BACS HW14

109062710

5/24/2021

Prepare the data set

```
cars_log <- with(Auto, data.frame(log(mpg), log(cylinders), log(displacement),
log(horsepower), log(weight), log(acceleration), year, origin, name))
weight_mean <- mean(cars_log$weight)
```

```
## Warning in mean.default(cars_log$weight): argument is not numeric or logical:
## returning NA
```

```
names(cars_log) <- names(Auto)
head(cars_log)
```

```
##        mpg cylinders displacement horsepower   weight acceleration year origin
## 1 2.890372  2.079442     5.726848   4.867534 8.161660     2.484907   70      1
## 2 2.708050  2.079442     5.857933   5.105945 8.214194     2.442347   70      1
## 3 2.890372  2.079442     5.762051   5.010635 8.142063     2.397895   70      1
## 4 2.772589  2.079442     5.717028   5.010635 8.141190     2.484907   70      1
## 5 2.833213  2.079442     5.710427   4.941642 8.145840     2.351375   70      1
## 6 2.708050  2.079442     6.061457   5.288267 8.375860     2.302585   70      1
##                        name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

Convert the numbers in `origin` column into names, namely 1 for USA, 2 for Europe, and 3 for Japan.

```
origins <- c("USA", "Europe", "Japan")
cars_log$origin <- factor(cars_log$origin, labels = origins)
```

Question 1

a. Compute direct effects

i. Regress `weight` over `cylinders`.

```
weight_cyl_regr <- lm(weight ~ cylinders, data = cars_log)
summary(weight_cyl_regr)
```

```
##
## Call:
## lm(formula = weight ~ cylinders, data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.35409 -0.09030 -0.00169  0.09271  0.40488
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.60059    0.03710  177.92   <2e-16 ***
## cylinders    0.82187    0.02208   37.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1319 on 390 degrees of freedom
## Multiple R-squared:  0.7804, Adjusted R-squared:  0.7798
## F-statistic:  1386 on 1 and 390 DF,  p-value: < 2.2e-16
```

In this case, just by looking at `Pr(>|t|)` column, number of `cylinders` has a significant effect on `weight`.

ii. Regress `mpg` over `weight` + control variables.

```
mpg_all_regr <- lm(mpg ~ weight + acceleration + year + origin, data = cars_log)
summary(mpg_all_regr)
```

```
##
## Call:
## lm(formula = mpg ~ weight + acceleration + year + origin, data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38259 -0.07054  0.00401  0.06696  0.39798
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.410974   0.316806  23.393  < 2e-16 ***
## weight       -0.875499   0.029086 -30.101  < 2e-16 ***
## acceleration  0.054377   0.037132   1.464  0.14389
## year          0.032787   0.001731  18.937  < 2e-16 ***
## originEurope  0.056111   0.018241   3.076  0.00225 **
## originJapan   0.031937   0.018506   1.726  0.08519 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1163 on 386 degrees of freedom
## Multiple R-squared:  0.8845, Adjusted R-squared:  0.883
## F-statistic: 591.1 on 5 and 386 DF,  p-value: < 2.2e-16
```

In this case, only `weight` and `year` have significant direct effect on `mpg`.

b. What is the indirect effect of cylinders on mpg?

```
mpg_all_regr$coefficients
```

```
##  (Intercept)       weight acceleration         year originEurope  originJapan
##   7.41097361  -0.87549901   0.05437701   0.03278658   0.05611103   0.03193692
```

```
weight_cyl_regr$coefficients
```

```
## (Intercept)   cylinders
##   6.6005907   0.8218704
```

```
weight_cyl_regr$coefficients[2] * mpg_all_regr$coefficients[2]
```

```
##  cylinders
## -0.7195467
```

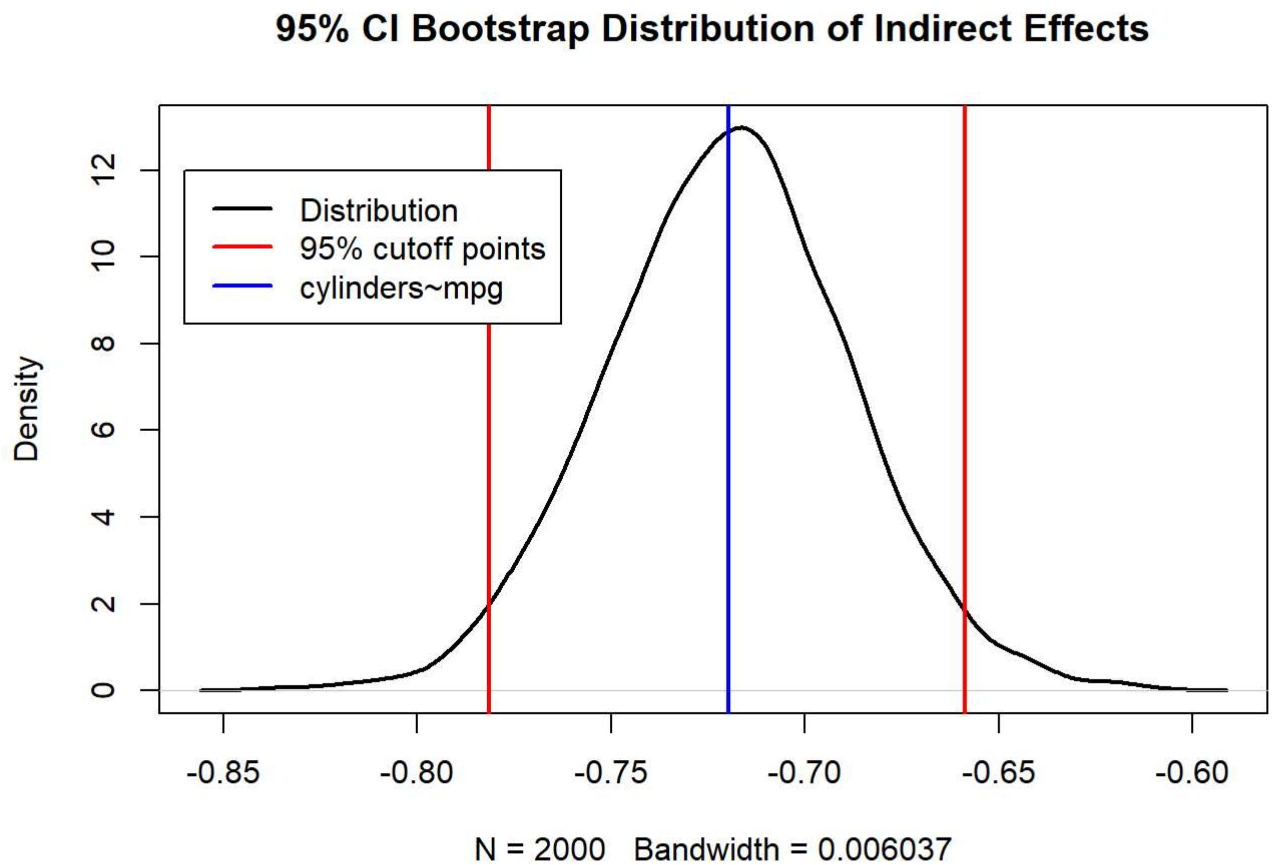
c. Bootstrap CI of the indirect effect of cylinders on mpg

```
set.seed(345)
boot_indirect <- function(model1, model2, data) {
    random_index <- sample(1:nrow(data), replace = TRUE)
    random_sample <- data[random_index,]
    lm1 <- lm(model1, data = random_sample)
    lm2 <- lm(model2, data = random_sample)
    return(lm1$coefficients[2] * lm2$coefficients[2])
}

bootstrap_ci <- replicate(
    2000,
    boot_indirect(weight ~
                      cylinders,
                  mpg ~
                      weight +
                      acceleration +
                      year +
                      origin,
                  cars_log)
    )
plot(
    density(bootstrap_ci),
    main =
        '95% CI Bootstrap Distribution of Indirect Effects',
    lwd = 2,
)

abline(
    v = quantile(bootstrap_ci, p = c(0.025, 0.975)),
    col = "red",
    lty = "solid",
    lwd = 2
)

abline(
    v = weight_cyl_regr$coefficients[2] * mpg_all_regr$coefficients[2],
    col = "blue",
    lty = "solid",
    lwd = 2
)
legend(
    -0.86,
    12,
    c("Distribution", "95% cutoff points", "cylinders~mpg"),
    col = c("black", "red", "blue"),
    lwd = c(2,2,2),
    lty = c(1,1,1)
)
```



We can see that the indirect effect we got from `weight_cyl_regr$coefficients[2] * mpg_all_regr$coefficients[2]` falls in within the 95% CI.

Question 2

a. Analyze the principal components of the four colinear variables

i. Make a new data.frame of the four log-transformed variables with high multi-collinearity

```
multicollinear_variables <- cars_log[, c("cylinders", "displacement", "horsepower", "weight")]
multicollinear_variables <- na.omit(multicollinear_variables)
head(multicollinear_variables)
```

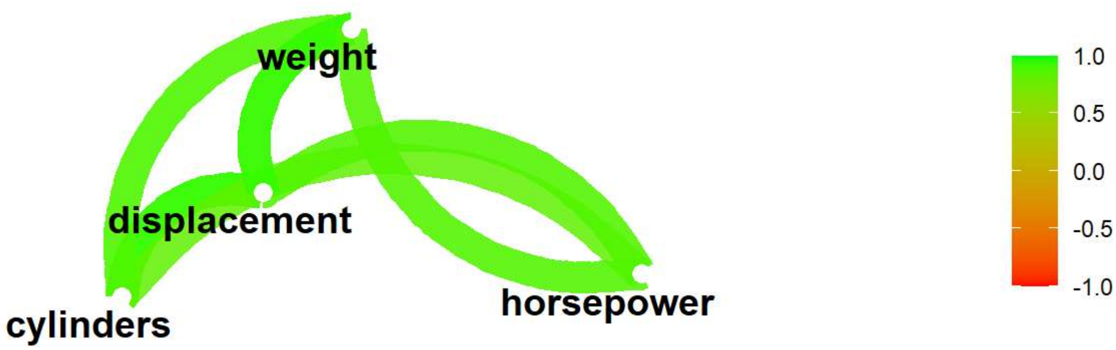
```
##   cylinders displacement horsepower   weight
## 1  2.079442     5.726848   4.867534 8.161660
## 2  2.079442     5.857933   5.105945 8.214194
## 3  2.079442     5.762051   5.010635 8.142063
## 4  2.079442     5.717028   5.010635 8.141190
## 5  2.079442     5.710427   4.941642 8.145840
## 6  2.079442     6.061457   5.288267 8.375860
```

```
cor(multicollinear_variables)
```

```
##              cylinders displacement horsepower    weight
## cylinders    1.0000000    0.9469109  0.8265831 0.8833950
## displacement 0.9469109    1.0000000  0.8721494 0.9428497
## horsepower   0.8265831    0.8721494  1.0000000 0.8739558
## weight       0.8833950    0.9428497  0.8739558 1.0000000
```

```
multicollinear_variables %>%
  correlate() %>%
  network_plot(min_cor = 0.7, colors = c("red", "green"), legend = TRUE)
```

```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```



From the graph above, we know these four variables are multi-collinear

ii. How much variance of the four variables is explained by their first principal component?

```
prcomp <- prcomp(multicollinear_variables, scale. = TRUE)

summary(prcomp)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4
## Standard deviation     1.9168 0.43316 0.32238 0.18489
## Proportion of Variance 0.9186 0.04691 0.02598 0.00855
## Cumulative Proportion  0.9186 0.96547 0.99145 1.00000
```

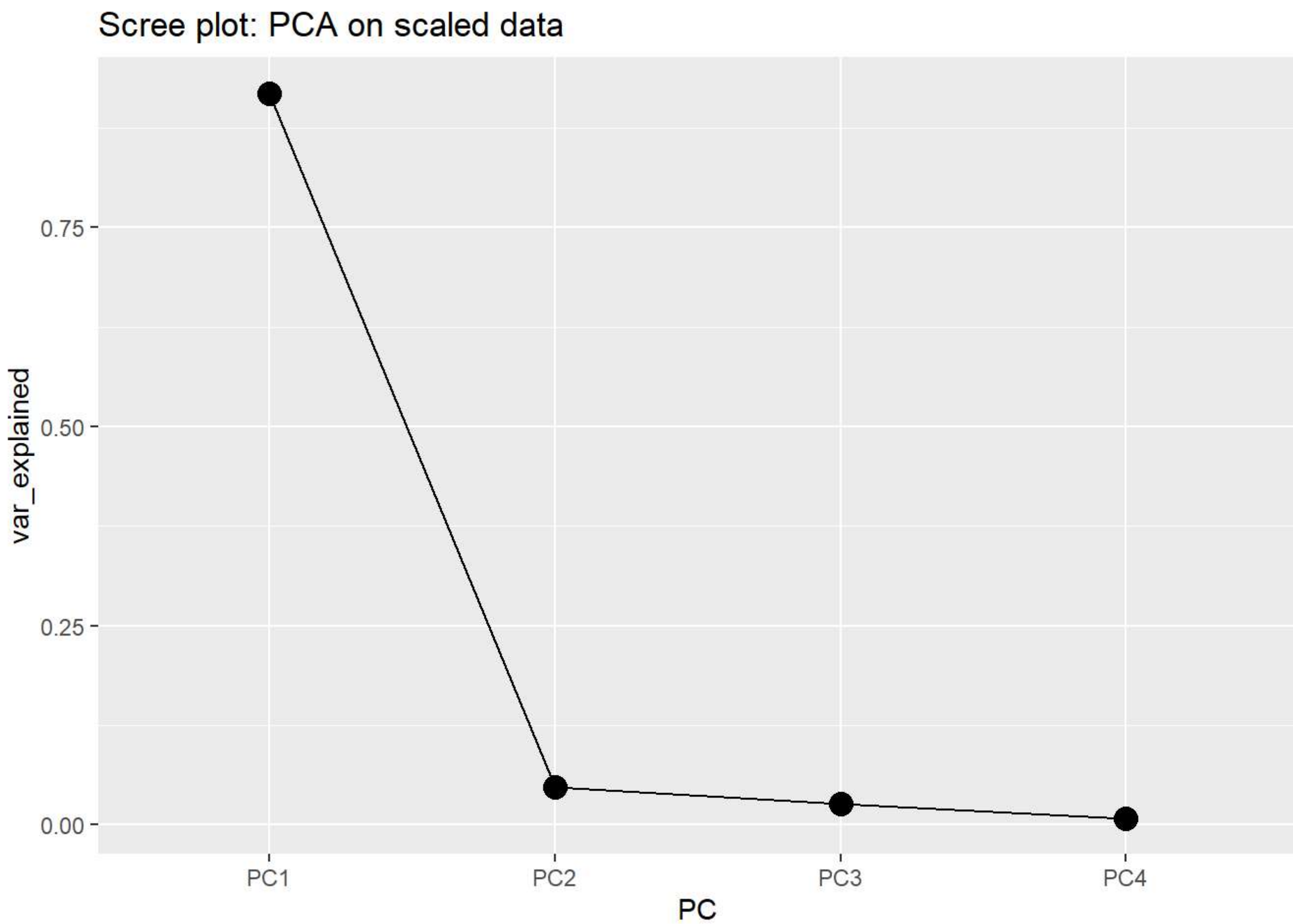
```
var_explained <- (prcomp$sdev)^2/sum((prcomp$sdev)^2)

var_explained_df <- data.frame(
    PC= paste0("PC",1:4),
    var_explained=var_explained)

var_explained_df
```

```
##    PC var_explained
## 1 PC1   0.918564696
## 2 PC2   0.046906929
## 3 PC3   0.025981967
## 4 PC4   0.008546408
```

```
var_explained_df %>%
  ggplot(aes(x=PC,y=var_explained, group=1))+
  geom_point(size=4)+
  geom_line()+
  labs(title="Scree plot: PCA on scaled data")
```



Clearly, the first principal component explains the most out of all variables present in multicollinear_variables data set.

iii. What would you call the information captured by this component?

b. Let's revisit our regression analysis on cars_log

i. Store the PC1 scores in the new column of `cars_log`

```
new_cars_log <- cars_log[, c(1:7)]
new_cars_log <- na.omit(new_cars_log)

pc1 <- prcomp$x

new_cars_log$PC1 <- pc1[, 1]
head(new_cars_log)
```

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

ii. Regress mpg over the the column with `PC1`, `acceleration`, `year` and `origin`

```
summary(
    lm(
        mpg ~
            PC1 +
            acceleration +
            year +
            Auto$origin,
        data = new_cars_log
    )
)
```

```
##
## Call:
## lm(formula = mpg ~ PC1 + acceleration + year + Auto$origin, data = new_cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51070 -0.06039 -0.00161  0.06271  0.46795
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.386083   0.166466   8.327 1.45e-15 ***
## PC1          0.145547   0.004886  29.786  < 2e-16 ***
## acceleration -0.191608   0.041645  -4.601 5.71e-06 ***
## year         0.029210   0.001776  16.444  < 2e-16 ***
## Auto$origin  0.009815   0.009680   1.014    0.311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1198 on 387 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.876
## F-statistic: 691.3 on 4 and 387 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?

```
standardized_cars_log <- as.data.frame(scale(new_cars_log))

summary(
    lm(
        mpg ~
            PC1 +
            acceleration +
            year +
            Auto$origin,
        data = standardized_cars_log
    )
)
```

```
##
## Call:
## lm(formula = mpg ~ PC1 + acceleration + year + Auto$origin, data = standardized_cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50188 -0.17759 -0.00472  0.18442  1.37615
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.04551    0.04828  -0.943    0.346
## PC1           0.82046    0.02755  29.786  < 2e-16 ***
## acceleration -0.10197    0.02216  -4.601 5.71e-06 ***
## year          0.31644    0.01924  16.444  < 2e-16 ***
## Auto$origin   0.02886    0.02847   1.014    0.311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3522 on 387 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.876
## F-statistic: 691.3 on 4 and 387 DF,  p-value: < 2.2e-16
```

Whether the data is standardized or not, it doesn't seem that `PC1` shows any importance over any over variables. However, intercept becomes less significant

Question 3

Import data set

```
security_questions <-
    read_csv("D:/git-repos/bacs-hw/hw14/security_questions.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Q1 = col_double(),
##   Q2 = col_double(),
##   Q3 = col_double(),
##   Q4 = col_double(),
##   Q5 = col_double(),
##   Q6 = col_double(),
##   Q7 = col_double(),
##   Q8 = col_double(),
##   Q9 = col_double(),
##   Q10 = col_double(),
##   Q11 = col_double(),
##   Q12 = col_double(),
##   Q13 = col_double(),
##   Q14 = col_double(),
##   Q15 = col_double(),
##   Q16 = col_double(),
##   Q17 = col_double(),
##   Q18 = col_double()
## )
```

```
head(security_questions)
```

```
## # A tibble: 6 x 18
##      Q1    Q2    Q3    Q4    Q5    Q6    Q7    Q8    Q9   Q10   Q11   Q12   Q13
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     7     5     5     7     7     4     4     7     5     7     5     7     5
## 2     5     5     6     6     6     5     5     7     5     6     6     6     6
## 3     6     6     6     6     7     6     6     6     5     7     6     6     5
## 4     5     5     5     5     5     5     5     5     5     5     5     5     4
## 5     7     7     7     7     7     4     5     7     6     7     6     7     6
## 6     6     5     4     5     4     4     4     5     6     2     5     5     5
## # ... with 5 more variables: Q14 <dbl>, Q15 <dbl>, Q16 <dbl>, Q17 <dbl>,
## #   Q18 <dbl>
```

a. How much variance did each extracted factor explain?

```
sec_ques_prcomp <- prcomp(security_questions, scale. = TRUE)
```

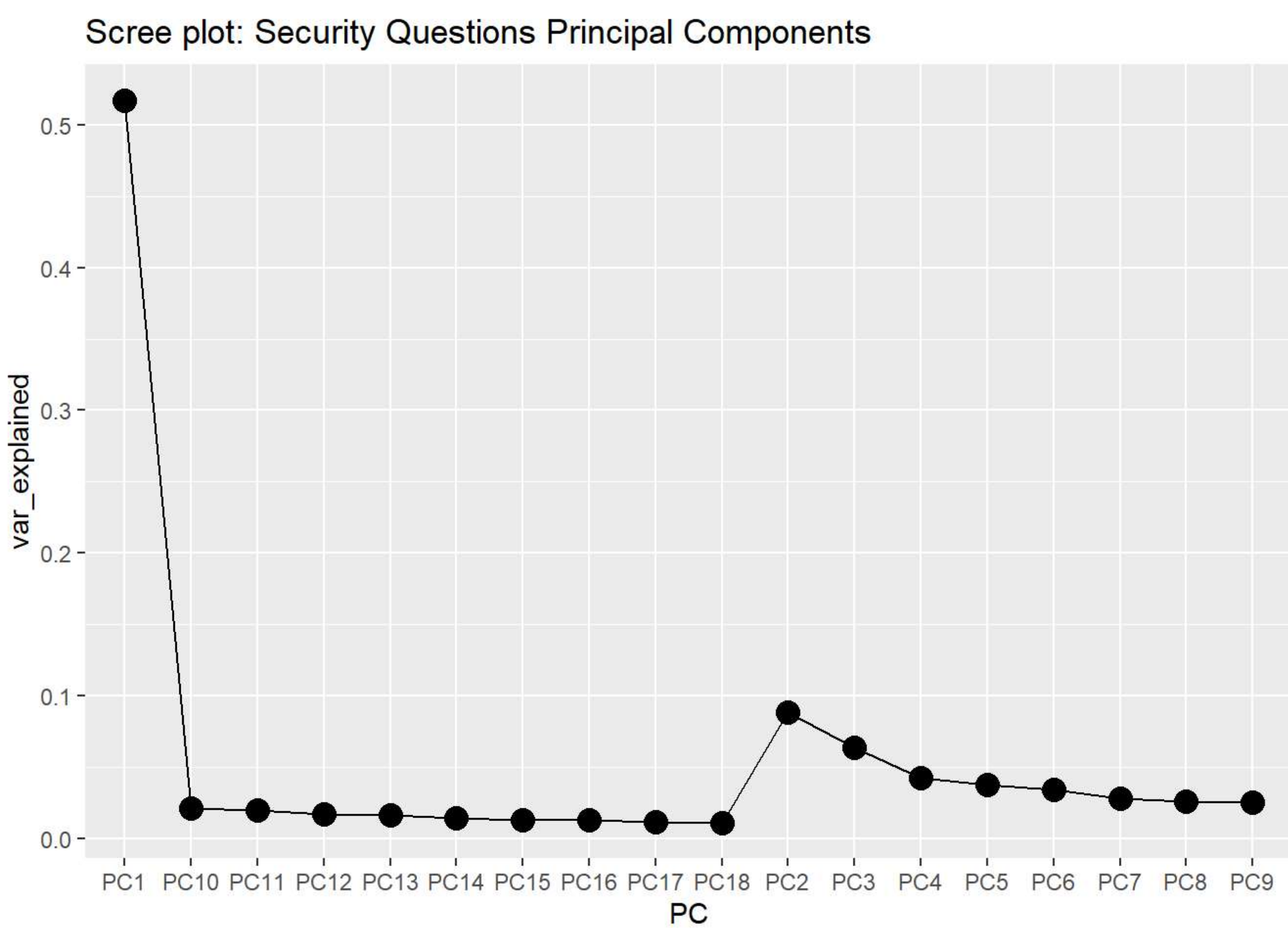
```
var_explained <- (sec_ques_prcomp$sdev)^2/sum((sec_ques_prcomp$sdev)^2)

var_explained_df <- data.frame(
    PC= paste0("PC", 1:18),
    var_explained=var_explained)

var_explained_df
```

```
##      PC var_explained
## 1   PC1    0.51727518
## 2   PC2    0.08868511
## 3   PC3    0.06386435
## 4   PC4    0.04233199
## 5   PC5    0.03750784
## 6   PC6    0.03398131
## 7   PC7    0.02794364
## 8   PC8    0.02601549
## 9   PC9    0.02510951
## 10 PC10    0.02139980
## 11 PC11    0.01971565
## 12 PC12    0.01673928
## 13 PC13    0.01623763
## 14 PC14    0.01456354
## 15 PC15    0.01303216
## 16 PC16    0.01280357
## 17 PC17    0.01159706
## 18 PC18    0.01119690
```

```
var_explained_df %>%
  ggplot(aes(x=PC,y=var_explained, group=1))+
  geom_point(size=4)+
  geom_line()+
  labs(title="Scree plot: Security Questions Principal Components")
```



b. How many dimensions would you retain, according to the criteria we discussed?

```
summary(sec_ques_prcomp)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.0514 1.26346 1.07217 0.87291 0.82167 0.78209 0.70921
## Proportion of Variance 0.5173 0.08869 0.06386 0.04233 0.03751 0.03398 0.02794
## Cumulative Proportion  0.5173 0.60596 0.66982 0.71216 0.74966 0.78365 0.81159
##                            PC8     PC9   PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.68431 0.67229 0.6206 0.59572 0.54891 0.54063 0.51200
## Proportion of Variance 0.02602 0.02511 0.0214 0.01972 0.01674 0.01624 0.01456
## Cumulative Proportion  0.83760 0.86271 0.8841 0.90383 0.92057 0.93681 0.95137
##                           PC15   PC16   PC17   PC18
## Standard deviation     0.48433 0.4801 0.4569 0.4489
## Proportion of Variance 0.01303 0.0128 0.0116 0.0112
## Cumulative Proportion  0.96440 0.9772 0.9888 1.0000
```

I would take 2 components which are `PC1` and `PC2` since two of them capture almost 60 percent variance of the data set.

c.Can you interpret what any of the principal components mean?

Geometrically speaking, principal components represent the directions of the data that explain a maximal amount of variance, that is to say, the lines that capture most information of the data.

The relationship between variance and information here, is that, the larger the variance carried by a line, the larger the dispersion of the data points along it, and the larger the dispersion along a line, the more the information it has.

Simply said, just think of principal components as new axes that provide the best angle to see and evaluate the data, so that the differences between the observations are better visible.