ . * poor, data = georgia)
##
## Residuals:
##       Min        1Q    Median        3Q       Max 
## -0.061540 -0.010739 -0.001310  0.009586  0.112128 
##
## Coefficients: (2 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)    
## (Intercept)        3.463e-02  5.817e-03   5.953 1.94e-08 ***
## equipOPTICAL       5.868e-03  5.422e-03   1.082   0.2809    
## equipPAPER        -1.066e-02  1.596e-02  -0.668   0.5053    
## equipPUNCH         1.351e-02  9.856e-03   1.371   0.1725    
## poor               2.313e-02  1.197e-02   1.933   0.0553 .  
## urban             -6.630e-03  5.650e-03  -1.173   0.2426    
## atlanta           -7.657e-03  8.591e-03  -0.891   0.3743    
## perAA              3.399e-03  2.208e-02   0.154   0.8779    
## gore               2.007e-07  1.889e-07   1.062   0.2898    
## bush              -2.334e-07  2.034e-07  -1.147   0.2533    
## equipOPTICAL:poor  2.049e-02  8.262e-03   2.479   0.0143 *  
## equipPAPER:poor           NA         NA      NA       NA    
## equipPUNCH:poor   -2.136e-03  1.344e-02  -0.159   0.8740    
## poor:urban        -6.984e-03  1.440e-02  -0.485   0.6284    
## poor:atlanta              NA         NA      NA       NA    
## poor:perAA        -1.553e-02  3.286e-02  -0.473   0.6372    
## poor:gore          1.707e-06  3.460e-06   0.493   0.6226    
## poor:bush         -3.532e-06  3.400e-06  -1.039   0.3007    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.0218 on 143 degrees of freedom
## Multiple R-squared:  0.3097, Adjusted R-squared:  0.2373
## F-statistic: 4.276 on 15 and 143 DF,  p-value: 1.587e-06
```

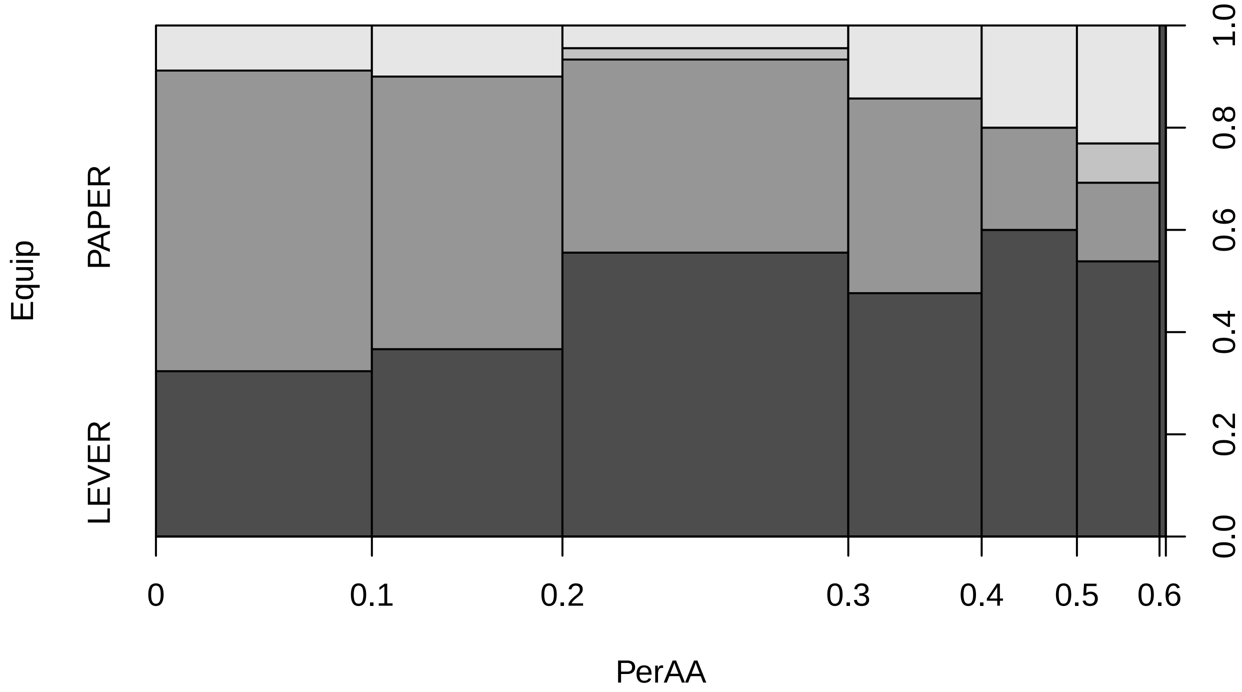
punch cards and optical scans correlate with higher undercount percentages due to outliers in boxplot and whether or not someone is poor is also associated significantly.When interacting with whether poor interacted with the data, it showed that the most significant influence on undercount happens to be from whether someone was poor interacted with optical equipment.

so now we investigate whether the poor and AAs have been using equipment that results in higher undercount.

```
# plot equip against poor, perAA, and urban
#xtabs(~georgia$equip+georgia$poor)
plot(georgia$equip~georgia$poor, xlab = "Poor", ylab = "Equip")
```



```
#xtabs(~georgia$equip+georgia$perAA)
plot(georgia$equip~georgia$perAA, xlab = "PerAA", ylab = "Equip")
```



we found that people who weren't poor primarily used optical and punch which generated more undercounting and that the percentage of african americans had no strong relationship with equipment used.

part2:

```
# import libraries
library(mosaic)
library(fImport)
library(foreach)
```

```
# import five years of daily data on ETFs SPY, TLT, LQD, EEM, and VNQ
funds = c("SPY", "TLT", "LQD", "EEM", "VNQ")
prices = yahooSeries(funds, from='2009-01-01', to='2014-12-31')

# add helper function for calculating percent returns from Yahoo Series
YahooPricesToReturns = function(series) {
    mycols = grep('Adj.Close', colnames(series))
    closingprice = series[,mycols]
    N = nrow(closingprice)
    percentreturn = as.data.frame(closingprice[2:N,]) / as.data.frame(closingprice[1:(N-1),]) - 1
    mynames = strsplit(colnames(percentreturn), '.', fixed=TRUE)
    mynames = lapply(mynames, function(x) return(paste0(x[1], ".PctReturn")))
    colnames(percentreturn) = mynames
    as.matrix(na.omit(percentreturn))
}

# compute the returns from the closing prices
```

```
returns = YahooPricesToReturns(prices)
head(returns,5)
```

```
##            SPY.PctReturn TLT.PctReturn LQD.PctReturn EEM.PctReturn
## 2009-01-05  -0.001183312 -0.0257842738   0.006858200   0.012996911
## 2009-01-06   0.006677471 -0.0100573375   0.001678158   0.022641531
## 2009-01-07  -0.029956170  0.0039212366  -0.008376838  -0.057564545
## 2009-01-08   0.004080768 -0.0007989698   0.007155565  -0.004306995
## 2009-01-09  -0.021419213  0.0015103609   0.012433419  -0.021628040
##            VNQ.PctReturn
## 2009-01-05  -0.018418865
## 2009-01-06   0.049942257
## 2009-01-07  -0.033544048
## 2009-01-08  -0.007396935
## 2009-01-09  -0.051304138
```

```
# plot returns for each ETF and assess risk and return
#plot(returns[,1], type='l',main='SPY')
#plot(returns[,2], type='l',main='TLT')
#plot(returns[,3], type='l',main='LQD')
#plot(returns[,4], type='l',main='EEM')
#plot(returns[,5], type='l',main='VNQ')
# mean(returns[,1])
#sd(returns[,1])
```

RISKY: VNQ and EEM have the biggest sd but EEM has a lower mean return then SPY. MEDIUM:SPY and TLT are around the same sd. SAFE: Lastly LQD has almost no sd with a stead mean return higher than TLT.

“‘ 20 day return simulated using bootstrap for an 20% even split in portfolio amongst all ETF: ended at 103766.74

```
# Perform bootstrap 5000 times for even split portfolio
n_days=20
set.seed(111)
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
    wealthtracker = rep(0, n_days)
    for(today in 1:n_days) {
        return.today = resample(returns, 1, orig.ids=FALSE)
        holdings = holdings + holdings*return.today
        totalwealth = sum(holdings)
        wealthtracker[today] = totalwealth
        holdings = weights * totalwealth
    }
    wealthtracker
}
```

20 day return simulated using bootstrap for a safe portfolio on three assets: ended at 101846.04. Chose the ETFs that were medium to safe that had high mean returns.

```r
# Perform bootstrap 5000 times
set.seed(111)
sim_safe = foreach(i=1:5000, .combine='rbind') %do% {
    totalwealth = 100000
    weights = c(0.5, 0.1, 0.4, 0.0, 0.0)
    holdings = weights * totalwealth
    wealthtracker = rep(0, n_days)
    for(today in 1:n_days) {
        return.today = resample(returns, 1, orig.ids=FALSE)
        holdings = holdings + holdings*return.today
        totalwealth = sum(holdings)
        wealthtracker[today] = totalwealth
        holdings = weights * totalwealth
    }
    wealthtracker
}
```

20 day return simulated using bootstrap for a risky portfolio on two assets: ended at 109643.77. Chose the ETFs that were risky.

```r
# Perform bootstrap 5000 times
set.seed(111)
sim_risky = foreach(i=1:5000, .combine='rbind') %do% {
    totalwealth = 100000
    weights = c(0.0, 0.0, 0.0, 0.3, 0.7)
    holdings = weights * totalwealth
    wealthtracker = rep(0, n_days)
    for(today in 1:n_days) {
        return.today = resample(returns, 1, orig.ids=FALSE)
        holdings = holdings + holdings*return.today
        totalwealth = sum(holdings)
        wealthtracker[today] = totalwealth
        holdings = weights * totalwealth
    }
    wealthtracker
}
```

You get higher profit at a higher risk. Returns ranked from risky, even, to safe. Risk at 5% ranked from risky, even, to safe. Furthermore you can see that the variation of each distribution of profit/Loss ranked from risky, even, to safe.

```r
# risk at 5% level for each portfolio
quantile(sim_even[,n_days], 0.05) - 100000
```

```
##       5%
## -5238.05
```

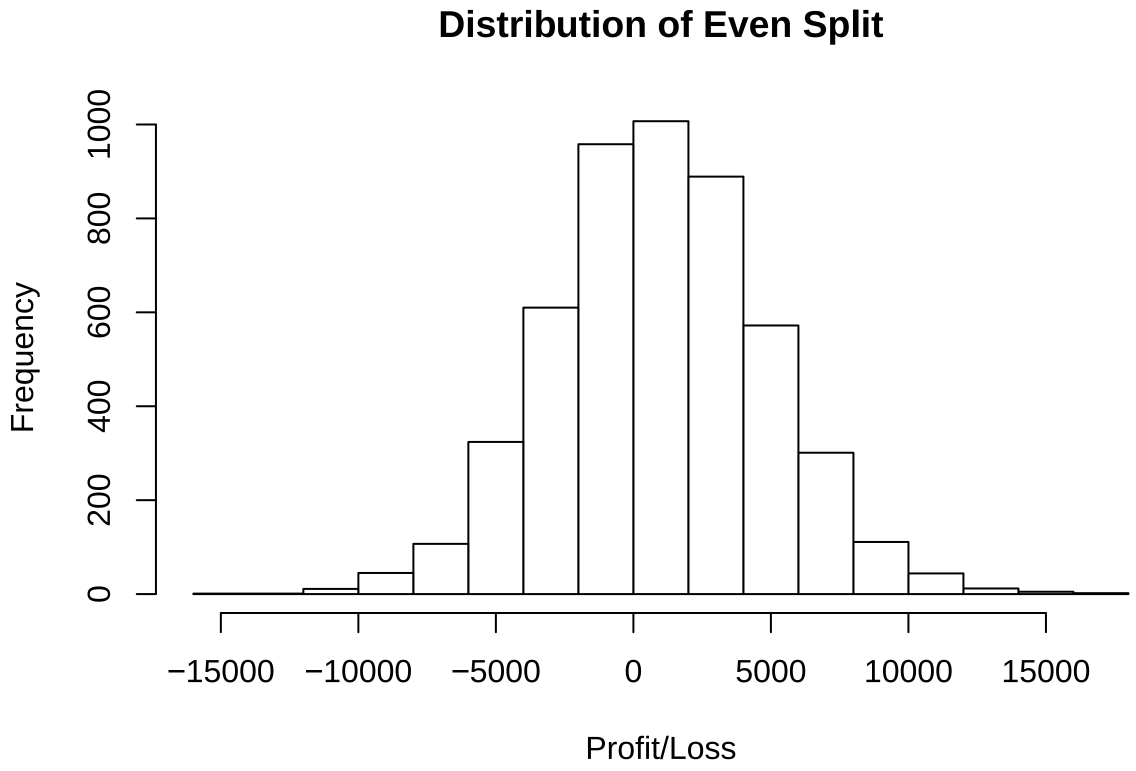
```r
quantile(sim_safe[,n_days], 0.05) - 100000
```

```
##       5%
## -3303.91
```

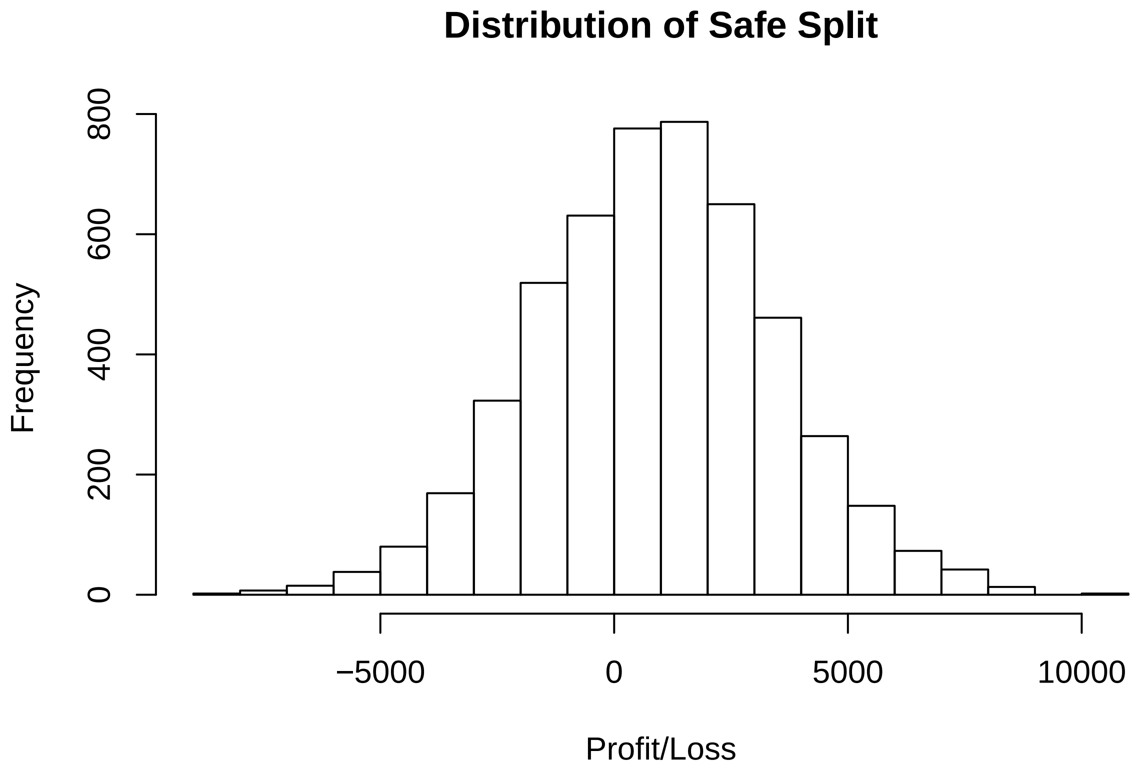
```r
quantile(sim_risky[,n_days], 0.05) - 100000
```

```
##        5%
## -11655.57
```

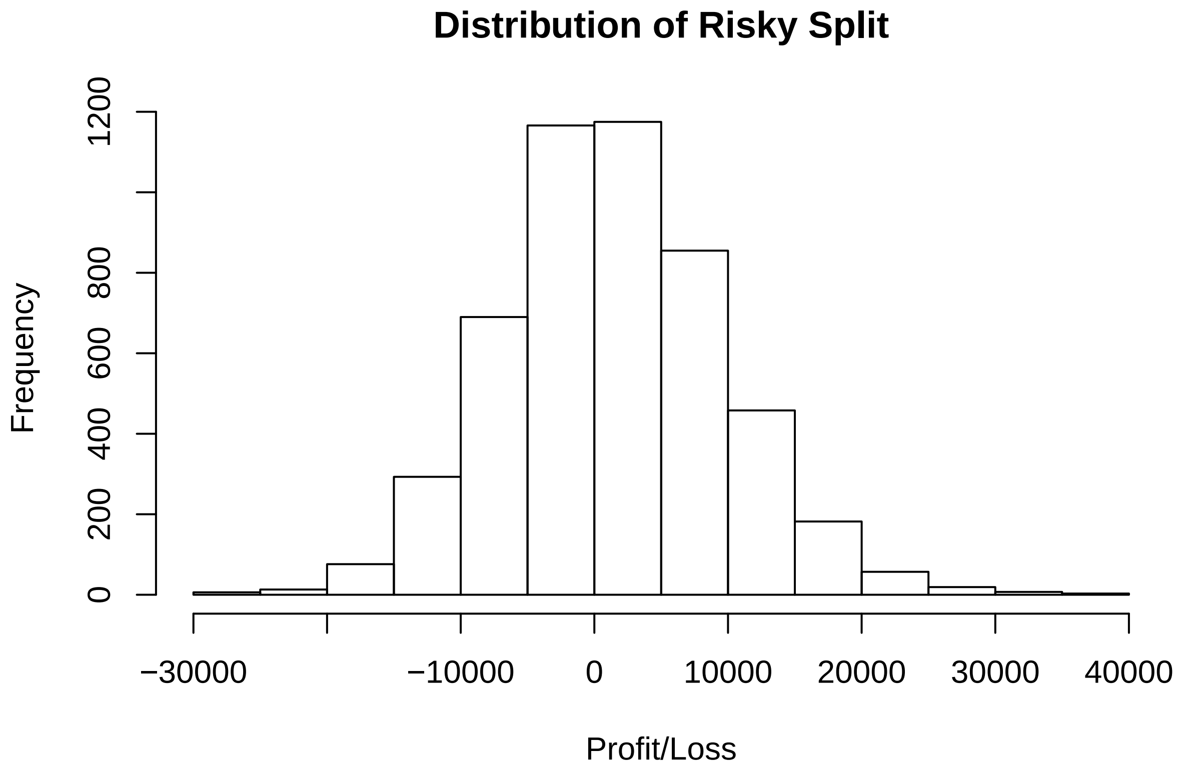
```r
hist(sim_even[,n_days]- 100000,xlab='Profit/Loss',main='Distribution of Even Split')
```



```r
hist(sim_safe[,n_days]- 100000,xlab='Profit/Loss',main='Distribution of Safe Split')
```



```
hist(sim_risky[,n_days]- 100000,xlab='Profit/Loss',main='Distribution of Risky Split')
```



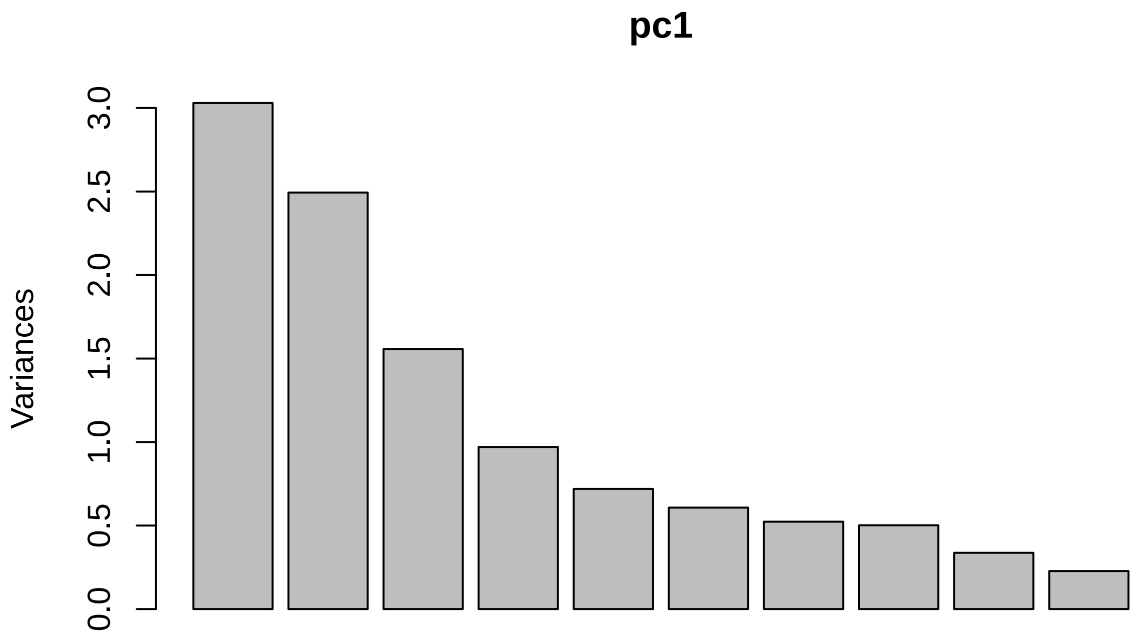
---

part 3:

PCA

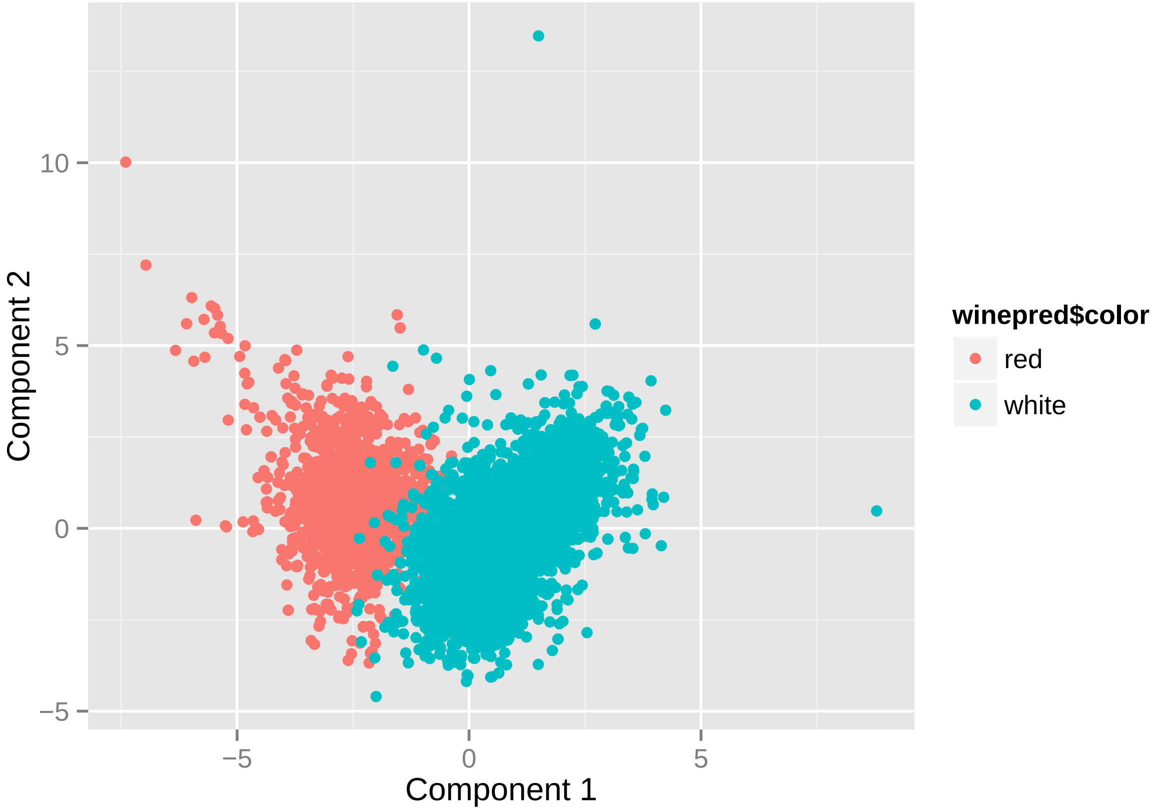
```
library(ggplot2)
setwd("C:\\Users\\Daniel\\Documents\\GitHub\\STA380\\data")
wine = read.csv("wine.csv")
wineclus = wine[,c(-13,-12)]
winepred = wine[,c(12,13)]
wineclus = scale(wineclus, center=TRUE)
pc1 = prcomp(wineclus, scale.=TRUE)

# Look at the basic plotting and summary methods
#pc1
#summary(pc1)
plot(pc1)
#biplot(pc1)
plot(pc1)
```

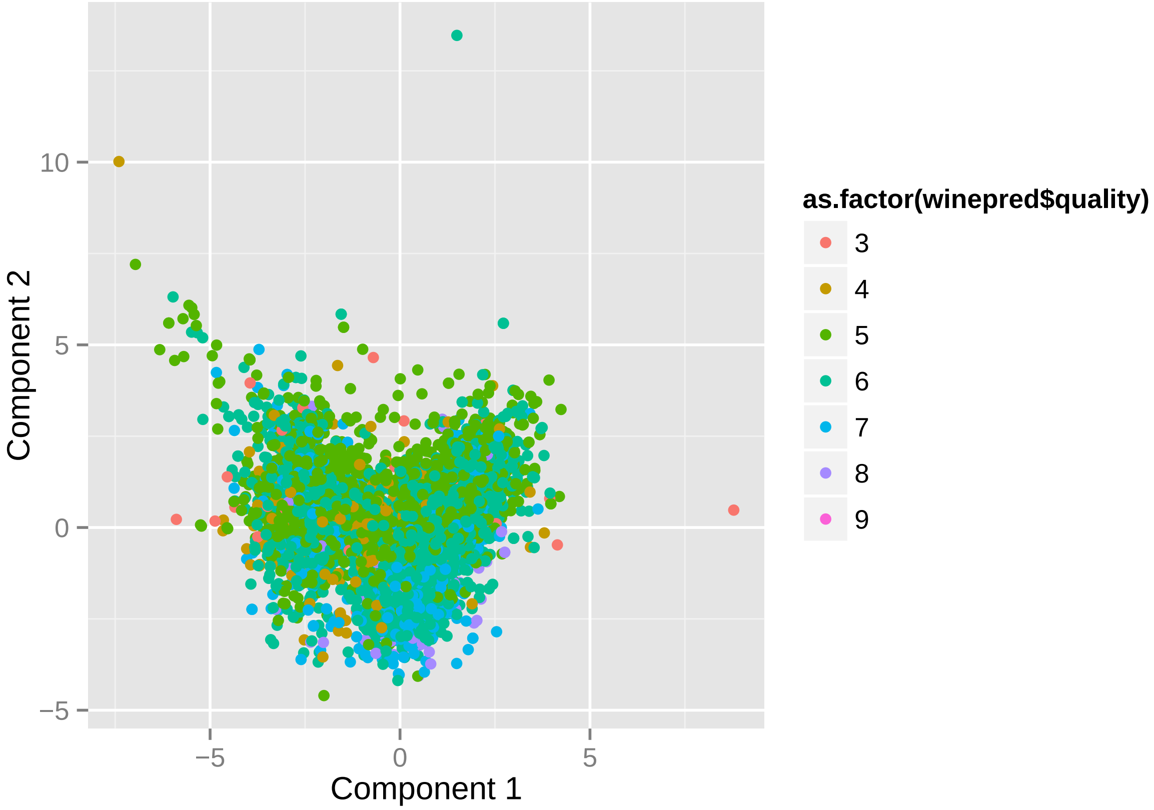


```
# A more informative biplot
loadings = pc1$rotation
scores = pc1$x

qplot(scores[,1], scores[,2] ,color=winepred$color, xlab='Component 1', ylab='Component 2')
```



```
qplot(scores[,1], scores[,2] ,color=as.factor(winepred$quality), xlab='Component 1', ylab='Component 2'
```



hierarchical

```
wineclus =  scale(wineclus, center=TRUE, scale=TRUE)
# First form a pairwise distance matrix
distance_between_wines= dist(wineclus)
# Now run hierarchical clustering
h1 = hclust(distance_between_wines, method='complete')
# Cut the tree into 10 clusters
cluster1 = cutree(h1, k=10)
#summary(factor(cluster1))
# Examine the cluster members
ind1 = which(cluster1 == 2)
ind2 = which(cluster1 == 6)

# find distribution of wine color within each cluster
table(winepred$color[ind1])
```

```
##
##   red white
##   554  3345
```

```
table(winepred$color[ind2])
```

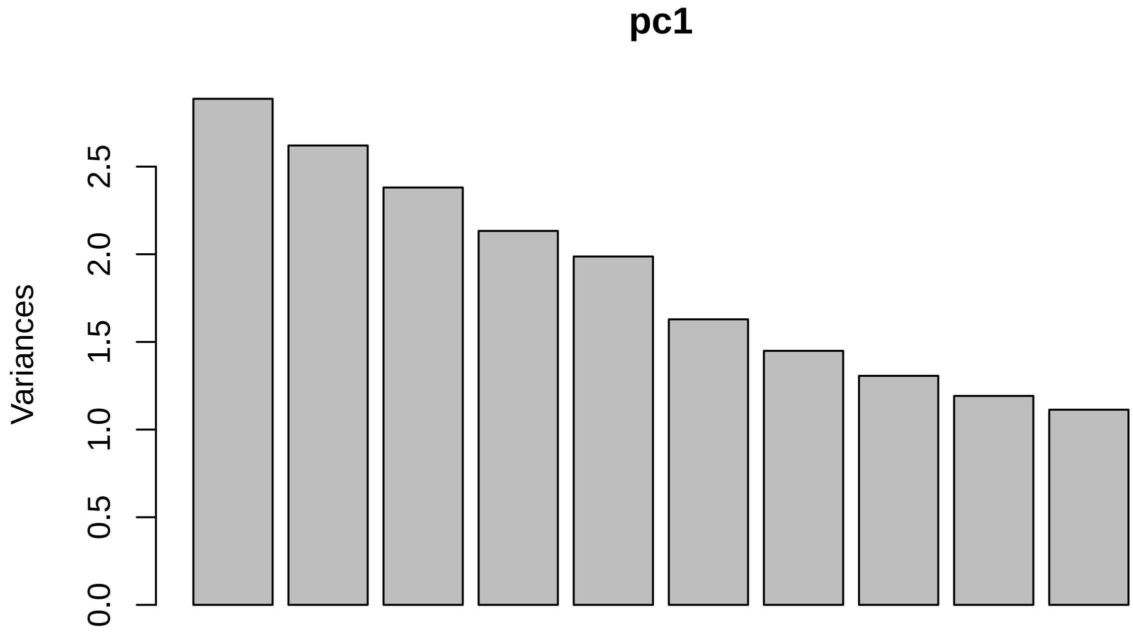
```
##
##   red white
##     8  1440
```

It makes more sense to use PCA since you only have to look at the first two PC loadings instead of sifting through each cluster when using hierarchical clustering. Wine quality isn't easily predicted using the dataset and can be easily shown by the qplot for wine quality above where you get quality index jumbled in each principal component.

part 4:

```
setwd("C:\\Users\\Daniel\\Documents\\GitHub\\STA380\\data")
social_marketing = read.csv("social_marketing.csv",header=TRUE, row.names=1)
social_marketing = social_marketing/rowSums(social_marketing)
pc1 = prcomp(social_marketing,scale. = TRUE)

# Look at the basic plotting and summary methods
#pc1
#summary(pc1)
plot(pc1)
```



```r
# A more informative biplot
loadings = pc1$rotation
scores = pc1$x
#since pc8 on variance is close to 1, clusters can be ignored
o1 = order(loadings[,1])
loadings[o1,1]
```

```
##    photo_sharing          chatter         shopping          cooking 
##     -0.235240727     -0.197589933     -0.178786654     -0.166888146 
##          fashion      college_uni    uncategorized    online_gaming 
##     -0.135348439     -0.087266061     -0.077233767     -0.070521043 
## health_nutrition personal_fitness   current_events           beauty 
##     -0.066831428     -0.063658685     -0.060891784     -0.056993344 
##   sports_playing            music         business          tv_film 
##     -0.056213619     -0.053914297     -0.041782821     -0.033405667 
##              eco   small_business  home_and_garden           dating 
##     -0.030835275     -0.030160417     -0.022090719     -0.019189393 
##         outdoors           travel              art             spam 
##     -0.016830745     -0.013234532     -0.009080086      0.002906308 
##            adult        computers         politics           crafts 
##      0.014152299      0.016086992      0.039561780      0.071710384 
##       automotive             news           family           school 
##      0.080843006      0.098260858      0.237001589      0.273059942 
##             food        parenting    sports_fandom         religion 
##      0.345591934      0.402533960      0.408035827      0.423334362
```

```r
#colnames(social_marketing)[head(o1,5)]
colnames(social_marketing)[tail(o1,5)]
```

```
## [1] "school"        "food"          "parenting"     "sports_fandom"
## [5] "religion"
```

```r
o2 = order(loadings[,2])
#loadings[o2,2]
#colnames(social_marketing)[head(o2,5)]
colnames(social_marketing)[tail(o2,5)]
```

```
## [1] "automotive" "shopping"   "travel"     "politics"   "chatter"
```

```r
o3 = order(loadings[,3])
#loadings[o3,3]
#colnames(social_marketing)[head(o3,5)]
colnames(social_marketing)[tail(o3,5)]
```

```
## [1] "health_nutrition" "outdoors"         "travel"          
## [4] "news"             "politics"
```

```r
o4 = order(loadings[,4])
#loadings[o4,4]
#colnames(social_marketing)[head(o4,5)]
colnames(social_marketing)[tail(o4,5)]
```

```
## [1] "news"          "politics"      "chatter"       "shopping"
## [5] "photo_sharing"
```

```
o5 = order(loadings[,5])
#loadings[o5,5]
#colnames(social_marketing)[head(o5,5)]
colnames(social_marketing)[tail(o5,5)]
```

```
## [1] "news"     "politics" "cooking"  "beauty"   "fashion"
```

```
o6 = order(loadings[,6])
#loadings[o6,6]
#colnames(social_marketing)[head(o6,5)]
colnames(social_marketing)[tail(o6,5)]
```

```
## [1] "photo_sharing"  "college_uni"    "sports_playing" "automotive"
## [5] "online_gaming"
```

```
o7 = order(loadings[,7])
#loadings[o7,7]
#colnames(social_marketing)[head(o7,5)]
colnames(social_marketing)[tail(o7,5)]
```

```
## [1] "music"      "art"        "tv_film"    "news"       "automotive"
```

```
o8 = order(loadings[,8])
#loadings[o8,8]
#colnames(social_marketing)[head(o8,5)]
colnames(social_marketing)[tail(o8,5)]
```

```
## [1] "shopping"         "health_nutrition" "business"
## [4] "tv_film"          "music"
```

Your biggest group of supporters love sportsfandom, religion, and parenting from the PCA study.

```
##  [1]  0.156703769  1.373811191  0.730670244 -1.350800927 -0.008514961
##  [6]  0.320981863 -1.778148409  0.909503835 -0.919404336 -0.157714831
```

```
##  [1]  0.156703769  1.373811191  0.730670244 -1.350800927 -0.008514961
##  [6]  0.320981863 -1.778148409  0.909503835 -0.919404336 -0.157714831
```