hw13

111078513

Help By 108078467

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. Important: remove any rows that have missing values.

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

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

```
round(cor(cars_log[, 2:8]), 2) # log.cylinders, log.displacement. , log.horsepower. , log.weight.
```

```
##                   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
## log.acceleration.          -0.50             -0.52            -0.72       -0.42
## model_year                 -0.34             -0.33            -0.40       -0.29
## origin                     -0.58             -0.67            -0.48       -0.61
##                   log.acceleration. model_year origin
## log.cylinders.                -0.50      -0.34  -0.58
## log.displacement.             -0.52      -0.33  -0.67
## log.horsepower.               -0.72      -0.40  -0.48
## log.weight.                   -0.42      -0.29  -0.61
## log.acceleration.              1.00       0.31   0.23
## model_year                     0.31       1.00   0.18
## origin                         0.23       0.18   1.00
```

```
cars_log_multicolline <- with(cars_log,
                              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?

```
cars_log_multicolline_eigen <- eigen(cov(cars_log_multicolline))
cars_log_multicolline_eigen$values[1]/sum(cars_log_multicolline_eigen$values)
```

```
## [1] 0.9346169
```

```
cars_log_multicolline_pca <- prcomp(cars_log_multicolline)
summary(cars_log_multicolline_pca)$importance[2, 1]
```

```
## [1] 0.93462
```

(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?

```
cars_log_multicolline_eigen$vectors
```

```
##            [,1]        [,2]       [,3]        [,4]
## [1,] -0.3944484  0.32615343  0.6895416  0.51241263
## [2,] -0.7221160  0.36134848 -0.1626248 -0.56703525
## [3,] -0.4322835 -0.87289692  0.2158783 -0.06766477
## [4,] -0.3689037 -0.03319916 -0.6719242  0.64134686
```

PC1 gets 72% information of log.displacement.

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

```
scores = cars_log_multicolline_pca$x
cars_log$PC1 <- scores[, "PC1"]
```

(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
## -0.53593 -0.06148  0.00149  0.06293  0.50928
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.395518   0.172873   8.073 8.84e-15 ***
## PC1               0.387073   0.014110  27.433  < 2e-16 ***
```

```
## log.acceleration. -0.189830   0.043246  -4.390 1.47e-05 ***
## model_year         0.029244   0.001871  15.628  < 2e-16 ***
## factor(origin)2   -0.010840   0.020738  -0.523    0.601
## factor(origin)3    0.002243   0.020517   0.109    0.913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1239 on 386 degrees of freedom
## Multiple R-squared:  0.8689, Adjusted R-squared:  0.8672
## F-statistic: 511.7 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?

```
cars_log_std <- data.frame(scale(cars_log))
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
## -1.57609 -0.18081  0.00438  0.18506  1.49772
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.004201   0.026912   0.156    0.876
## PC1                              0.832369   0.030342  27.433  < 2e-16 ***
## log.acceleration.               -0.101021   0.023014  -4.390 1.47e-05 ***
## model_year                       0.316814   0.020272  15.628  < 2e-16 ***
## factor(origin)0.525710525810929 -0.031878   0.060987  -0.523    0.601
## factor(origin)1.76714743013553   0.006595   0.060336   0.109    0.913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3644 on 386 degrees of freedom
## Multiple R-squared:  0.8689, Adjusted R-squared:  0.8672
## F-statistic: 511.7 on 5 and 386 DF,  p-value: < 2.2e-16
```

```
cor(cars_log_std)[9, ]
```

```
##          log.mpg.    log.cylinders. log.displacement.   log.horsepower.
##         0.8830302        -0.9542881        -0.9912446        -0.9205490
##       log.weight. log.acceleration.        model_year            origin
##        -0.9592987         0.5636235         0.3497284         0.6325888
##               PC1
##         1.0000000
```

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

```
# install.packages("readxl")
library(readxl)
sec_q <- data.frame(read_excel("security_questions.xlsx", sheet = "data", col_names = T))
```

A group of researchers is studying how customers who shopped on e-commerce websites over the winter holiday season perceived the security of their most recently used e-commerce site. Based on feedback from experts, the company has created eighteen questions (see 'questions' tab of excel file) regarding security considerations at e-commerce websites. Over 400 customers responded to these questions (see 'data' tab of Excel file). The researchers now wants to use the results of these eighteen questions to reveal if there are some underlying dimensions of people's perception of online security that effectively capture the variance of these eighteen questions. Let's analyze the principal components of the eighteen items.

a. How much variance did each extracted factor explain?

```
sec_q_pca <- prcomp(sec_q, scale. = TRUE)
summary(sec_q_pca)$importance[2, ]
```

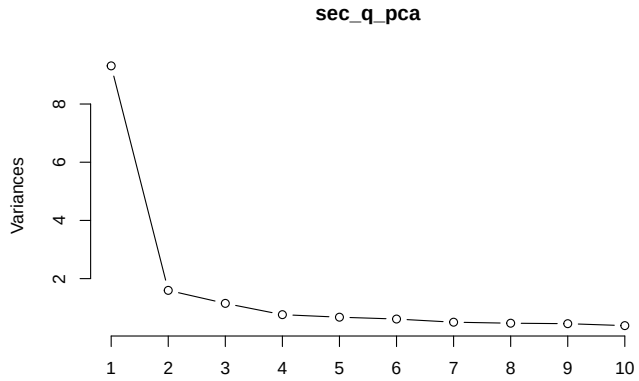
```
##     PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10
## 0.51728 0.08869 0.06386 0.04233 0.03751 0.03398 0.02794 0.02602 0.02511 0.02140
##    PC11    PC12    PC13    PC14    PC15    PC16    PC17    PC18
## 0.01972 0.01674 0.01624 0.01456 0.01303 0.01280 0.01160 0.01120
```

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

```
sec_q_eigen <- eigen(cor(sec_q))
sec_q_eigen$values
```

```
##  [1] 9.3109533 1.5963320 1.1495582 0.7619759 0.6751412 0.6116636 0.5029855
##  [8] 0.4682788 0.4519711 0.3851964 0.3548816 0.3013071 0.2922773 0.2621437
## [15] 0.2345788 0.2304642 0.2087471 0.2015441
```

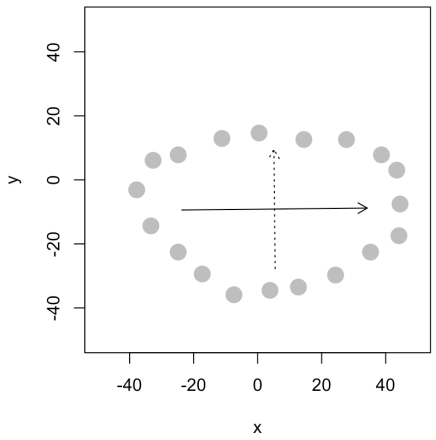
```
screeplot(sec_q_pca, type="lines")
```



According to the criterion of Eigenvalue ≥ 1 and the Scree Plot, we can see that retaining 3 PCs is the best scenario. Increasing the number of PCs beyond 3 does not lead to a significant increase in the explained variance.

Question 3) Let's simulate how principal components behave interactively: run the interactive_pca() function from the compstatslib package we have used earlier:

a. Create an oval shaped scatter plot of points that stretches in two directions – you should find that the principal component vectors point in the major and minor directions of variance (dispersion). Show this visualization.



b. Can you create a scatterplot whose principal component vectors do NOT seem to match the major directions of variance? Show this visualization.