EDA - Exam 1

Aaron Niskin

October 04, 2016

This is a 60 minute exam. You are to write an R Markdown document that provides scripts for carrying out the following tasks. It is the work, not the answers, which will be evaluated. You may use the electronic resources at your disposal, but please do your own work. Do as much as you can. Mail your pdf to mcdonald@ncf.edu at the end of the 60 minute period. Navigate to the following page:

https://archive.ics.uci.edu/ml/datasets/Automobile

Read the annotation.

Importing data

1. **Read the data into R as a csv file.**

```r
download.file(data_url, "data.csv")
```

```r
dats <- read.csv("data.csv", header = FALSE)
str(dats)
```

```
## 'data.frame':    205 obs. of  26 variables:
##  $ V1 : int  3 3 1 2 2 2 1 1 1 0 ...
##  $ V2 : Factor w/ 52 levels "?","101","102",..: 1 1 1 29 29 1 27 1 27 1 ...
##  $ V3 : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ V4 : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ V5 : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1 1 1 2 2 ...
##  $ V6 : Factor w/ 3 levels "?","four","two": 3 3 3 2 2 3 2 2 2 3 ...
##  $ V7 : Factor w/ 5 levels "convertible",..: 1 1 3 4 4 4 4 5 4 3 ...
##  $ V8 : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2 1 2 2 2 2 1 ...
##  $ V9 : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
##  $ V10: num  88.6 88.6 94.5 99.8 99.4 ...
##  $ V11: num  169 169 171 177 177 ...
##  $ V12: num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
##  $ V13: num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
##  $ V14: int  2548 2548 2823 2337 2824 2507 2844 2954 3086 3053 ...
##  $ V15: Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4 4 4 4 4 4 4 ...
##  $ V16: Factor w/ 7 levels "eight","five",..: 3 3 4 3 2 2 2 2 2 2 ...
##  $ V17: int  130 130 152 109 136 136 136 136 131 131 ...
##  $ V18: Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ V19: Factor w/ 39 levels "?","2.54","2.68",..: 25 25 3 15 15 15 15 15 12 12 ...
##  $ V20: Factor w/ 37 levels "?","2.07","2.19",..: 6 6 29 26 26 26 26 26 26 26 ...
##  $ V21: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ V22: Factor w/ 60 levels "?","100","101",..: 7 7 22 4 10 6 6 6 17 25 ...
##  $ V23: Factor w/ 24 levels "?","4150","4200",..: 12 12 12 18 18 18 18 18 18 18 ...
##  $ V24: int  21 21 19 24 18 19 19 19 17 16 ...
##  $ V25: int  27 27 26 30 22 25 25 25 20 22 ...
##  $ V26: Factor w/ 187 levels "?","10198","10245",..: 33 52 52 38 63 43 65 73 83 1 ...
```

Which tells us that they are using “?” to denote NA values. So let’s use bash to remove those (it’s easier for me right now).

```
cat data.csv | sed 's/\?/NA/g' > data_NA.csv
```

```
dats <- read.csv("data_NA.csv", header = FALSE)
str(dats)
```

```
## 'data.frame':    205 obs. of  26 variables:
##  $ V1 : int  3 3 1 2 2 2 1 1 1 0 ...
##  $ V2 : int  NA NA NA 164 164 NA 158 NA 158 NA ...
##  $ V3 : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ V4 : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ V5 : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1 1 1 2 2 ...
##  $ V6 : Factor w/ 2 levels "four","two": 2 2 2 1 1 2 1 1 1 2 ...
##  $ V7 : Factor w/ 5 levels "convertible",..: 1 1 3 4 4 4 4 5 4 3 ...
##  $ V8 : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2 1 2 2 2 2 1 ...
##  $ V9 : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
##  $ V10: num  88.6 88.6 94.5 99.8 99.4 ...
##  $ V11: num  169 169 171 177 177 ...
##  $ V12: num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
##  $ V13: num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
##  $ V14: int  2548 2548 2823 2337 2824 2507 2844 2954 3086 3053 ...
##  $ V15: Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4 4 4 4 4 4 4 ...
##  $ V16: Factor w/ 7 levels "eight","five",..: 3 3 4 3 2 2 2 2 2 2 ...
##  $ V17: int  130 130 152 109 136 136 136 136 131 131 ...
##  $ V18: Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ V19: num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
##  $ V20: num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
##  $ V21: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ V22: int  111 111 154 102 115 110 110 110 140 160 ...
##  $ V23: int  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
##  $ V24: int  21 21 19 24 18 19 19 19 17 16 ...
##  $ V25: int  27 27 26 30 22 25 25 25 20 22 ...
##  $ V26: int  13495 16500 16500 13950 17450 15250 17710 18920 23875 NA ...
```

Now it seems like everything is working quite well.

2. **Use the following vector to name the features of the csv file:**

```
features <- c("symboling", "normalized-losses","make", "fuel-type","aspiration", "num-of-doors","bo
```

```
names(dats) <- features
str(dats)
```

```
## 'data.frame':    205 obs. of  26 variables:
##  $ symboling        : int  3 3 1 2 2 2 1 1 1 0 ...
##  $ normalized-losses: int  NA NA NA 164 164 NA 158 NA 158 NA ...
##  $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ fuel-type        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ aspiration       : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1 1 1 2 2 ...
##  $ num-of-doors     : Factor w/ 2 levels "four","two": 2 2 2 1 1 2 1 1 1 2 ...
##  $ body-style       : Factor w/ 5 levels "convertible",..: 1 1 3 4 4 4 4 5 4 3 ...
## $ drive-wheels     : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2 1 2 2 2 2 1 ...
## $ engine-location  : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
## $ wheel-base       : num  88.6 88.6 94.5 99.8 99.4 ...
## $ length           : num  169 169 171 177 177 ...
## $ width            : num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
## $ height           : num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
## $ curb-weight      : int  2548 2548 2823 2337 2824 2507 2844 2954 3086 3053 ...
## $ engine-type      : Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4 4 4 4 4 4 4 ...
## $ num-of-cylinders : Factor w/ 7 levels "eight","five",..: 3 3 4 3 2 2 2 2 2 2 ...
## $ engine-size      : int  130 130 152 109 136 136 136 136 131 131 ...
## $ fuel-system      : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 6 6 ...
## $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
## $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
## $ compression-ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
## $ horsepower       : int  111 111 154 102 115 110 110 110 140 160 ...
## $ peak-rpm         : int  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
## $ city-mpg         : int  21 21 19 24 18 19 19 19 17 16 ...
## $ highway-mpg      : int  27 27 26 30 22 25 25 25 20 22 ...
## $ price            : int  13495 16500 16500 13950 17450 15250 17710 18920 23875 NA ...
```

Initial vetting

3. **How many complete cases are there?**

```r
sum(complete.cases(dats))
```

```
## [1] 159
```

4. **Subset the data on complete cases.**

```r
dats_complete <- dats[complete.cases(dats),]
```

5. **Working with complete cases, what is the range of values for the feature "horsepower"? What is the mean value?**

```r
summary(dats_complete$horsepower)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   48.00   69.00   88.00   95.84  114.00  200.00
```

So, 48 to 200 with a mean of 95.

Subsetting and binning

6. **Subset the data to include "make"" and ten of the last eleven variables, omitting "fuel-system". Assign the outcome to a file called "car_data_reduced". Convert the last ten variables of "car_data_reduced" to numeric.**

```r
which(names(dats) == "make")
```

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

```r
which(names(dats) == "fuel-system")
```

```
## [1] 18
```

```r
length(names(dats))
```

```
## [1] 26
```

```r
car_data_reduced <- dats[,c(3,16:17,19:26)]
str(car_data_reduced)
```

```
## 'data.frame':    205 obs. of  11 variables:
##  $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ num-of-cylinders : Factor w/ 7 levels "eight","five",..: 3 3 4 3 2 2 2 2 2 2 ...
##  $ engine-size      : int  130 130 152 109 136 136 136 136 131 131 ...
##  $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
##  $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
##  $ compression-ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ horsepower       : int  111 111 154 102 115 110 110 110 140 160 ...
##  $ peak-rpm         : int  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
##  $ city-mpg         : int  21 21 19 24 18 19 19 19 17 16 ...
##  $ highway-mpg      : int  27 27 26 30 22 25 25 25 20 22 ...
##  $ price            : int  13495 16500 16500 13950 17450 15250 17710 18920 23875 NA ...
```

So it seems like a straight conversion should be fine. If there were still factors in the last ten rows, we'd have to use `as.character` first.

```r
car_data_reduced[,2:11] <- sapply(car_data_reduced[,2:11], as.numeric)
str(car_data_reduced)
```

```
## 'data.frame':    205 obs. of  11 variables:
##  $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ num-of-cylinders : num  3 3 4 3 2 2 2 2 2 2 ...
##  $ engine-size      : num  130 130 152 109 136 136 136 136 131 131 ...
##  $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
##  $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
##  $ compression-ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ horsepower       : num  111 111 154 102 115 110 110 110 140 160 ...
##  $ peak-rpm         : num  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
##  $ city-mpg         : num  21 21 19 24 18 19 19 19 17 16 ...
##  $ highway-mpg      : num  27 27 26 30 22 25 25 25 20 22 ...
##  $ price            : num  13495 16500 16500 13950 17450 ...
```

7. **Bin the "horsepower" feature as 5 intervals of equal length, with the right-hand-endpoint of the last interval determined by maximal value of the feature and the left-hand-endpoint determined by the minimal value of the feature. Add this "binned information" as a feature to car_data_reduced.**

```r
seqMin <- min(car_data_reduced$horsepower, na.rm = TRUE)
seqMax <- max(car_data_reduced$horsepower, na.rm = TRUE)
step <- (seqMax - seqMin) / 5
hp_binned <- cut(car_data_reduced$horsepower,
                 breaks = seq(seqMin, seqMax, step),
                 include.lowest = TRUE,
                 labels = 1:5)
car_data_reduced$hp_binned <- as.integer(as.character(hp_binned))
str(car_data_reduced)
```

```
## 'data.frame':    205 obs. of  12 variables:
##  $ make             : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ num-of-cylinders : num  3 3 4 3 2 2 2 2 2 2 ...
##  $ engine-size      : num  130 130 152 109 136 136 136 136 131 131 ...
##  $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
##  $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
##  $ compression-ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ horsepower       : num  111 111 154 102 115 110 110 110 140 160 ...
##  $ peak-rpm         : num  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
##  $ city-mpg         : num  21 21 19 24 18 19 19 19 17 16 ...
##  $ highway-mpg      : num  27 27 26 30 22 25 25 25 20 22 ...
##  $ price            : num  13495 16500 16500 13950 17450 ...
##  $ hp_binned        : int  2 2 3 2 2 2 2 2 2 3 ...
```

8. **What is the make of the car belonging to the third interval and having maximal price?**

```r
max(car_data_reduced[car_data_reduced$hp_binned == 3,]$price, na.rm = TRUE)
```

```
## [1] 45400
```

Aggregation

9. **Compute the median values for all variables in car_data_reduced except "make" and the binned information, aggregating on binned horsepower and the number of cylinders.**

```r
which(names(car_data_reduced) == "make")
```

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

```r
length(names(car_data_reduced))
```

```
## [1] 12
```

```r
agg <- aggregate(car_data_reduced[,2:12],
                 by=list(num_cyl = car_data_reduced$`num-of-cylinders`,
                         hp_binned = car_data_reduced$hp_binned),
                 FUN = median,
                 na.rm=TRUE)
agg
```

```
##    num_cyl hp_binned num-of-cylinders engine-size bore stroke
## 1        3         1                3          98 3.15  3.290
## 2        5         1                5          61 2.91  3.030
## 3        2         2                2         136 3.19  3.400
## 4        3         2                3         120 3.50  3.190
## 5        4         2                4         164 3.31  3.190
## 6        7         2                7          70   NA     NA
## 7        1         3                1         269 3.63  3.225
## 8        2         3                2         131 3.13  3.400
## 9        3         3                3         141 3.59  3.150
## 10       4         3                4         181 3.43  3.350
## 11       4         4                4         194 3.74  2.900
## 12       1         5                1         203 3.94  3.110
## 13       6         5                6         326 3.54  2.760
##    compression-ratio horsepower peak-rpm city-mpg highway-mpg   price
## 1               9.00       70.0     5000       30          34  7799.0
## 2               9.50       48.0     5100       47          53  5151.0
## 3               8.50      119.0     5500       19          25 21397.5
## 4               9.10      111.0     5400       23          29 11900.0
## 5               9.00      121.0     4250       21          27 21485.0
## 6               9.40      101.0     6000       17          23 12745.0
## 7               8.15      169.5     4625       15          17 38008.0
## 8               7.00      160.0     5500       16          22      NA
## 9               7.50      160.0     5000       19          24 16503.0
## 10              9.00      160.0     5200       19          24 16558.0
## 11              9.50      207.0     5900       17          25 33278.0
## 12             10.00      288.0     5750       17          28      NA
## 13             11.50      262.0     5000       13          17 36000.0
##    hp_binned
## 1          1
## 2          1
## 3          2
## 4          2
## 5          2
## 6          2
## 7          3
## 8          3
## 9          3
## 10         3
## 11         4
## 12         5
## 13         5
```

10. **What pair of the last 10 variables (omitting the binned information) is maximally correlated? Construct a scatterplot for these variables.**

```r
cor(car_data_reduced[complete.cases(car_data_reduced[,2:11]),2:11])
```

```
##                   num-of-cylinders engine-size         bore      stroke
## num-of-cylinders        1.00000000  0.09580375 -0.014151453 -0.07620869
## engine-size             0.09580375  1.00000000  0.583091333  0.21198929
## bore                   -0.01415145  0.58309133  1.000000000 -0.06679316
## stroke                 -0.07620869  0.21198929 -0.066793158  1.00000000
## compression-ratio      -0.07573865  0.02461689  0.003056787  0.19988189
## horsepower              0.26140747  0.84269102  0.568527205  0.10003999
## peak-rpm                0.10008369 -0.21900779 -0.277661680 -0.06829951
## city-mpg               -0.03708887 -0.71062448 -0.591950375 -0.02764104
## highway-mpg             0.01696944 -0.73213800 -0.600039574 -0.03645288
## price                   0.01063105  0.88894226  0.546872917  0.09374644
##                   compression-ratio horsepower    peak-rpm    city-mpg
## num-of-cylinders       -0.075738653  0.2614075  0.10008369 -0.03708887
## engine-size             0.024616889  0.8426910 -0.21900779 -0.71062448
## bore                    0.003056787  0.5685272 -0.27766168 -0.59195038
## stroke                  0.199881893  0.1000400 -0.06829951 -0.02764104
## compression-ratio       1.000000000 -0.2144010 -0.44458194  0.33141295
## horsepower             -0.214401004  1.0000000  0.10565391 -0.83411654
## peak-rpm               -0.444581936  0.1056539  1.00000000 -0.06949336
## city-mpg                0.331412952 -0.8341165 -0.06949336  1.00000000
## highway-mpg             0.267940954 -0.8129169 -0.01695001  0.97234992
## price                   0.069500205  0.8110268 -0.10433340 -0.70268485
##                   highway-mpg       price
## num-of-cylinders   0.01696944  0.01063105
## engine-size       -0.73213800  0.88894226
## bore              -0.60003957  0.54687292
## stroke            -0.03645288  0.09374644
## compression-ratio  0.26794095  0.06950020
## horsepower        -0.81291687  0.81102684
## peak-rpm          -0.01695001 -0.10433340
## city-mpg           0.97234992 -0.70268485
## highway-mpg        1.00000000 -0.71558976
## price             -0.71558976  1.00000000
```

So, “stroke” and “highway-mpg” have a 0.97 correlation.

```r
#### editted after turnin (this mistake bothered me so I found a way to automate it)
findMaxCor <- function(df) {
  c <- cor(df[complete.cases(df),])
  diag(c) <- 0
  tmp2 <- which(c == max(c)) %% length(names(df))
  tmp2[tmp2 == 0] <- length(names(df))
  return(c(names(df)[tmp2], max(c)))
}
findMaxCor(car_data_reduced[,2:11])
```

```
## [1] "highway-mpg"       "city-mpg"          "0.972349916539918"
```

Data reduction

11. **Consider a data frame consisting of the last 11 features of the original data. Perform a principal component analysis using the tool of your choice. How many components are required to account for 90% of the data?**

```r
dim(dats_complete)
```

```
## [1] 159  26
```

```r
str(dats_complete)
```

```
## 'data.frame':    159 obs. of  26 variables:
##  $ symboling        : int  2 2 1 1 2 0 0 0 2 1 ...
##  $ normalized-losses: int  164 164 158 158 192 192 188 188 121 98 ...
##  $ make             : Factor w/ 22 levels "alfa-romero",..: 2 2 2 2 3 3 3 3 4 4 ...
##  $ fuel-type        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ aspiration       : Factor w/ 2 levels "std","turbo": 1 1 1 2 1 1 1 1 1 1 ...
##  $ num-of-doors     : Factor w/ 2 levels "four","two": 1 1 1 1 2 1 2 1 2 2 ...
##  $ body-style       : Factor w/ 5 levels "convertible",..: 4 4 4 4 4 4 4 4 3 3 ...
##  $ drive-wheels     : Factor w/ 3 levels "4wd","fwd","rwd": 2 1 2 2 3 3 3 3 2 2 ...
##  $ engine-location  : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
##  $ wheel-base       : num  99.8 99.4 105.8 105.8 101.2 ...
##  $ length           : num  177 177 193 193 177 ...
##  $ width            : num  66.2 66.4 71.4 71.4 64.8 64.8 64.8 64.8 60.3 63.6 ...
##  $ height           : num  54.3 54.3 55.7 55.9 54.3 54.3 54.3 54.3 53.2 52 ...
##  $ curb-weight      : int  2337 2824 2844 3086 2395 2395 2710 2765 1488 1874 ...
##  $ engine-type      : Factor w/ 7 levels "dohc","dohcv",..: 4 4 4 4 4 4 4 4 3 4 ...
##  $ num-of-cylinders : Factor w/ 7 levels "eight","five",..: 3 2 2 2 3 3 4 4 5 3 ...
##  $ engine-size      : int  109 136 136 131 108 108 164 164 61 90 ...
##  $ fuel-system      : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 2 2 ...
##  $ bore             : num  3.19 3.19 3.19 3.13 3.5 3.5 3.31 3.31 2.91 3.03 ...
##  $ stroke           : num  3.4 3.4 3.4 3.4 2.8 2.8 3.19 3.19 3.03 3.11 ...
##  $ compression-ratio: num  10 8 8.5 8.3 8.8 8.8 9 9 9.5 9.6 ...
##  $ horsepower       : int  102 115 110 140 101 101 121 121 48 70 ...
##  $ peak-rpm         : int  5500 5500 5500 5500 5800 5800 4250 4250 5100 5400 ...
##  $ city-mpg         : int  24 18 19 17 23 23 21 21 47 38 ...
##  $ highway-mpg      : int  30 22 25 20 29 29 28 28 53 43 ...
##  $ price            : int  13950 17450 17710 23875 16430 16925 20970 21105 5151 6295 ...
```

```r
summary(prcomp(dats_complete[19:26], center = TRUE, scale. = TRUE))
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.9714 1.2883 1.0628 0.74635 0.61234 0.47220
## Proportion of Variance 0.4858 0.2075 0.1412 0.06963 0.04687 0.02787
## Cumulative Proportion  0.4858 0.6933 0.8345 0.90411 0.95098 0.97885
##                            PC7     PC8
## Standard deviation     0.37921 0.15939
## Proportion of Variance 0.01797 0.00318
## Cumulative Proportion  0.99682 1.00000
```

12. **(Extra credit)Does the answer change if the variables are standardized prior to performing PCA?**