

STATS 419 Survey of Multivariate Analysis

Week 03 Assignment: datasets_revisted

Kevin Black
(kevin.black@wsu.edu)
[11480710]

Instructor: Monte J. Shaffer

11 September, 2020

1 Question 1

Create the “rotate matrix” functions described in lectures from the sample matrix. Apply these to the example “myMatrix”.

I was able to create matrix rotation functions by implementing two simple steps in the rotateMatrix90 function: Take a matrix and reverse the columns using the apply function, then transpose that reversed matrix, thus creating a rotation effect. For the rotateMatrix180 and rotateMatrix270 functions, I simply performed two 90° rotations for the former, as well as a 180° rotation and a 90° rotation for the latter. R code for the rotation functions can be found below along with their respective outputs. I chose this approach of reversing and transposing because matrix multiplication was preferred by the instructor, not required, and this approach was simpler/easier to implement and gives the desired output.

```
myMatrix = matrix(c(
  1, 0, 2,
  0, 3, 0,
  4, 0, 5
), nrow=3, byrow=T);

transposeMatrix = function(mat)
{
  t(mat);
}

rotateMatrix90 = function(mat)
{
  t(apply(mat, 2, rev)); # Rotation by reversing the cols, then transpose
}

rotateMatrix180 = function(mat)
{
  rotateMatrix90(rotateMatrix90(mat)); # Repeat two 90 deg rotations
}
```

```
rotateMatrix270 = function(mat)
{
  rotateMatrix180(rotateMatrix90(mat)); # Rotate a 90 deg rotation with a 180 deg rotation
}
```

```
myMatrix
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    2
## [2,]    0    3    0
## [3,]    4    0    5
```

```
rotateMatrix90(myMatrix)
```

```
##      [,1] [,2] [,3]
## [1,]    4    0    1
## [2,]    0    3    0
## [3,]    5    0    2
```

```
rotateMatrix180(myMatrix)
```

```
##      [,1] [,2] [,3]
## [1,]    5    0    4
## [2,]    0    3    0
## [3,]    2    0    1
```

```
rotateMatrix270(myMatrix)
```

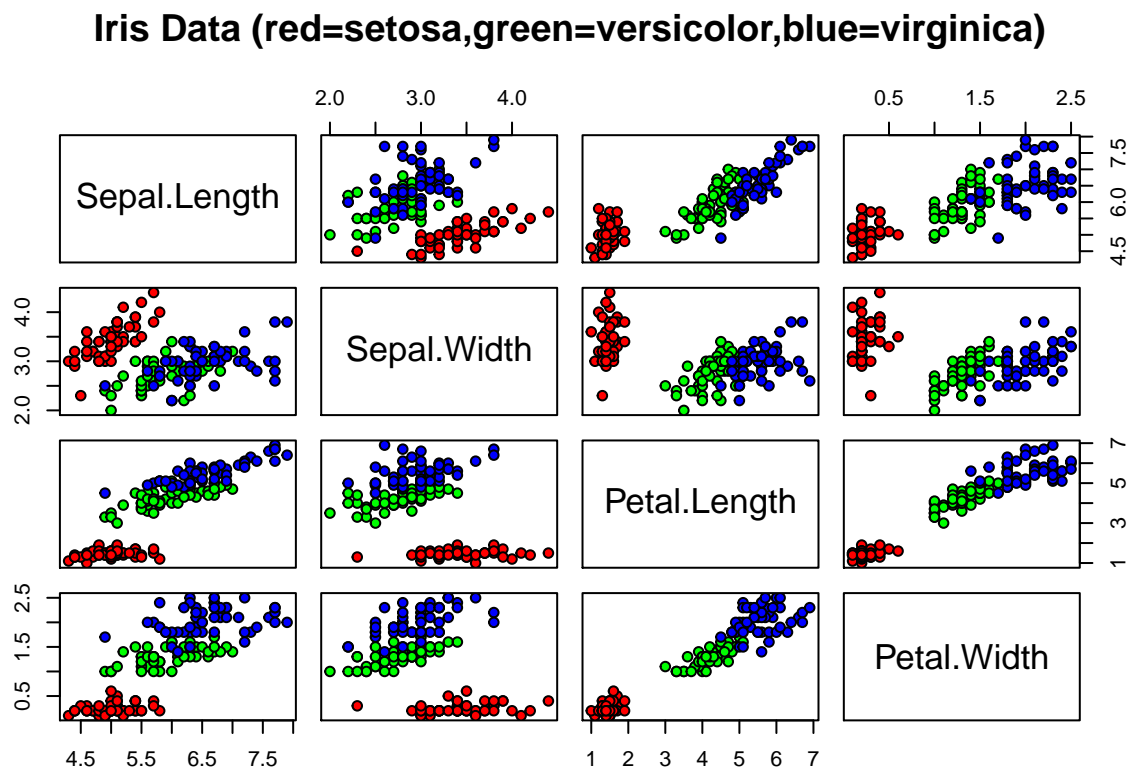
```
##      [,1] [,2] [,3]
## [1,]    2    0    5
## [2,]    0    3    0
## [3,]    1    0    4
```

2 Question 2

Recreate the graphic for the IRIS Data Set using R. Same titles, same scales, same colors. See: https://en.wikipedia.org/wiki/Iris_flower_data_set#/media/File:Iris_dataset_scatterplot.svg

I was able to recreate the specified graphic with the R code below, which produced the following scatterplot matrix. This block of code can be described by the following: `x=iris[,1:4]` supplies pairs with the data points for just the sepal and petal lengths/widths, `pch = 21` is a particular plotting symbol that plots those data points as a filled circle, `bg = c("red", "green", "blue")[iris$Species]` colors those data points according to their species type, and the main parameter specifies the title of the scatterplot matrix.

```
pairs(x = iris[,1:4], # Supply just the sepal and petal lengths/widths
      pch = 21, # Plotting symbol 21, a filled circle
      bg = c("red", "green", "blue")[iris$Species], # Color points by species
      main = "Iris Data (red=setosa,green=versicolor,blue=virginica)") # Title
```



3 Question 3

Write 2-3 sentences concisely defining the IRIS Data Set. Maybe search KAGGLE for a nice template. Be certain the final writeup are your own sentences (make certain you modify what you find, make it your own, but also cite where you got your ideas from).

The Iris data set is a multivariate collection of 150 total observations consisting of an ID, measured in centimeters the sepal length, sepal width, petal length, petal width, and describing the species for three different iris flower species; Iris setosa, Iris virginica, and Iris versicolor. This dataset was introduced by Ronald Fisher in his 1936 paper ‘The use of multiple measurements in taxonomic problems’, but the data itself was collected by Edgar Anderson and utilized in his own paper ‘The species problem in Iris’ in the same year [2, 1]. It is quite apparent that this Iris data set is relatively famous as it is typically used as a starting point in understanding common concepts in data analysis and practicing exploratory data analysis for aspiring data scientists and analysts.

4 Question 4

Import “personality-raw.txt” into R. Remove the V00 column. Create two new columns from the current column “date_test”: year and week. Stack Overflow may help: <https://stackoverflow.com/questions/22439540/how-to-get-week-numbers-from-dates> ... Sort the new data frame by YEAR, WEEK so the newest tests are first ... The newest tests (e.g., 2020 or 2019) are at the top of the data frame. Then remove duplicates using the unique function based on the column “md5_email”. Save the data frame in the same “pipe-delimited format” (| is a pipe) with the headers. You will keep the new data frame as “personality-clean.txt” for future work (you will not upload it at this time). In the homework, for this task, report how many records your raw dataset had and how many records your clean dataset has.

For cleaning up and modifying personality-raw.txt, I was able to accomplish the required steps with the R code below. As a side note: I did not use the unique function for removing duplicates, I used duplicated since it was easier, more straightforward, and still part of base R. For the last step, I used dim(df) before and after cleaning to see how many records each of the datasets had; personality-raw.txt had 838 rows, and personality-clean.txt had 678 rows.

```
df <- read.table("../datasets/personality/personality-raw.txt",
                 header=T, sep="|", dec=".")[, -3] # Import data, omit V00
YEAR <- c() # Create empty vectors for YEAR and WEEK
WEEK <- c()

for (i in 1:length(df$date_test)){ # Loop through date_test col of df
  curr <- unlist(strsplit(as.character(df$date_test[i]), " "))[1] # Grab date
  curr.form <- as.Date(curr, "%m/%d/%Y") # Convert character to formatted date
  YEAR[i] <- format(curr.form,"%Y") # Add year to YEAR vector
  WEEK[i] <- format(curr.form,"%V") # Add week number to WEEK vector
}

df$YEAR <- YEAR # Add YEAR and WEEK cols to original df
df$WEEK <- WEEK
df <- df[order(df$YEAR,df$WEEK, decreasing=TRUE),] # Sort dataframe by YEAR and WEEK

df <- df[!duplicated(df$md5_email),] # Keep unique rows based on df$md5_email

# Write cleaned df to .txt file
write.table(df, file = "../datasets/personality/personality-clean.txt", sep="|")

dim(df)
```

```
## [1] 678 64
```

```
head(df)
```

```
##                md5_email      date_test V01 V02 V03 V04 V05 V06
## 838 b62c73cdaf59e0a13de495b84030734e 4/6/2020 12:57 3.4 4.2 2.6 4.2 2.6 2.6
## 837 1358d38e6898b1a0e5940f8b99ba2325 12/1/2019 22:12 3.4 3.4 3.4 4.2 4.2 4.2
## 835 f529455e4400e76f323f8c68154e194b 5/6/2019 4:44 4.2 5.0 1.8 4.2 4.2 5.0
## 836 0445a05e751e17de30ebdcdbcdb575d59 5/6/2019 10:32 1.8 2.6 3.4 4.2 5.0 3.4
## 828 bfd1c69406d322d17312e965752813c2 5/2/2019 10:26 2.6 4.2 1.0 4.2 4.2 2.6
## 829 9cf05d7d516099c9533b98beb91993b9 5/2/2019 10:48 5.0 5.0 1.8 5.0 5.0 4.2
##      V07 V08 V09 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21 V22 V23 V24 V25
## 838 4.2 2.6 3.4 4.2 4.2 3.4 3.4 4.2 5.0 3.4 5.0 3.4 1.8 2.6 2.6 2.6 4.2 3.4 5.0
## 837 5.0 3.4 4.2 3.4 2.6 3.4 3.4 4.2 4.2 4.2 4.2 4.2 3.4 2.6 3.4 4.2 4.2 4.2 2.6
```

```

## 835 3.4 3.4 4.2 3.4 2.6 2.6 4.2 5.0 3.4 4.2 5.0 4.2 2.6 2.6 1.8 3.4 5.0 3.4 1.8
## 836 2.6 2.6 5.0 3.4 2.6 4.2 2.6 3.4 4.2 3.4 4.2 4.2 3.4 1.8 2.6 3.4 4.2 5.0 1.8
## 828 3.4 1.8 4.2 4.2 1.8 2.6 3.4 5.0 4.2 4.2 5.0 4.2 3.4 1.8 1.0 3.4 4.2 3.4 1.8
## 829 1.0 5.0 5.0 5.0 1.0 5.0 5.0 5.0 5.0 5.0 5.0 3.4 3.4 3.4 4.2 5.0 5.0 3.4 1.8
##      V26 V27 V28 V29 V30 V31 V32 V33 V34 V35 V36 V37 V38 V39 V40 V41 V42 V43 V44
## 838 2.6 4.2 3.4 2.6 2.6 4.2 1.8 3.4 4.2 4.2 4.2 2.6 4.2 2.6 4.2 4.2 4.2 4.2 2.6
## 837 4.2 4.2 3.4 2.6 4.2 4.2 3.4 4.2 3.4 4.2 5.0 3.4 4.2 4.2 4.2 4.2 4.2 4.2 4.2
## 835 4.2 3.4 5.0 1.8 5.0 4.2 1.8 4.2 3.4 2.6 3.4 2.6 3.4 3.4 5.0 3.4 3.4 3.4 3.4
## 836 3.4 2.6 3.4 2.6 4.2 5.0 5.0 5.0 5.0 5.0 5.0 3.4 5.0 5.0 5.0 4.2 5.0 5.0 5.0
## 828 4.2 5.0 3.4 1.8 4.2 3.4 4.2 4.2 3.4 4.2 3.4 1.8 5.0 3.4 4.2 1.8 2.6 4.2 4.2
## 829 5.0 5.0 4.2 3.4 5.0 5.0 4.2 5.0 5.0 5.0 5.0 2.6 3.4 5.0 4.2 5.0 5.0 3.4 5.0
##      V45 V46 V47 V48 V49 V50 V51 V52 V53 V54 V55 V56 V57 V58 V59 V60 YEAR WEEK
## 838 4.2 4.2 2.6 3.4 2.6 4.2 1.8 4.2 2.6 3.4 4.2 4.2 1.8 4.2 2.6 4.2 2020   15
## 837 3.4 4.2 4.2 2.6 3.4 4.2 3.4 4.2 4.2 4.2 4.2 3.4 4.2 4.2 3.4 3.4 2019   48
## 835 4.2 5.0 3.4 4.2 3.4 4.2 2.6 3.4 5.0 5.0 3.4 3.4 3.4 3.4 1.8 3.4 2019   19
## 836 5.0 5.0 5.0 4.2 4.2 3.4 3.4 5.0 5.0 5.0 2.6 5.0 5.0 5.0 4.2 5.0 2019   19
## 828 4.2 4.2 4.2 2.6 3.4 1.8 2.6 2.6 5.0 4.2 3.4 2.6 2.6 4.2 4.2 4.2 2019   18
## 829 5.0 5.0 5.0 3.4 5.0 5.0 5.0 5.0 5.0 5.0 1.0 2.6 3.4 5.0 5.0 3.4 2019   18

```

5 Question 5

Write functions for `doSummary` and `sampleVariance` and `doMode`. . . test these functions in your homework on the “monte.shaffer@gmail.com” record from the clean dataset. Report your findings. For this “monte.shaffer@gmail.com” record, also create z-scores. Plot(x,y) where x is the raw scores for “monte.shaffer@gmail.com” and y is the z-scores from those raw scores. Include the plot in your assignment, and write 2 sentences describing what pattern you are seeing and why this pattern is present.

After creating functions for `doSummary`, `sampleVariance`, and `doMode`, I’m able to report the various summary statistics about the “monte.shaffer@gmail.com” record or, in terms of md5, “b62c73c...”. It is also worth noting that the record was trimmed of the columns `md5_email`, `date_test`, `YEAR`, and `WEEK`, leaving only the numerical values. Additionally, per the assignment description, I generated z-scores for the record’s numerical values, of which a plot for the scores can be found below. After plotting the raw scores against the z-scores, there appears to be a linear relationship between both sets of scores. I believe this pattern is present because we are standardizing the data using the formula $z = \frac{x - \mu}{\sigma}$, which establishes how z-scores for a normal distribution can be expressed by the slope of a linear function. R code for the above functions and other functionalities can be found below.

```
doSummary = function(x)
{
  z.scores <- c()
  naive = doSampleVariance(as.numeric(x[,3:62]),'naive') # Calculate naive var
  two.pass = doSampleVariance(as.numeric(x[,3:62]),'two-pass') # Calculate two-pass var
  mean = mean(as.numeric(x[,3:62])) # Calculate mean
  stdev = sd(as.numeric(x[,3:62])) # Calculate stdev
  x = as.numeric(x[,3:62]) # Convert vector to numerical values

  print(paste("Length:",length(x))) # Length of x
  print(paste("# of NAs:",sum(is.na(x)))) # Number of NAs
  print(paste("Mean:",mean)) # Mean of x
  print(paste("Median:",median(x))) # Median of x
  print(paste("Mode:",doMode(x))) # Mode of x
  cat("",sep="\n")
  print(paste("Variance (Base):",var(x))) # Base R var
  cat("",sep="\n")
  print(paste("Variance (Naive, Sum):",naive[1])) # Sum when calculating naive var
  print(paste("Variance (Naive, SumSq):",naive[2])) # SumSq when calculating naive var
  print(paste("Variance (Naive, Var):",naive[3])) # Naive var
  cat("",sep="\n")
  print(paste("Variance (Two-pass, sum1):",two.pass[1])) # sum1 for two-pass var
  print(paste("Variance (Two-pass, sum2):",two.pass[2])) # sum2 for two-pass var
  print(paste("Variance (Two-pass, Var):",two.pass[3])) # Two-pass var
  cat("",sep="\n")
  print(paste("Stdev (Base):",stdev)) # Base R stdev
  print(paste("Stdev (Naive):",sqrt(as.numeric(naive[3])))) # Naive stdev
  print(paste("Stdev (Two-pass):",sqrt(as.numeric(two.pass[3])))) # Two-pass stdev

  for (i in 1:length(x)) # Compute z-scores for specified record
  {
    z = (x[i]-mean)/stdev # z-score formula
    z.scores <- append(z.scores,z) # Collection of z-scores
  }

  plot(x,z.scores) # Plot raw scores against z-scores
```

```

unique(x)
}

doSampleVariance = function(x,method) # Can perform two variance calculations
{
  if(method=='naive') # Algorithm from https://bit.ly/31QSe2I
  {
    n = Sum = SumSq = 0
    for (i in 1:length(x))
    {
      n = n+1
      Sum = Sum + x[i]
      SumSq = SumSq+(x[i]*x[i])
    }
    Var = (SumSq - (Sum*Sum)/n)/(n-1)
    stats <- list(Sum,SumSq,Var)
    return(stats)
  }
  else # Algorithm from https://bit.ly/32R77kS
  {
    n = sum1 = sum2 = 0
    for (i in 1:length(x))
    {
      n = n+1
      sum1 = sum1 + x[i]
    }
    mean = sum1 / n
    for (i in 1:length(x))
    {
      sum2 = sum2 + (x[i]-mean)^2
    }
    Var = sum2 / (n-1)
    stats <- list(sum1,sum2,Var)
    return(stats)
  }
}

doMode = function(x) # Perform mode calculation
{
  uniq <- unique(x) # Isolate unique values
  matches <- match(x,uniq) # Indexes for every x value in uniq
  freq <- tabulate(matches) # Frequencies for each x value
  return(uniq[which.max(freq)]) # Index for max frequency in uniq
}

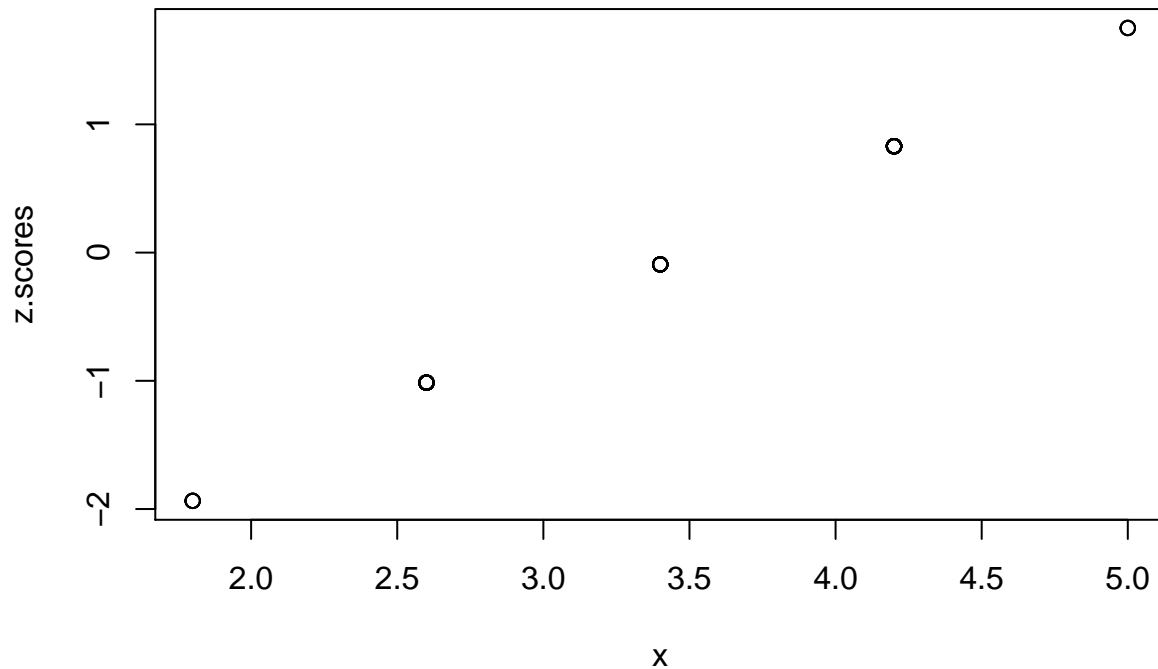
df <- read.table("personality-clean.txt", header=T, sep="|", dec=".") # Read clean dataset
monte.record <- df[1,] # Record for monte.shaffer@gmail.com
doSummary(monte.record) # Various statistics for the monte record

## [1] "Length: 60"
## [1] "# of NAs: 0"
## [1] "Mean: 3.48"
## [1] "Median: 3.4"
## [1] "Mode: 4.2"

```



```
##  
## [1] "Variance (Base): 0.752813559322034"  
##  
## [1] "Variance (Naive, Sum): 208.8"  
## [1] "Variance (Naive, SumSq): 771.04"  
## [1] "Variance (Naive, Var): 0.752813559322044"  
##  
## [1] "Variance (Two-pass, sum1): 208.8"  
## [1] "Variance (Two-pass, sum2): 44.416"  
## [1] "Variance (Two-pass, Var): 0.752813559322034"  
##  
## [1] "Stdev (Base): 0.867648292410026"  
## [1] "Stdev (Naive): 0.867648292410032"  
## [1] "Stdev (Two-pass): 0.867648292410026"
```



```
## [1] 3.4 4.2 2.6 5.0 1.8
```

6 Question 6

Compare Will Smith and Denzel Washington. [See 03.n greater 1-v2.txt for the necessary functions and will-vs-denzel.txt for some sample code]. You will have to create a new variable \$millions.2000 that converts each movie's \$millions based on the \$year of the movie, so all dollars are in the same time frame. You will need inflation data from about 1980-2020 to make this work.

To compare Will Smith and Denzel Washington in an objective manner, Monte Shaffer supplied us with somewhat abstruse R code, expects us to compare the various variables for their movies, and provided us with the research question “Who is a better actor? Will Smith or Denzel Washington?” The latter question will be answered in question 7 as that question focuses on comparing several variables between both actors, while this question will focus more on adjusting box office values of the past to today's time frame for a more accurate comparison of films by gross income. To make the data easier to work with, I isolated the year and millions columns from the actor\$movies.50 data frame, sorted the data frame by year, and converted the millions column into their actual values by multiplying each value by 1,000,000. I then created the new variable millions.2000 in this same data frame and added the adjusted values to each film's box office value. R code for the described functionalities can be found below and is generally the same for both actors. Example output for the resulting data frames can be found immediately following the R code. Note: For some reason the functions provided by Monte Shaffer didn't work for the years 2019-2020 for Will Smith as it probably has something to do with the web page elements being slightly different for those years, although I just manually calculated the adjusted values and hard-coded them into the data frame since there were only five entries that had this issues. Additionally, the only thing I changed in the provided functions was rearranging the operations for scraping the inflation data into its own function for ease of access as I went through the data frame.

```
## Will Smith movie millions adjusted for inflation
nmid = "nm0000226";
will = grabFilmsForPerson(nmid);
Will.Year.Mil <- will$movies.50[,c(4,11)] # Extract just year and millions
Will.Year.Mil <- Will.Year.Mil[order(Will.Year.Mil$year),] # Sort by year
Will.Year.Mil$millions <- Will.Year.Mil$millions*1000000 # Actual money amount

for(j in 1:dim(Will.Year.Mil)[1])
{
  if (is.na(Will.Year.Mil[j,2]))
  {
    Will.Year.Mil$millions.2000[j] <- NA
    next
  }
  if (Will.Year.Mil[j,1] == 2019) # Hard coded since 2019-2020 don't work with website
  {
    Will.Year.Mil$millions.2000[46] <- 360349105.97
    Will.Year.Mil$millions.2000[47] <- 20826791.90
    Will.Year.Mil$millions.2000[48] <- NA
    Will.Year.Mil$millions.2000[49] <- 790505.97
    Will.Year.Mil$millions.2000[50] <- 204420000.00
    break
  }
  infl <- paste("https://www.officialdata.org/us/inflation/",Will.Year.Mil[j,1],
               "?endYear=2020&amount=",Will.Year.Mil[j,2],sep="")
  result <- inflation(infl) # Millions value with 2020 inflation
  Will.Year.Mil$millions.2000[j] <- result[1]; # Add to millions.2000 column
}

## Denzel Washington movie millions adjusted for inflation
```

```

nmid = "nm0000243";
denzel = grabFilmsForPerson(nmid);
Denzel.Year.Mil <- denzel$movies.50[,c(4,11)]
Denzel.Year.Mil <- Denzel.Year.Mil[order(Denzel.Year.Mil$year),] # Sort by year
Denzel.Year.Mil$millions <- Denzel.Year.Mil$millions*1000000 # Actual money amount

for(j in 1:dim(Denzel.Year.Mil)[1])
{
  if (is.na(Denzel.Year.Mil[j,2]))
  {
    Denzel.Year.Mil$millions.2000[j] <- NA
    next
  }
  if (Denzel.Year.Mil[j,1] == 2020)
  {
    break
  }
  infl = paste("https://www.officialdata.org/us/inflation/",Denzel.Year.Mil[j,1],
               "?endYear=2020&amount=",Denzel.Year.Mil[j,2],sep="")
  result <- inflation(infl) # Millions value with 2020 inflation
  Denzel.Year.Mil$millions.2000[j] <- result[1]; # Add to millions.2000 column
}

head(Will.Year.Mil)

```

```

##   year  millions  millions.2000
## 44 1993   6410000      11493685
## 48 1993  44940000      80581308
## 12 1995  65810000     111886068
## 31 1995   7920000      13465091
##  3 1996 306170000     505601996
##  4 1997 250690000     404698004

```

```
head(Denzel.Year.Mil)
```

```

##   year  millions  millions.2000
## 46 1981   9570000      27278290.1
## 41 1984  21820000     54413703.8
## 45 1986   3800000      8983428.8
## 38 1987   5900000     13456830.1
## 47 1988   190000      416138.5
## 19 1989  26830000     56061934.1

```

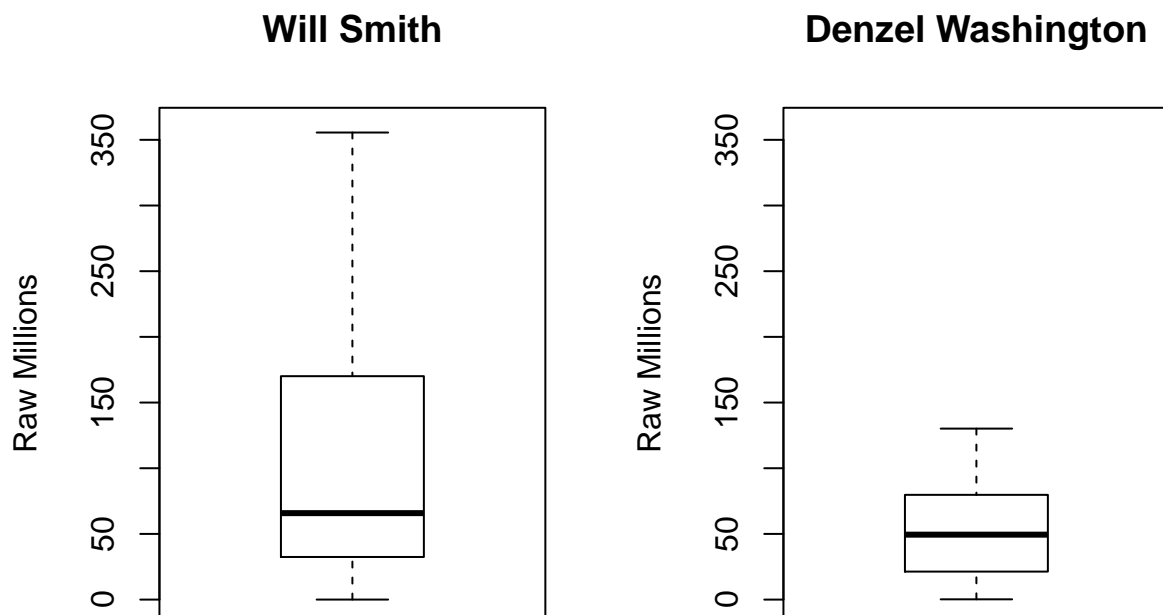
7 Question 7

Build side-by-side box plots on several of the variables (including #6) to compare the two movie stars. After each box plot, write 2+ sentences describing what you are seeing, and what conclusions you can logically make. You will need to review what the box plot is showing with the box portion, the divider in the box, and the whiskers.

To objectively determine who is the better actor, I will be comparing the two actors based on five variables: millions, millions.2000, ratings, metacritic, and votes. Also, to swiftly review on the anatomy of a box plot; the box portion represents the interquartile range of the data, the divider in the box represents the median of the data, and the whiskers represent the minimum and maximum of the data (left-to-right or bottom-to-top depending on the orientation of the box plot). Each box plot comparison and conclusive statements can be found below.

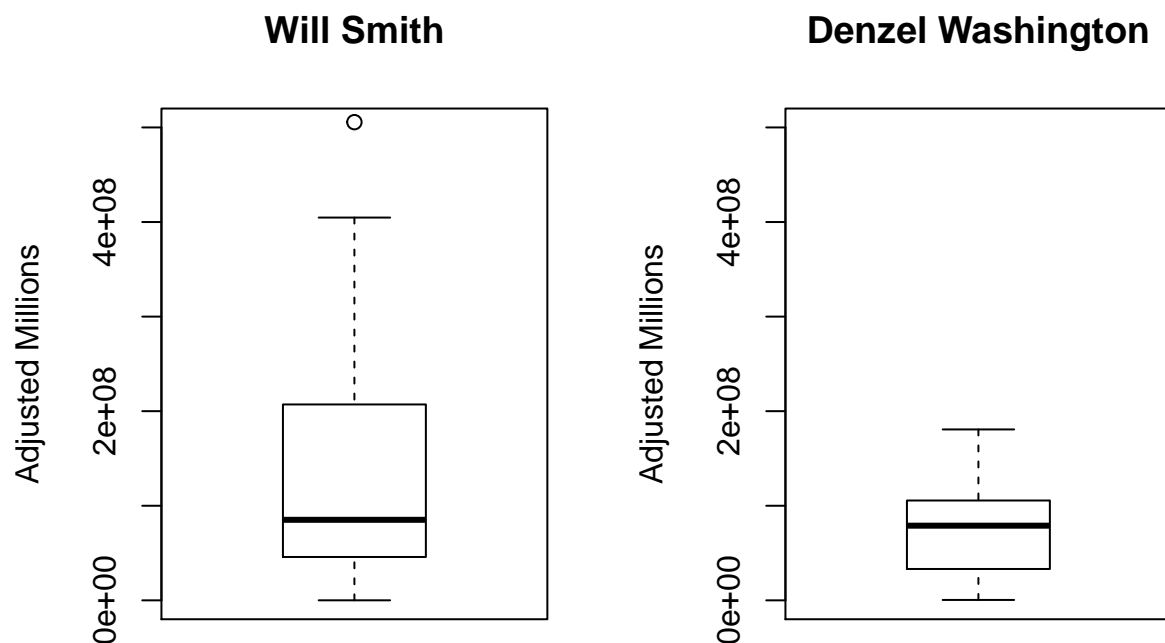
In terms of raw millions, the following side-by-side box plots show that Will Smith excels over Denzel Washington when it comes to the interquartile range (covers a wider and higher range of values), median (slightly higher than Denzel's median), and maximum value (nearly twice the value of Denzel's maximum). Although, since these are the raw millions data, it's not a reliable source of information for determining whose films made more from the box office since their films span numerous decades without adjusting for inflation. This is addressed in the next box plot comparison and will hopefully give some definite insight into whose films brought in more money.

```
# Comparing millions
par(mfrow=c(1,2));
boxplot(will$movies.50$millions, main=will$name, ylim=c(0,360), ylab="Raw Millions");
boxplot(denzel$movies.50$millions, main=denzel$name, ylim=c(0,360), ylab="Raw Millions");
```



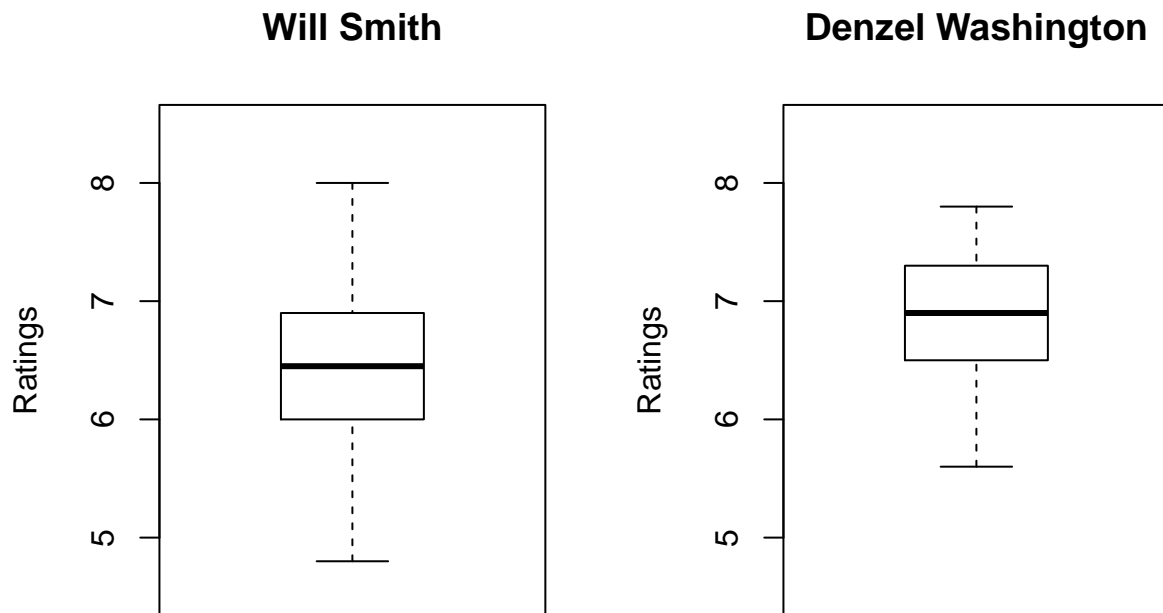
After adjusting the number of millions to today's values in question 6, it appears to be that Will Smith still made more from his top 50 movies than Denzel Washington even after adjusting for inflation. We can see that the following box plots in this comparison look pretty similar to the raw millions one, so it's fair to make the same conclusions as the one above with regard to the interquartile range, median, and max value. Thus, I can make the firm statement that Will Smith's top 50 movies brought in more money at the box office than the top 50 movies of Denzel Washington.

```
# Comparing millions.2000
par(mfrow=c(1,2));
boxplot(Will.Year.Mil$millions.2000, main=will$name, ylim=c(0,5e+08),
        ylab="Adjusted Millions");
boxplot(Denzel.Year.Mil$millions.2000, main=denzel$name, ylim=c(0,5e+08),
        ylab="Adjusted Millions");
```



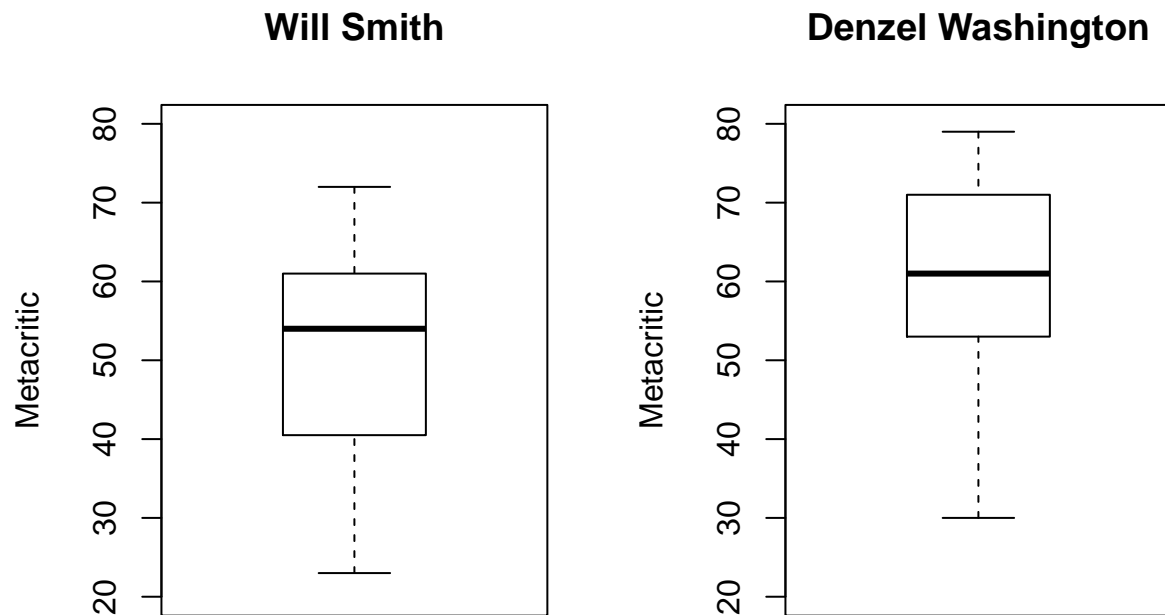
One could argue that based on the adjusted millions box plot that Will Smith was clearly the “better” actor, but other variables, like ratings of their movies, play a significant role in making that claim as well. From what we can see in the following box plots, Denzel Washington has a higher median and interquartile range for his ratings, as well as a higher minimum than Will Smith. On the contrary, Will Smith has a lower minimum rating, lower interquartile range, and lower median, but he possess’ a higher maximum rating. Based on these observations, I can make the objective claim that the quality of Denzel Washington’s movies are higher than Will Smith’s.

```
# Comparing ratings
par(mfrow=c(1,2));
boxplot(will$movies.50$ratings, main=will$name, ylim=c(4.5,8.5), ylab="Ratings");
boxplot(denzel$movies.50$ratings, main=denzel$name, ylim=c(4.5,8.5), ylab="Ratings");
```



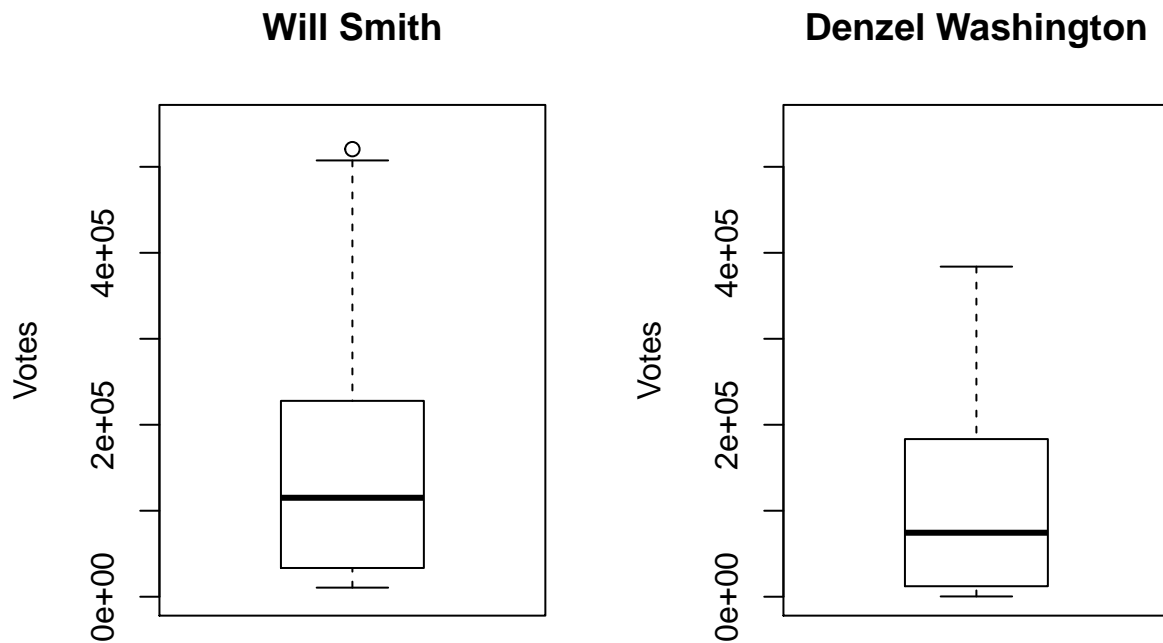
While we just went over ratings for their movies, those ratings came strictly from IMDb, whereas the following box plots illustrate the more general ratings for their movies, known as Metacritic. For this box plot comparison, we can pretty much make the same conclusions as the box plot comparison for IMDb ratings, except in this case Denzel Washington excels in every box plot category over Will Smith, including the maximum rating value. Given the previous statement, I can claim that Denzel Washington generally has a higher quality of his top 50 movies than that of Will Smith.

```
# Comparing metacritic
par(mfrow=c(1,2));
boxplot(will$movies.50$metacritic, main=will$name, ylim=c(20,80), ylab="Metacritic");
boxplot(denzel$movies.50$metacritic, main=denzel$name, ylim=c(20,80), ylab="Metacritic");
```



The following final box plot comparison shows the IMDb votes for both actors, which potentially shows the loyalty of each actor's fan base for them going out of their way to vote for those movies and could possibly translate to a more favored actor. From what we can observe, Denzel Washington's top 50 movies generally have a lower minimum and maximum of votes, smaller interquartile range, and lower median. These observations could, as mentioned earlier, indicate the loyalty of a fanbase because of them actually voting for these actor's movies, in which case Will Smith appears to be the more beloved actor in terms of votes received for his top 50 movies, but does not necessarily contribute to him being the "better" actor.

```
# Comparing votes
par(mfrow=c(1,2));
boxplot(will$movies.50$votes, main=will$name, ylim=c(0,5.5e+05), ylab="Votes");
boxplot(denzel$movies.50$votes, main=denzel$name, ylim=c(0,5.5e+05), ylab="Votes");
```



After performing multiple box plot comparisons for a multitude of variables from the top 50 movies for Will Smith and Denzel Washington, I can objectively come to a definite answer for the research question presented by instructor Monte Shaffer. To reiterate, the research question stated earlier in question 6 is “Who is a better actor? Will Smith or Denzel Washington?”. Based on the comparisons made above, Will Smith’s top 50 movies took over Denzel Washington’s movies when it came to the raw/adjusted millions and IMDb votes received, but Denzel Washington came out on top with his IMDb and Metacritic ratings. In my opinion, and what I’m sure many others can agree on, the question of “who is a better actor” really comes down to the quality of the actor and the everlasting impact of their movies, and not necessarily how much money one’s movies made; there have definitely been some high grossing movies with bad actors. Thus, I can conclude that while Will Smith excelled in some box plot comparisons, the better actor between the two ultimately comes down to **Denzel Washington**, for his superior quality of acting in his top 50 movies, which is portrayed by the box plots of his IMDb and Metacritic ratings.

References

- [1] Anderson, E. (1936). The species problem in iris. *Annals of the Missouri Botanical Garden* 23(3), 457–509.
- [2] Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7(2), 179–188.