HW13

107070008

Question 1) Let's revisit the issue of multicollinearity of main effects (between cylinders, displacement, horsepower, and weight) we saw in the cars dataset, and try to apply principal components to it. Start by recreating the cars_log dataset, which log-transforms all variables except model year and origin.

```
auto <- read.table("auto-data.txt", header=FALSE, na.strings = "?")
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower",
                 "weight", "acceleration", "model_year", "origin", "car_name")
cars <- na.omit(auto)

cars_log <- with(cars, data.frame(log(mpg), log(cylinders), log(displacement), log(acceleration),
                log(horsepower), log(weight), model_year, factor(origin)))
head(cars_log)
```

```
##   log.mpg. log.cylinders. log.displacement. log.acceleration. log.horsepower.
## 1 2.890372       2.079442          5.726848          2.484907        4.867534
## 2 2.708050       2.079442          5.857933          2.442347        5.105945
## 3 2.890372       2.079442          5.762051          2.397895        5.010635
## 4 2.772589       2.079442          5.717028          2.484907        5.010635
## 5 2.833213       2.079442          5.710427          2.351375        4.941642
## 6 2.708050       2.079442          6.061457          2.302585        5.288267
##   log.weight. model_year factor.origin.
## 1    8.161660         70              1
## 2    8.214194         70              1
## 3    8.142063         70              1
## 4    8.141190         70              1
## 5    8.145840         70              1
## 6    8.375860         70              1
```

a. Let's analyze the principal components of the four collinear variables.

i. Create a new data.frame of the four log-transformed variables with high multicollinearity (Give this smaller data frame an appropriate name – what might they jointly mean?)

```
var4_log <- with(cars, data.frame(log(cylinders),
            log(displacement), log(horsepower), log(weight)))
```

ii. How much variance of the four variables is explained by their first principal component? (a summary of the prcomp() shows it, but try computing this from the eigenvalues alone)

```
round(cor(var4_log), 2)
```

```
##                    log.cylinders. log.displacement. log.horsepower. log.weight.
## log.cylinders.               1.00              0.95            0.83        0.88
## log.displacement.            0.95              1.00            0.87        0.94
## log.horsepower.              0.83              0.87            1.00        0.87
## log.weight.                  0.88              0.94            0.87        1.00
```

```
eigen <- eigen(cor(var4_log))
eigen$values
```

```
## [1] 3.67425879 0.18762771 0.10392787 0.03418563
```

iii. Looking at the values and valence (positiveness/negativeness) of the first principal component's eigenvector, what would you call the information captured by this component? (i.e., think what concept the first principal component captures or represents)

```
Move the center of the coordinate axis to the center of the data, and then rotate the
coordinate axis, so that the variance of the data on the C1 axis is the largest, that is,
the projections of all n data individuals in this direction are the most dispersed. Means
more information is preserved. C1 becomes the first principal component.
```

b.Let's revisit our regression analysis on cars_log:

i. Store the scores of the first principal component as a new column of cars_log Give this new column a name suitable for what it captures (see 1.a.i.)

```
cars_log <- na.omit(cars_log)
pca <- prcomp(var4_log, scale. = TRUE)
cars_log$PC1 <- pca$x[,1]
head(cars_log)
```

```
##   log.mpg. log.cylinders. log.displacement. log.acceleration. log.horsepower.
## 1 2.890372       2.079442          5.726848          2.484907        4.867534
## 2 2.708050       2.079442          5.857933          2.442347        5.105945
## 3 2.890372       2.079442          5.762051          2.397895        5.010635
## 4 2.772589       2.079442          5.717028          2.484907        5.010635
## 5 2.833213       2.079442          5.710427          2.351375        4.941642
## 6 2.708050       2.079442          6.061457          2.302585        5.288267
##   log.weight. model_year factor.origin.       PC1
## 1    8.161660         70              1 -2.036645
## 2    8.214194         70              1 -2.593998
## 3    8.142063         70              1 -2.237767
## 4    8.141190         70              1 -2.192902
## 5    8.145840         70              1 -2.097313
## 6    8.375860         70              1 -3.337215
```

ii. Regress mpg over the column with PC1 scores (replacing cylinders, displacement, horsepower, and weight), as well as acceleration, model_year and origin.

```
summary(lm(log.mpg. ~ PC1 + log.acceleration. +
             model_year + factor.origin., data = cars_log))
```

```
##
## Call:
## lm(formula = log.mpg. ~ PC1 + log.acceleration. + model_year +
##     factor.origin., data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51137 -0.06050 -0.00183  0.06322  0.46792
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.398114   0.166554   8.394 8.99e-16 ***
## PC1                0.145663   0.005057  28.804  < 2e-16 ***
## log.acceleration. -0.191482   0.041722  -4.589 6.02e-06 ***
## model_year         0.029180   0.001810  16.122  < 2e-16 ***
## factor.origin.2    0.008272   0.019636   0.421    0.674
## factor.origin.3    0.019687   0.019395   1.015    0.311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1199 on 386 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.8756
## F-statistic: 551.6 on 5 and 386 DF,  p-value: < 2.2e-16
```

iii. Try running the regression again over the same independent variables, but this time with everything standardized. How important is this new column relative to other columns?

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

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

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

```
cars_log_std <- cars_log %>% mutate_if(is.numeric,scale)
summary(lm(log.mpg. ~ PC1 + log.acceleration. + model_year + factor.origin., data = cars_log_std))
```

```
##
## Call:
## lm(formula = log.mpg. ~ PC1 + log.acceleration. + model_year +
##     factor.origin., data = cars_log_std)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -1.50385 -0.17791 -0.00538  0.18591  1.37608 
## 
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)    
## (Intercept)         -0.01589    0.02563  -0.620    0.536    
## PC1                  0.82112    0.02851  28.804  < 2e-16 ***
## log.acceleration.   -0.10190    0.02220  -4.589 6.02e-06 ***
## model_year           0.31611    0.01961  16.122  < 2e-16 ***
## factor.origin.2      0.02433    0.05775   0.421    0.674    
## factor.origin.3      0.05790    0.05704   1.015    0.311    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.3526 on 386 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.8756 
## F-statistic: 551.6 on 5 and 386 DF,  p-value: < 2.2e-16
```

```
The new column is 0.1% significance to other columns.
```

Question 2) Please download the Excel data file security_questions.xlsx from Canvas. In your analysis, you can either try to read the data sheet from the Excel file directly from R (there might be a package for that!) or you can try to export the data sheet to a CSV file before reading it into R.

```r
library(readxl)
sa <- read_excel("security_questions.xlsx")
```

a. How much variance did each extracted factor explain?

```r
sa_eigen <- eigen(cor(sa))
sa_eigen$values/sum(sa_eigen$values)
```

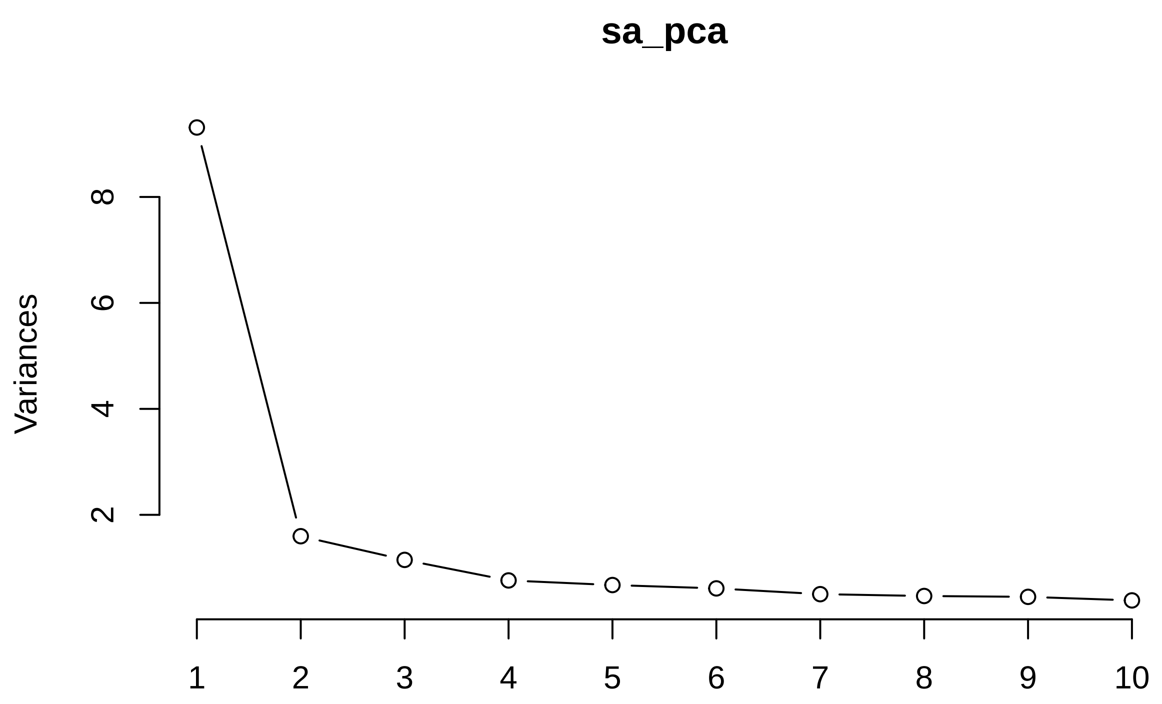
```
##  [1] 0.51727518 0.08868511 0.06386435 0.04233199 0.03750784 0.03398131
##  [7] 0.02794364 0.02601549 0.02510951 0.02139980 0.01971565 0.01673928
## [13] 0.01623763 0.01456354 0.01303216 0.01280357 0.01159706 0.01119690
```

b. How many dimensions would you retain, according to the two criteria we discussed? (Eigenvalue >= 1 and Scree Plot – can you show the screeplot with eigenvalue=1 threshhold?)

```r
which(sa_eigen$values > 1)
```

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

```r
sa_pca <- prcomp(sa, scale. = TRUE)
screeplot(sa_pca, type="lines")
```



`one.` **(ungraded) Can you interpret what any of the principal components mean? Try guessing the meaning of the first two or three PCs looking at the PC-vs-variable matrix** `Principal components means that they can linear transform obeservation values.`

Question 3) Let's simulate how principal components behave interactively

```r
library(devtools)
```

```
## Loading required package: usethis
```

```r
install_github("soumyaray/compstatslib")
```

```
## Skipping install of 'compstatslib' from a github remote, the SHA1 (633d9772) has not changed since la
##   Use `force = TRUE` to force installation
```

```r
library(compstatslib)
```