CS544\_HW3\_Escandon

Phillip Escandon - [escandon@bu.edu](mailto:escandon@bu.edu)

07 February, 2021

Table of Contents

## Part 1. Read in some prime number data, plot and draw inferences.

df <- read.csv("http://people.bu.edu/kalathur/datasets/myPrimes.csv")

dim(df)

## [1] 1229 3

colnames(df)

## [1] "Prime" "LastDigit" "FirstDigit"

head(df)

## Prime LastDigit FirstDigit  
## 1 2 2 2  
## 2 3 3 3  
## 3 5 5 5  
## 4 7 7 7  
## 5 11 1 1  
## 6 13 3 1

tail(df)

## Prime LastDigit FirstDigit  
## 1224 9929 9 9  
## 1225 9931 1 9  
## 1226 9941 1 9  
## 1227 9949 9 9  
## 1228 9967 7 9  
## 1229 9973 3 9

summary(df)

## Prime LastDigit FirstDigit   
## Min. : 2 Min. :1.000 Min. :1.00   
## 1st Qu.:2029 1st Qu.:3.000 1st Qu.:3.00   
## Median :4523 Median :3.000 Median :5.00   
## Mean :4668 Mean :4.985 Mean :4.82   
## 3rd Qu.:7213 3rd Qu.:7.000 3rd Qu.:7.00   
## Max. :9973 Max. :9.000 Max. :9.00

dplyr::count(df,df$First)

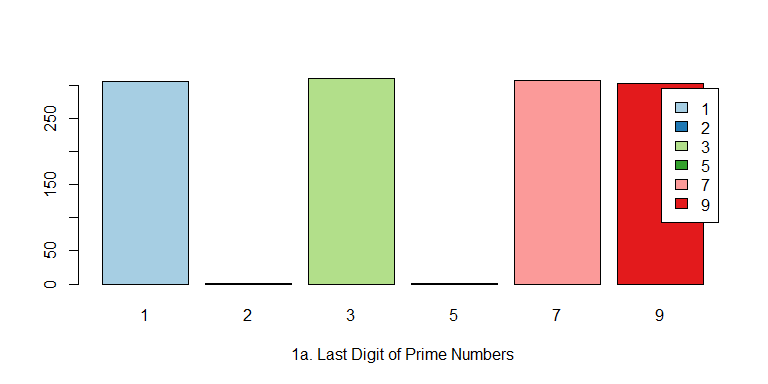
## df$First n  
## 1 1 160  
## 2 2 146  
## 3 3 139  
## 4 4 139  
## 5 5 131  
## 6 6 135  
## 7 7 125  
## 8 8 127  
## 9 9 127

dplyr::count(df,df$Last)

## df$Last n  
## 1 1 306  
## 2 2 1  
## 3 3 310  
## 4 5 1  
## 5 7 308  
## 6 9 303

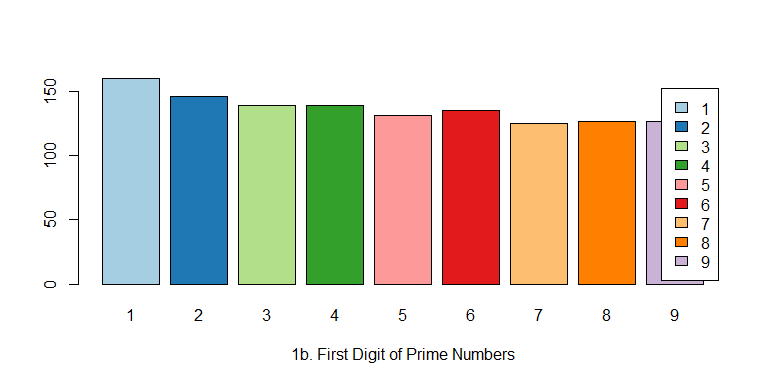
### 1a. Show a barplot of the last digit

#barplot(table(df$LastDigit),col = 'green')  
#table(df$LastDigit) %>% barplot(col = 'green')  
table(df$LastDigit)%>%barplot(legend=TRUE,col = brewer.pal(n=9,"Paired"),xlab="1a. Last Digit of Prime Numbers")



### 1b. Show a barplot of the first digit

# Different approaches to getting the data.   
# data <- df$FirstDigit  
# barplot( table(data),col = 'Blue')  
# Tried using the %>% operator   
table(df$FirstDigit) %>% barplot(legend=TRUE,col = brewer.pal(10,"Paired"),xlab="1b. First Digit of Prime Numbers")



# Different notes to see what was possible  
#table(df$FirstDigit) %>% barplot(col = 'gold')  
#count(df,df$FirstDigit)%>%table()%>%barplot()

### 1c What two interences can you draw from the plots above?

Last Digit Plot: 1. No prime numbers > 10 end with 2 or 5 2. Close distribution of remaining numbers used

First Digits Plot: 1. First digit of a prime is almost evenly distributed

## Part 2. Initialize the quarter coin productions dataset

about the of the 50 US states by the DenverMint and PhillyMint. The numbers in the dataset (in thousands ) are the number of quarters minted.

us\_quarters <- read.csv("http://people.bu.edu/kalathur/datasets/us\_quarters.csv")

Basic Exploration

# See some basic metrics   
#dim(us\_quarters)  
#colnames(us\_quarters)  
#head(us\_quarters)  
#tail(us\_quarters)  
  
#might be easier to view this 'wide' mint data as a single column  
gQuarters<-gather(us\_quarters,Mint,qrtCount,c(DenverMint,PhillyMint))  
  
# Now filter:  
pm<- filter(gQuarters,Mint=="PhillyMint")  
dm<- filter(gQuarters,Mint=="DenverMint")  
#sum(gQuarters$qrtCount)/4  
  
head(arrState <-arrange(gQuarters,State),4)

## State Mint qrtCount  
## 1 Alabama DenverMint 232400  
## 2 Alabama PhillyMint 225000  
## 3 Alaska DenverMint 254000  
## 4 Alaska PhillyMint 251800

arrMint <- arrange(gQuarters,Mint)  
arrCount <- arrange(gQuarters,desc(qrtCount))  
  
# ggplot(g,aes(x=State, y = qrtCount, color = Mint))+  
# geom\_point( )+  
# theme(axis.text.x = element\_text(angle = 80, hjust = 1))

### 2a. Highest number of quarters produced by each mint?

For which state were the lowest number of quarters produced by each mint?

# Highest  
# Find the index of the max  
c<- which(max(us\_quarters$DenverMint) == us\_quarters$DenverMint)  
state<-us\_quarters$State[c]  
  
d<- which(min(us\_quarters$DenverMint) == us\_quarters$DenverMint)  
state<-us\_quarters$State[d]  
  
e<- which(max(us\_quarters$PhillyMint) == us\_quarters$PhillyMint)  
state<-us\_quarters$State[e]  
  
f<- which(min(us\_quarters$PhillyMint) == us\_quarters$PhillyMint)  
state<-us\_quarters$State[f]  
  
  
paste("Denver Mint: State with the most printed quarters: ", state)

## [1] "Denver Mint: State with the most printed quarters: Iowa"

paste("Denver Mint: State with the least printed quarters: ", state)

## [1] "Denver Mint: State with the least printed quarters: Iowa"

paste("Philly Mint: State with the most printed quarters: ", state)

## [1] "Philly Mint: State with the most printed quarters: Iowa"

paste("Philly Mint: State with the least printed quarters: ", state)

## [1] "Philly Mint: State with the least printed quarters: Iowa"

### 2b. What is the value of the total coins in dollars?

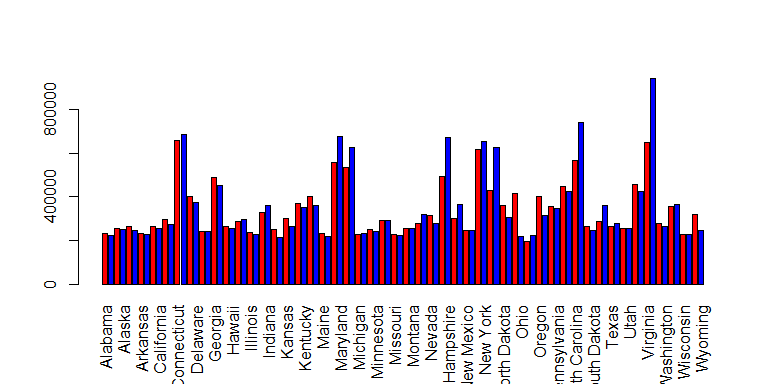
# the number of quarters is in THOUSANDS  
totalQuarters <- sum(us\_quarters$DenverMint, us\_quarters$PhillyMint)  
totalDollars <- totalQuarters / 4  
paste("Dollar amount of all quarters printed from both mints is $",totalDollars," dollars")

## [1] "Dollar amount of all quarters printed from both mints is $ 8699400 dollars"

### 2c. Produce barplot from the data using the R barplot function

with the data for the two mints as a matrix. Write any two striking inferences you can observe by looking at the plot.

# transpose but leave out states  
# help from https://www.r-graph-gallery.com/210-custom-barplot-layout.html  
options(scipen = 1)  
usq\_t<-t(us\_quarters[-1])  
#barplot(usq\_t,names=us\_quarters$State,las = 3,beside= TRUE,col = c("red","blue"))  
# Using the arranged by State dataset (~line 111)   
barplot(arrState$qrtCount,names=arrState$State,las = 3,beside= TRUE,col = c("red","blue"))

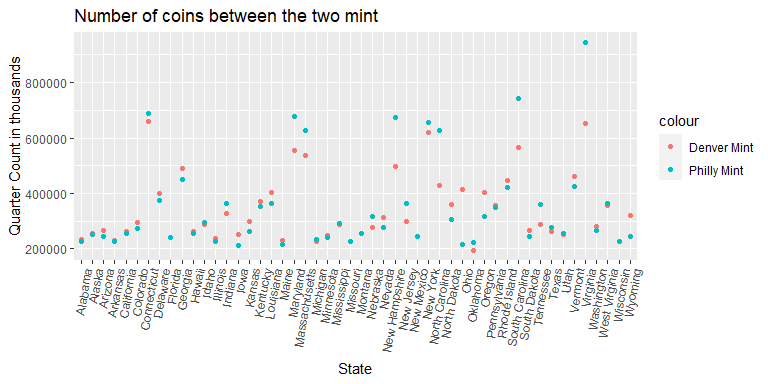


pars=list # controls the following graphical parameters   
 ((boxwex=.8, # determines the width of the box   
 staplewex=.6), # determines the width of the whisker see "help(bxp)" for more information   
 ylab="Response", # y axis label   
 xlab="Treatments", # x axis label   
 main="Experiment", # graphic title   
 las=1, # controls the orientation of the axis labels (1=horizontal)   
 cex.lab=1.1) # controls the font size of the axis labels

### 2d. Scatter plot: Number of coins between the two mints.

Write any two inferences you can observe looking at the plot.

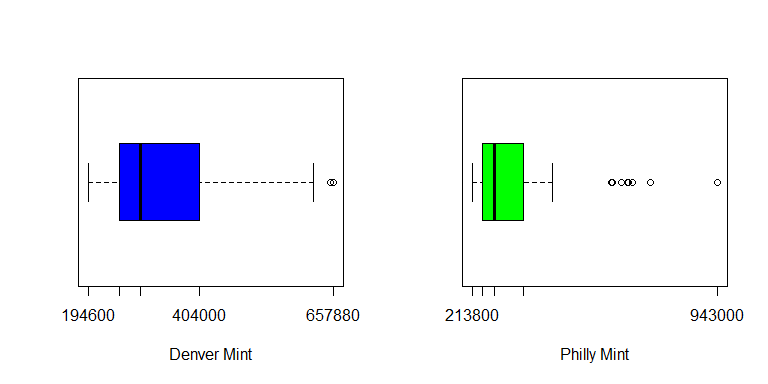
options(scipen = 1)  
#plot(us\_quarters$DenverMint)  
#plot(xus\_quarters$PhillyMint)  
  
  
p<-ggplot(data = us\_quarters) +   
 geom\_point(mapping = aes(x = State, y= DenverMint,col='Denver Mint')) +  
 geom\_point(mapping = aes(x = State, y= PhillyMint,col='Philly Mint'))+  
 theme(axis.text.x = element\_text(angle = 80, hjust = 1))  
p+ggtitle("Number of coins between the two mint")+   
 ylab("Quarter Count in thousands")

 Inferences from the above plot: 1. The Philly mint has a bias toward printing Quarters from Virginia for some reason. 2. The Denver mint doesn’t print many quarters from Ohio

### 2e. Show the side-by-side box plots for the two mints.

Write any two inferences for each of the box plots.

par( mfrow=c(1,2))  
boxplot(us\_quarters$DenverMint,horizontal = TRUE,xaxt = "n", xlab = "Denver Mint", col = "blue")  
axis(side = 1, at = fivenum(us\_quarters$DenverMint), labels = TRUE)  
  
boxplot(us\_quarters$PhillyMint, horizontal = TRUE,xaxt = "n", xlab = "Philly Mint", col = "green")  
axis(side = 1, at = fivenum(us\_quarters$PhillyMint), labels = TRUE)

 The Denver Mint box plot show several outliers, while the Philly mint shows many more. We can infer that the denver mint prints quarters the most equitably from all the states. The Philly mint, less so.

### 2f. What states would be considered as outliers

For each of the two mints. Use the five number summary function to derive the outlier bounds

# Use the fivenum for detecting the outliers  
f<- fivenum(us\_quarters$DenverMint)  
lower <- f[2] - 1.5\*(f[4] - f[2])  
upper <- f[4] + 1.5\*(f[4] - f[2])  
outlier\_ind <- which(us\_quarters$DenverMint < lower | us\_quarters$DenverMint > upper)   
#us\_quarters$State[outlier\_ind]  
paste("The outlier states from the Denver Mint are: ")

## [1] "The outlier states from the Denver Mint are: "

paste(us\_quarters$State[outlier\_ind])

## [1] "Connecticut" "Virginia"

f<- fivenum(us\_quarters$PhillyMint)  
outlier\_ind2 <- which(us\_quarters$PhillyMint < lower | us\_quarters$PhillyMint > upper)   
#us\_quarters$State[outlier\_ind2]  
paste("The outlier states from the Philly mint are: ")

## [1] "The outlier states from the Philly mint are: "

paste(us\_quarters$State[outlier\_ind2])

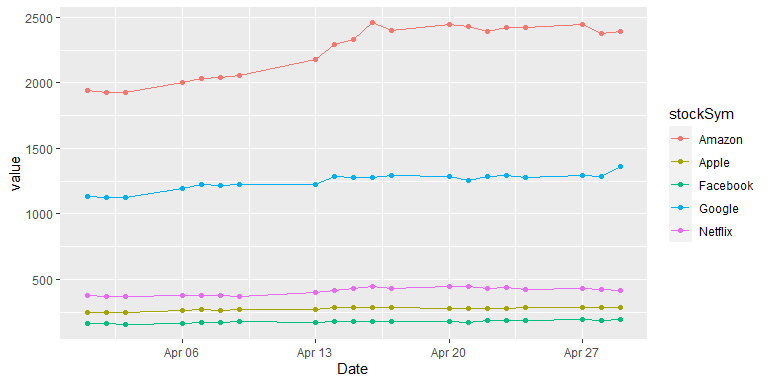
## [1] "Connecticut" "Maryland" "South Carolina" "New Hampshire"   
## [5] "Virginia" "New York"

## Part 3. Stock Data

stocks <- read.csv("http://people.bu.edu/kalathur/datasets/faang.csv")  
# Get some basic details  
#head(stocks,2)  
#tail(stocks,2)  
#summary(stocks[-1])

### 3a. Show the pair wise plots for all the 5 stocks in the dataset in a single plot.

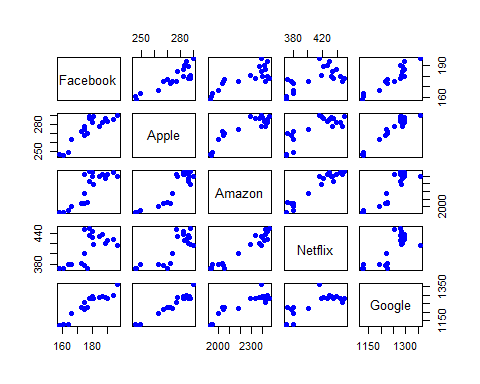
# Like with the mint data - it would be easier to reformat the data  
# putting all of the stock symbols in one column and all of the values in one column.  
# This does mean that there will be repeating date values, but this works well for plotting.  
  
# Use gather(data, <newVar>,<newVar>,c(<column names>))  
gStocks<-gather(stocks, stockSym, value,c(Facebook,Apple,Amazon,Netflix,Google))  
# Change the date vector using lubridate...  
gStocks$Date <- ymd(stocks$Date)  
# now pipe it all together  
gStocks %>% ggplot(aes(x=Date,y=value,color = stockSym))+ geom\_line()+ geom\_point()



# theme(axis.text.x = element\_text(angle = 80, hjust = 1))

### 3b. Show the correlation matrix for the 5 stocks in the dataset.

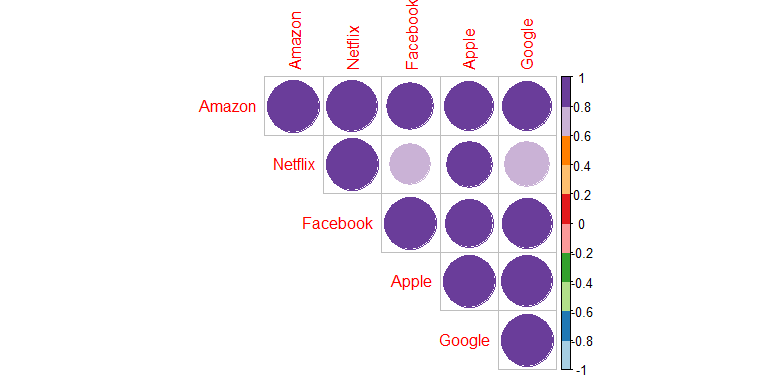
# use the pairs - but dont include the date!  
pairs(stocks[-1],pch=16, col = 'Blue')



(c<-cor(stocks[-1]))

## Facebook Apple Amazon Netflix Google  
## Facebook 1.0000000 0.8600322 0.8158242 0.6455015 0.9248847  
## Apple 0.8600322 1.0000000 0.8909518 0.8039984 0.9646654  
## Amazon 0.8158242 0.8909518 1.0000000 0.9622807 0.8894434  
## Netflix 0.6455015 0.8039984 0.9622807 1.0000000 0.7738125  
## Google 0.9248847 0.9646654 0.8894434 0.7738125 1.0000000

# different view of the corr matrix. Only include the upper and  
# make it easy to visually see corr/ non-corr  
corrplot(c,type = "upper",order = "hclust",col = brewer.pal(n=10,name = "Paired"))



### 3c. Provide at least 4 interpretations of the results.

1. All the stocks are correlated positively  
2. All stocks can be grouped together - say 'Tech Stocks'  
3. Netflix is not as strongly correlated with Google or Facebook  
4. Netflix would be the first choice to exclude from the 'Tech Stock' group since it is not as strongly correlated with the others.

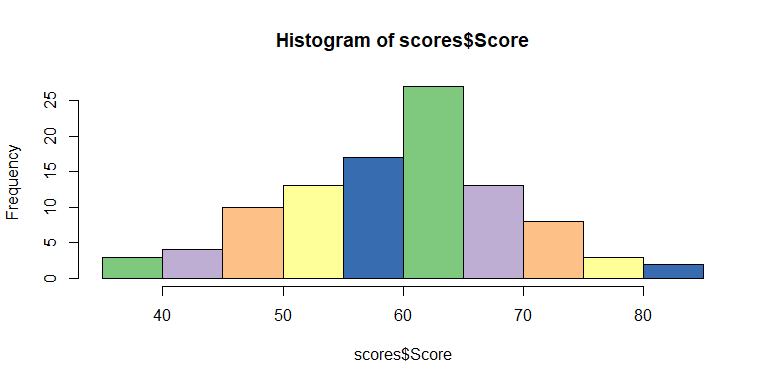
## 4. Scores of 100 students

scores <- read.csv("http://people.bu.edu/kalathur/datasets/scores.csv")  
  
# Get some basics  
#head(scores)  
#summary(scores$Score)

### 4a. Save the result of the histogram

into a variable. Using the counts and breaks property of this variable, write the R code to produce the following output. The code for the following output should not refer to the individual scores.

library(knitr)  
studentHist<-hist(scores$Score, col = brewer.pal(5,"Accent"))



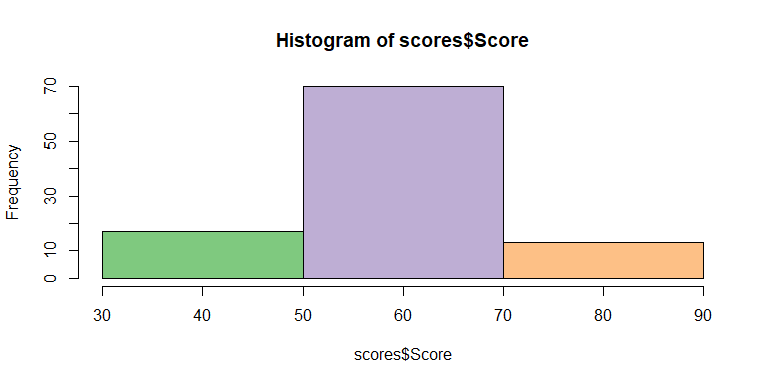
# I don't care for the sprintf. Change to print( paste() )  
for (i in seq(1,length(studentHist$count)) ) {  
 j<-sprintf("%d students in range (,%d,%d ] ", studentHist$counts[i],studentHist$breaks[i],studentHist$breaks[i+1])  
 print(j)  
 }

## [1] "3 students in range (,35,40 ] "  
## [1] "4 students in range (,40,45 ] "  
## [1] "10 students in range (,45,50 ] "  
## [1] "13 students in range (,50,55 ] "  
## [1] "17 students in range (,55,60 ] "  
## [1] "27 students in range (,60,65 ] "  
## [1] "13 students in range (,65,70 ] "  
## [1] "8 students in range (,70,75 ] "  
## [1] "3 students in range (,75,80 ] "  
## [1] "2 students in range (,80,85 ] "

### 4b. Show the histogram and the custom output

Using the breaks option of the histogram. Show that students in the 1. range (70,90] get an A grade, 2. (50,70] get a B grade, and 3. (30-50] get a C grade. The code for the following output should not refer to the individual scores.

newHist<-hist(scores$Score,breaks = seq(30,90,20),col=brewer.pal(4,"Accent"))



for (i in seq(1,length(newHist$count)) ) {  
 j<-sprintf("%d students in range (,%d,%d ] ", newHist$counts[i],newHist$breaks[i],newHist$breaks[i+1])  
 print(j)  
}

## [1] "17 students in range (,30,50 ] "  
## [1] "70 students in range (,50,70 ] "  
## [1] "13 students in range (,70,90 ] "