STATS 419 Survey of Multivariate Analysis

Week 03 Assignment

Kathleen Rivas
(kathleen.rivas@wsu.edu)
[]

Instructor: Monte J. Shaffer

21 September, 2020

```
## Loading required package: usethis

## SHA-1 hash of file is 700ec64bd3746deb893bcb1b00e58e8dfc6339c7

## Registered S3 method overwritten by 'pryr':
##   method      from
##   print.bytes Rcpp

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: xml2

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

##
## Attaching package: 'readr'
```

```
## The following object is masked from 'package:rvest':
##
##     guess_encoding

## SHA-1 hash of file is 600ff5b754e70489540d0802ffec13e75fa1762b

## SHA-1 hash of file is ebc1c3f0fccfab7a2b9acb2e20d56142564f2694

## SHA-1 hash of file is b3ea6d77308877c254e8659afe7b93c64a558ec6

## SHA-1 hash of file is 09f182aed8a064442f9330d5d5fd0a00bb310642
```

1 Rotation Functions Using a 3x3 Matrix

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

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
```

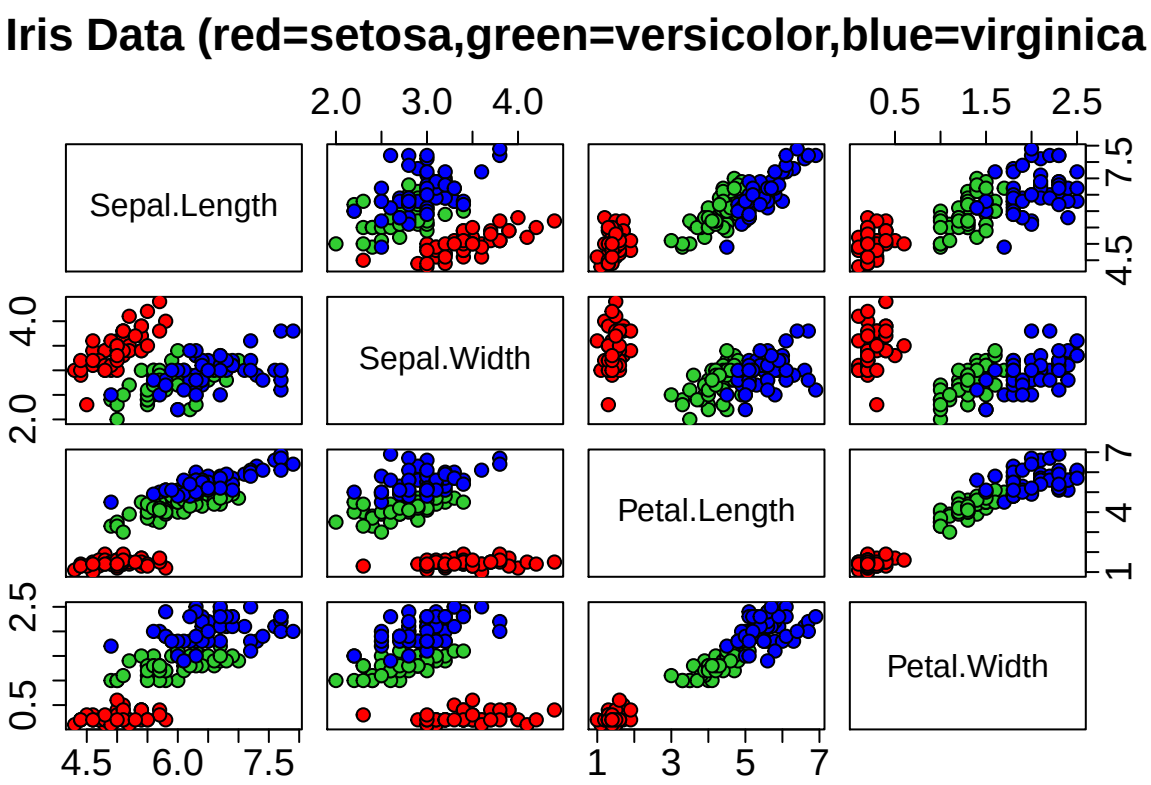
```
rotateMatrix270(myMatrix)
```

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

2 Iris Scatterplot Matrix

2.1 Recreate the Wikipedia scatterplot matrix exactly.

```
#color designations for species
my_cols <- c("red", "lime green", "blue")

pairs(~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width, data=iris, main="Iris Data (red=setosa,green=
```



3 Iris Data Set description

This data set was published in 1936, and is so popular for computer statistical programming, that it is built into R. It consists of 50 samples of 3 different types of irises, with four measurements: sepal length, sepal width, petal length, and petal width, for a total of 150 observations.

The three iris species are Iris setosa, versicolor, and virginica. Setosa grows in the largest geographic region: a large area from Japan to coastal Alaska, Canada, and dips into the Great Lakes region of the United States. Iris versicolor grows from mainly in the north east coast from northern Virginia to Canada, and from Winnipeg, Canada. It's habitat is considerably smaller than Iris Setosa. Iris virginica grows from Virginia southward along the coast to the border of Florida. Another way to describe the region is that virginica grows in Ozark-Appalachian landmass.

The model which was used to create the ideograms for each flower is essentially a sepal superimposed on a petal. The ideogram is a black rectangle with a white rectangle inside it. The rectangles are drawn from the height and width of the sepal and petal. The ideogram, in a single image, shows the four measurements (sepal length & width, petal length& width), along with the relation the sepal and petal size have to each other (how large the petal is vs how small the sepal is, some iris have a greater disparity in size between the two, others have a smaller disparity).

4 Import personality-raw.txt into R

Number of records in personality-raw.txt: 838 Number of records in personality_clean.txt: 678

```
library(readr)
library(dplyr)
```

```
#read txt file, remove V00
personality_dat<-read_delim("personality-raw.txt", delim='|')
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   md5_email = col_character(),
##   date_test = col_character()
## )

## See spec(...) for full column specifications.
```

```
remove_V00_dat<-subset(personality_dat, select = -c(V00))

#remove date_test column to turn into year and week columns
date_col<-remove_V00_dat[[2]]

just_date<-gsub(" .*", "", date_col)
formatted_date<-as.Date(just_date, format="%m/%d/%Y")

year_col<-format(formatted_date, format="%Y")
week_col<-format(formatted_date, format="%V")

#remove date_test column from remove_V00_dat and create a frame
remove_date_col<-subset(remove_V00_dat, select=-c(date_test))
asframe<-as.data.frame(remove_date_col, stringsAsFactors=FALSE)

#add year and week columns to remove_dat_col
added_year<-cbind(asframe, Year=year_col)
added_week<-cbind(added_year, Week=week_col)
reorder<- added_week[, colnames(added_week)[c(1, 62, 63, 2:61 )]]

#descending order by date
descending<-reorder[order(-xtfrm(reorder[,2]), -xtfrm(reorder[,3])), ]

#filter for duplicates and finalize cleaned data
personality_clean<- descending %>% distinct(md5_email, .keep_all=TRUE)

#export to txt file
write.table(personality_clean, file="personality_clean.txt",
            sep="|", dec=".", row.names=FALSE, col.names=TRUE)
```

5 Custom functions for doSummary, sampleVariance, and doMode

Using personality_clean.txt to illustrate the functions

```
library(readr)
library(dplyr)

#read personality_clean.txt and extract first row of data for summary statistics
personality_clean<-read_delim("personality_clean.txt",delim='|')
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   md5_email = col_character(),
##   Week = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

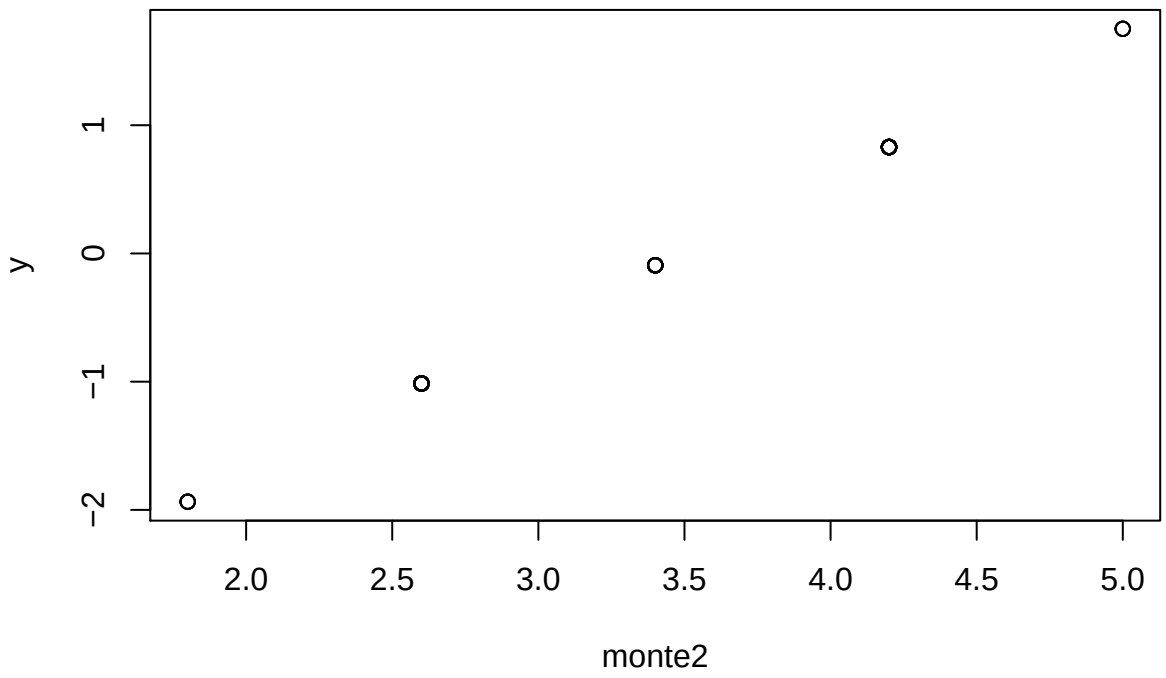
```
monte<-personality_clean %>% slice(1:1)
monte2<- as.double(subset(monte, select=-c(md5_email, Year, Week)))

doSummary(monte2)
```

```
##        length number of NAs          mean        median          mode
##    60.0000000     0.0000000     3.4800000     3.4000000     4.2000000
##       var.sum     var.sumSq       var.var   built-in sd custom sd.var
##   208.8000000   771.0400000     0.7528136     0.8676483     0.8676483
```

6 Create z-scores and plot

The pattern is a straight, diagonal line. Simply, the z-scores are linearly transformed scores from the raw scores. The raw scores are within two standard deviations, and the points are roughly equidistant from each other.

```
y<-doZScore(monte2)
plot(monte2,y)
```



7 Compare Will Smith and Denzel Washington

I converted raw value into 2020 value. For example, in 1995, $7.92 million is the equivalent to $13.465 million dollars today in 2020. This was, by far, the most difficult for me because I had trouble understanding how the inflation rate worked. (I did have my husband help me with the math, and it took him some time as well. I am not sure why it was so difficult, perhaps we just weren't seeing what was, in essence, simple).

```
library(readr)
library(dplyr)

inflation<-read_delim("inflation.txt",delim='|')
```

```
## Parsed with column specification:
## cols(
##   year = col_double(),
##   dollar = col_double(),
##   inflation = col_double()
## )
```

```
#reverse columns
inflation<-inflation[rev(1:nrow(inflation)), ]

#get deflation rate
adjusted<-getDeflationRate(inflation)
```

```
## Warning: The `i` argument of ``[.tbl_df`()` must lie in [0, rows] if positive, as of tibble 3.0.0.
## Use `NA_integer_` as row index to obtain a row full of `NA` values.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

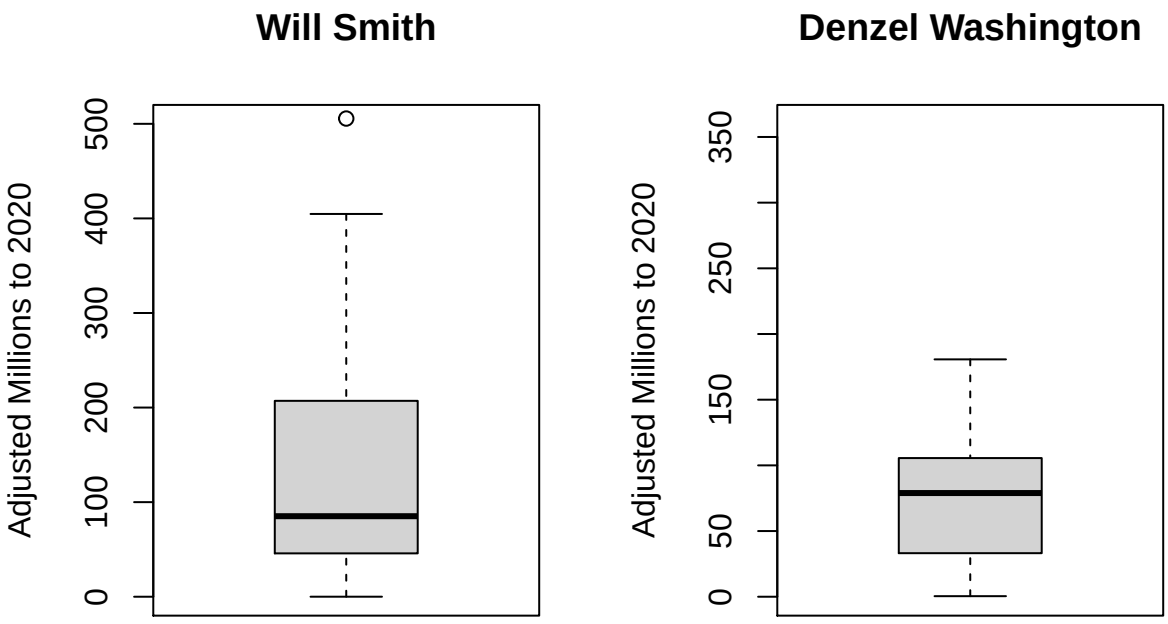
```
adjusted<-as.vector(unlist(adjusted))
adjusted<-data.frame(adjusted) #converted to dataframe
inflation<-cbind(inflation, adjusted) #added adjusted rate to inflation dataframe

#prepped the inflation data set shift adjust down by 1
inflation<-inflation %>% mutate_at(c("adjusted"), list(lag),n=1)
inflation<-inflation %>% replace(is.na(.),1)
inflation<-inflation[rev(1:nrow(inflation)), ] #reverse order back to ascending

#prep will and denzel data
#will data frame for raw year and gross
nmid = "nm0000226";
will = grabFilmsForPerson(nmid)
will.gross<-will$movies.50$millions
will.year<-will$movies.50$year
will.raw<-data.frame(will.year, will.gross)
colnames(will.raw) = c("year", "dollar")
clean.will.raw<-na.omit(will.raw)

#denzel data frame for raw year and gross
nmid = "nm0000243";
denzel = grabFilmsForPerson(nmid);
denzel.gross<-denzel$movies.50$millions
denzel.year<-denzel$movies.50$year
```

```
denzel.raw<-data.frame(denzel.year, denzel.gross)
colnames(denzel.raw) = c("year", "dollar")
clean.denzel.raw<-na.omit(denzel.raw)

#get will and denzel movie year matched to index position of inflation year
m<-as.numeric(match(clean.will.raw$year, inflation$year)) #matches the will year to inflation
d<-as.numeric(match(clean.denzel.raw$year, inflation$year)) #matches denzel year to inflation table

#get inflation rate equivalent by year of movies
final.rate.will<-sapply(m, FUN=getRate2020) #m (will)
final.rate.denzel<-sapply(d, FUN=getRate2020) #d (denzel)

#add final.rate to will.raw.clean
will.2<-cbind(clean.will.raw,final.rate.will)
#change will dollar amount with final rate
will.2$dollar_adjusted=will.2$dollar*will.2$final.rate

#final rate to denzel.raw.clean
denzel.2<-cbind(clean.denzel.raw, final.rate.denzel)
denzel.2$dollar_adjusted=denzel.2$dollar*denzel.2$final.rate
```

8 Will Smith vs Denzel Washington Side by Side Box Plots

Boxplot highlights: The interquartile range is the 25th to 75th percentile of data points, and then the whiskers are Q1 and Q4. Dots are the outliers. The line represents the median.

```
par(mfrow=c(1,2));
boxplot(will.2$dollar_adjusted, main=will$name, ylim=c(0,500), ylab="Adjusted Millions to 2020")
boxplot(denzel.2$dollar_adjusted, main=denzel$name, ylim=c(0,360), ylab="Adjusted Millions to 2020")
```

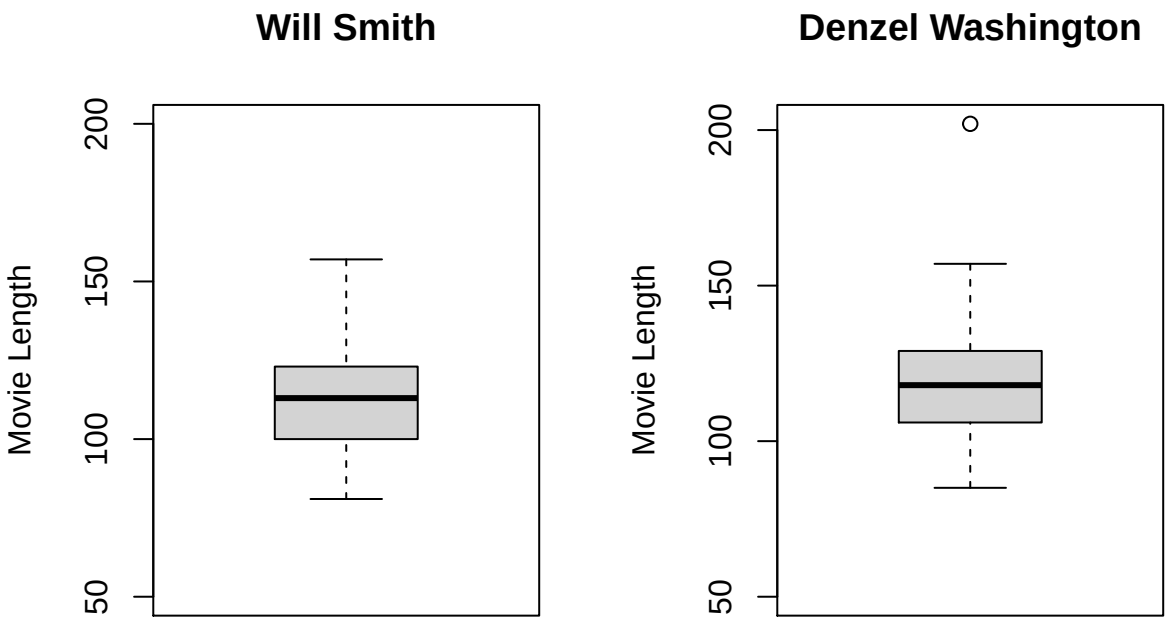


The boxplot above is the adjusted to 2020 dollar value. The raw boxplot implies that Will Smith movies made considerably more than Denzel, and this is true even for the adjusted value. However, adjusting for value, Denzel movies do make more than the raw value implies. Will Smith movie gross sales has a far wider range than Denzels. His median is also higher than Will's. Will Smith has one outlier at $500mil. This

```
par(mfrow=c(1,2));
boxplot(will$movies.50$ratings, main=will$name, ylim=c(1,10), ylab="Ratings 1-10")
boxplot(denzel$movies.50$ratings, main=denzel$name, ylim=c(1,10), ylab="Ratings 1-10")
```

This side by side boxplot represents the ratings of the movies. Again, Denzel movies are rated in general higher than Will Smith's, due to median and bottom whisker. Will Smith has a 2 rating outlier.

```
par(mfrow=c(1,2));
boxplot(will$movies.50$minutes, main=will$name, ylim=c(50,200), ylab="Movie Length")
boxplot(denzel$movies.50$minutes, main=denzel$name, ylim=c(50,202), ylab="Movie Length")
```



This side by side boxplot represents the length of movies. Not surprisingly, these boxplots are similar because movie lengths have a narrower range than other factors like gross sales. Denzel's median movie is higher in length and the bottom whisker is higher than Will's. Overall, Denzel makes longer movies than Will.

8.1 Conclusion

In general, these boxplots visually show how compact Denzel's statistics are compared to Will Smith. Will Smith has a wider diversity/range in movies – in terms of sales, ratings, and even movie length. He has grossed movies much higher than Denzel, but he has also had some poorly grossing movies. Same with both movie length, but especially in terms of ratings.