EDA - Assignment 9

Aaron Niskin

September 29, 2016

Navigate to the website

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

Read the annotation provided. Download the associated hepatitis.data and the hepatitis.names files. Prepare an R Markdown file which documents the steps you take in carrying out the following

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

```
data_url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/hepatitis/hepatitis.data"
data_file <- "data.csv"
download.file(data_url, data_file)
names_url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/hepatitis/hepatitis.names"
names_file <- "names.csv"
download.file(names_url, names_file)
```

I do that at first so that every time I compile this PDF it won't try to download the files again and again. So now we want to read these files into memory. But before we do that, we're going to want to get rid of those pesky "?" NA values. So let's replace all of those using our favorite bash stream editing program, "sed"!

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

Now when we import the data into R, it should identify the types far more easily.

```
hep_data <- read.csv("data_na.csv", header=FALSE)
str(hep_data)
```

```
## '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 ...
```

2. **Name the features as described in the names file**

   So, first we check out that "names.txt" file. Then we notice section 7 seems to be describing the names of each of the columns. But then section 9 says,

   9. Class Distribution:

      `DIE: 32`

      `LIVE: 123`

      So if we were unsure of whether our names are listed in ascending or descending order, we can check by confirming that there are indeed 32 deaths and 123 live records. Unfortunately, if you check out column 1, the data is binary {1,2}, and not labeled as `live` or `die`. But we can count them. If there is a 32/123 split, we can be ralatively assured that the data is presented in ascending order (especially since this is the most logical way to present it anyway).

```r
sum(hep_data$V1 == 1)
```

```
## [1] 54
```

```r
sum(hep_data$V1 == 2)
```

```
## [1] 32
```

This also gave us the beneifit of identifying that 1 corresponds with "die" and 2 with "live". Now since there are 20 different names here, and I'm pretty lazy, let's see if we can get this done programatically. First we're going to want to cat lines 30 - 50 into sed, then use sed to grab just the names, then feed that into a new file. And of course it turns out, as most things of this nature do, it would have been a little easier if I had just done it directly in a text editor. But this way was far more fun.

```bash
head -n -41 names.txt | tail -n -21 | sed -r '/^\ *--.*$/d;s/^[0-9 \.]+([a-zA-Z ]+)\:.*$/\1/g;' >
```

Now we're just about ready to read that file in!

```r
names(hep_data) <- lapply(strsplit(tolower(readLines("namesJust.txt")), "\n"), as.character)
```

3. **Write the result to memory as a csv file**

```r
write.csv(hep_data, "hep_data.csv", row.names = FALSE)
```

Continuing your document, addressing the following questions

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

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

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

5. **Subsetting the data on Age, Sex, Bilirubin, ALK, SGOT and Albumin, compute the number of missing values for the Bilirubin feature. Convert the last four features to numeric values. How many complete cases are there for the subsetted frame?**

```r
str(hep_data)
```

```
## 'data.frame':    205 obs. of  26 variables:
##  $ class          : int  3 3 1 2 2 2 1 1 1 0 ...
##  $ age            : int  NA NA NA 164 164 NA 158 NA 158 NA ...
##  $ sex            : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ steroid        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ antivirals     : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1 1 1 2 2 ...
##  $ fatigue        : Factor w/ 2 levels "four","two": 2 2 2 1 1 2 1 1 1 2 ...
##  $ malaise        : Factor w/ 5 levels "convertible",..: 1 1 3 4 4 4 4 5 4 3 ...
##  $ anorexia       : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2 1 2 2 2 2 1 ...
##  $ liver big      : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
##  $ liver firm     : num  88.6 88.6 94.5 99.8 99.4 ...
##  $ spleen palpable: num  169 169 171 177 177 ...
##  $ spiders        : num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
##  $ ascites        : num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
##  $ varices        : int  2548 2548 2823 2337 2824 2507 2844 2954 3086 3053 ...
##  $ bilirubin      : Factor w/ 7 levels "dohc","dohcv",..: 1 1 6 4 4 4 4 4 4 4 ...
##  $ alk phosphate  : Factor w/ 7 levels "eight","five",..: 3 3 4 3 2 2 2 2 2 2 ...
##  $ sgot           : int  130 130 152 109 136 136 136 136 131 131 ...
##  $ albumin        : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ protime        : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
##  $ histology      : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
##  $ NA             : num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
##  $ NA             : int  111 111 154 102 115 110 110 110 140 160 ...
##  $ NA             : int  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
##  $ NA             : int  21 21 19 24 18 19 19 19 17 16 ...
##  $ NA             : int  27 27 26 30 22 25 25 25 20 22 ...
##  $ NA             : int  13495 16500 16500 13950 17450 15250 17710 18920 23875 NA ...
```

```r
tmp <- as.data.frame(cbind(hep_data[,c("age", "sex")], sapply(
      hep_data[, c("bilirubin", "alk phosphate", "sgot", "albumin")],
        as.numeric)))
str(tmp)
```

```
## 'data.frame':    205 obs. of  6 variables:
##  $ age          : int  NA NA NA 164 164 NA 158 NA 158 NA ...
##  $ sex          : Factor w/ 22 levels "alfa-romero",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ bilirubin    : num  1 1 6 4 4 4 4 4 4 4 ...
##  $ alk phosphate: num  3 3 4 3 2 2 2 2 2 2 ...
##  $ sgot         : num  130 130 152 109 136 136 136 136 131 131 ...
##  $ albumin      : num  6 6 6 6 6 6 6 6 6 6 ...
```

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

```
## [1] 164
```

```r
sum(complete.cases(tmp[,"bilirubin"]))
```

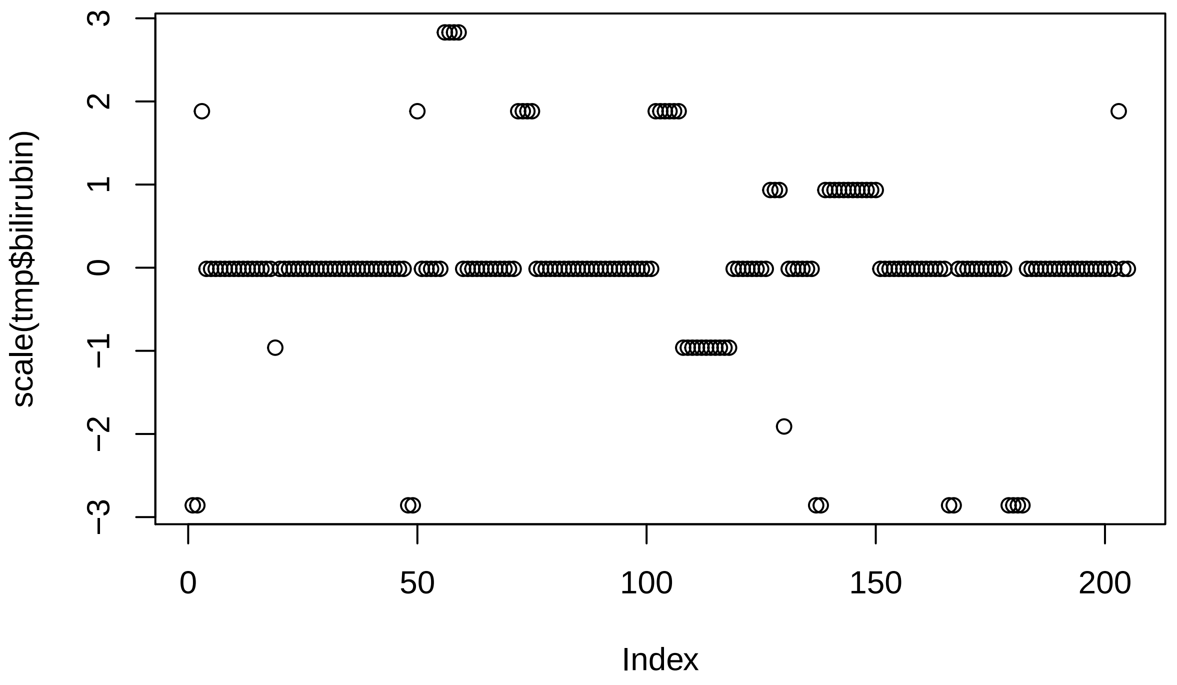
```
## [1] 205
```

6. **Are there any outliers in the Bilirubin and Albumin entries?**

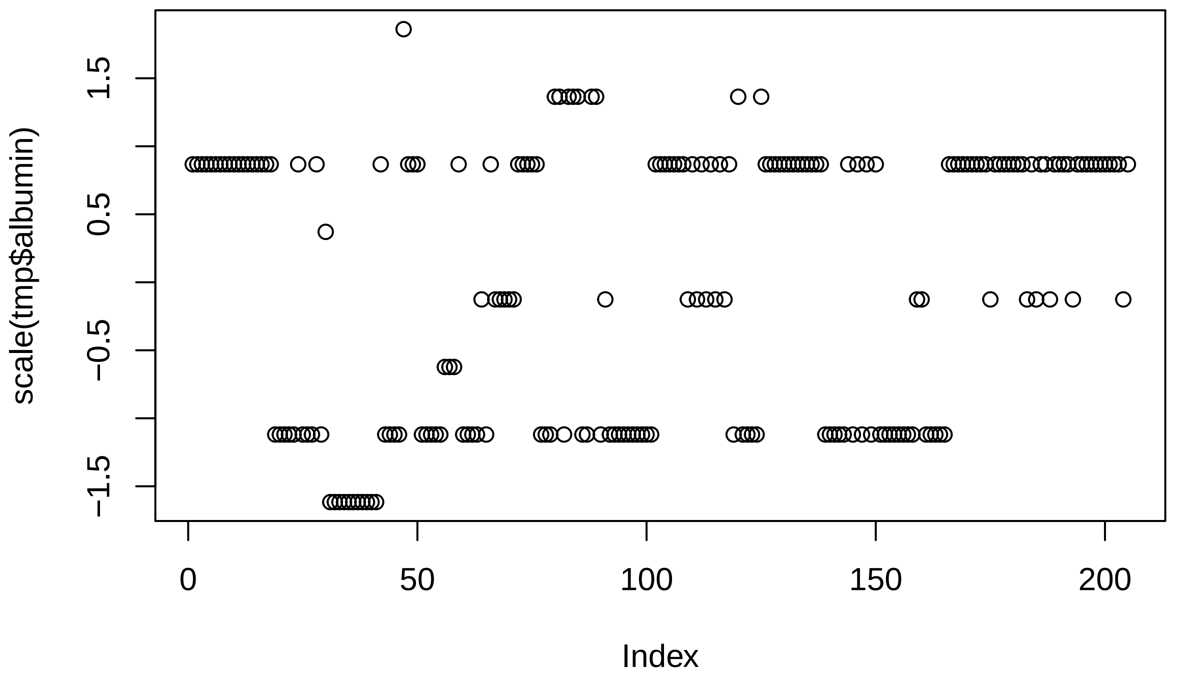
```r
quantile(tmp$bilirubin, na.rm = TRUE)
```

```
##   0%  25%  50%  75% 100% 
##    1    4    4    4    7
```

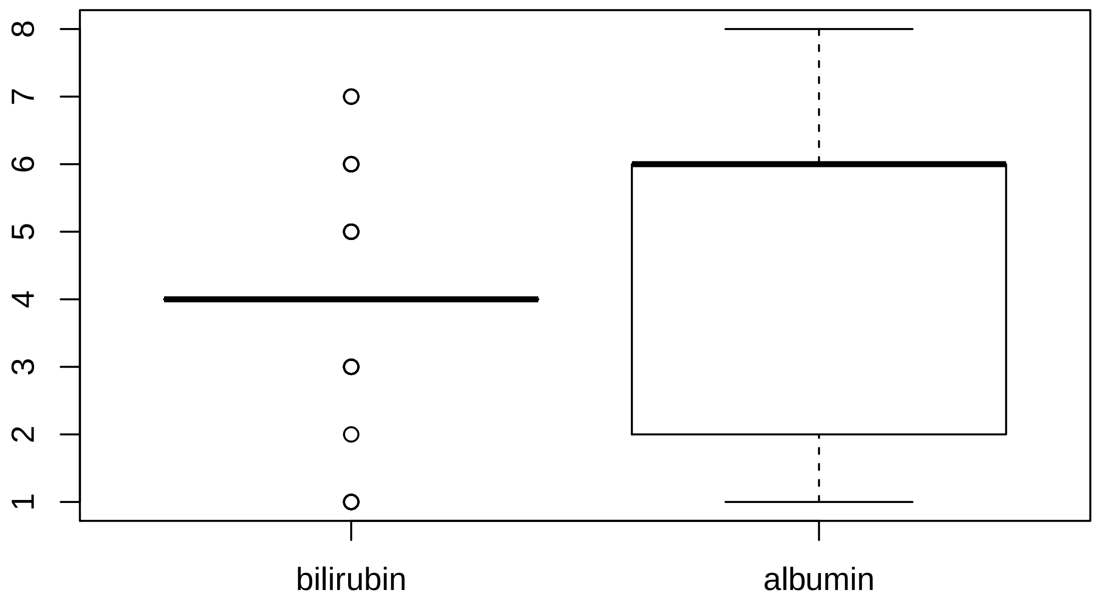
```r
plot(scale(tmp$bilirubin))
```



```r
plot(scale(tmp$albumin))
```



```r
boxplot(tmp[,c("bilirubin", "albumin")])
```



There do seem to be outliers, and there is a note about bilirubin specifically, but that seems to be mainly concerning the data's continuity. After doing a bit of research, it seems very likely that the bilirubin data corresponds to something called "direct bilirubin" measured in units of mmol. I owuld need more information to actually assess with any validity of this data with any measure of accuracy, but according to MedicalHealthTests.com, if the previous assumption/semi-conclusion is true, then 5.1 is on the high side and our two data points around 8 are way too high.

7. **Bin the age variables in units of decades**

   So just to make this easier, let's do this…

```
max(tmp$age)
```

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

```
min(tmp$age)
```

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

So we can bin our data from 0 to 80.

```
age_groups <- cut(tmp$age, breaks = seq(0, 80, 10))
tmp[,"age"] <- age_groups
```

8. **Aggregate the data to obtain mean readings for the last 4 variables as a function of sex and age, with age as a binned factor.**

   By last four variables, I'm going to assume that you mean, "BILIRUBIN", "ALK PHOSPHATE", "SGOT", "ALBUMIN".

```
agg <- aggregate(tmp[,3:6], by=list(age=tmp$age, sex=tmp$sex), FUN=mean, na.rm=TRUE)
agg
```

```
##       age      sex bilirubin alk phosphate     sgot albumin
## 1 (70,80]    honda         4             3  92.0000     1.0
## 2 (70,80] plymouth         4             3 122.0000     2.0
## 3 (60,70]   toyota         4             3 119.6000     5.6
## 4 (70,80]   toyota         4             3  92.0000     2.0
## 5 (70,80]    volvo         4             3 137.3333     6.0
```

9. **Sort the data on the Bilirubin columns (ascending)**

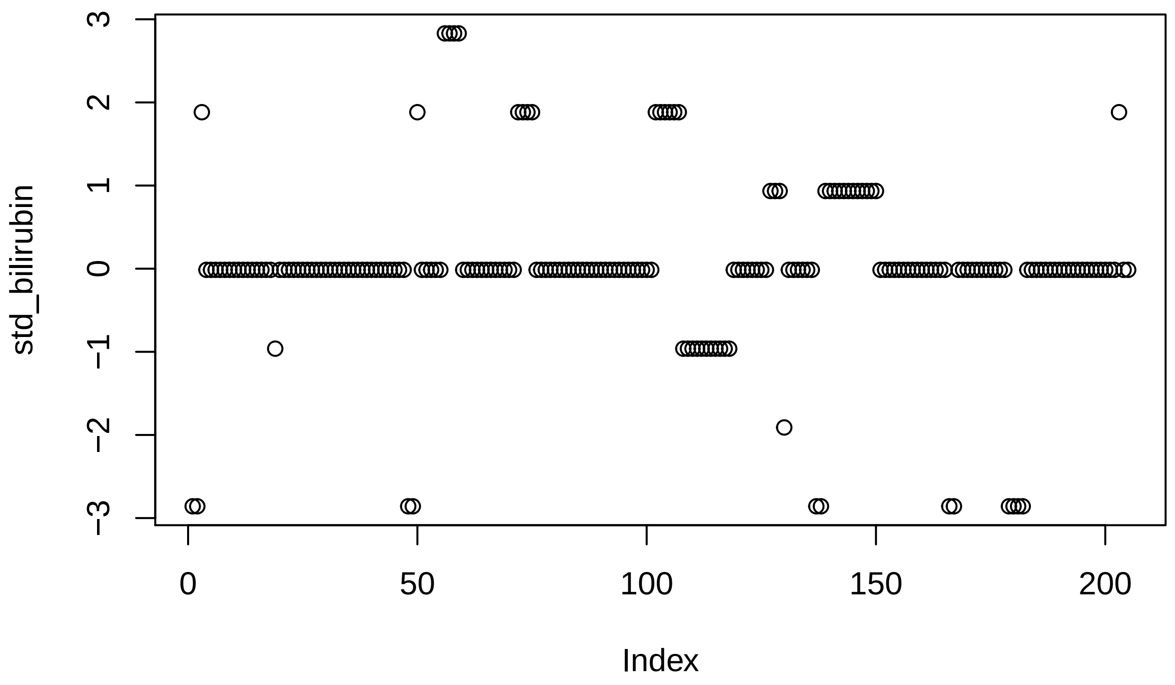
```
hep_sort <- hep_data[order(hep_data$bilirubin),]
str(hep_sort)
```

```
## 'data.frame':    205 obs. of  26 variables:
##  $ class          : int  3 3 0 0 3 2 1 1 3 3 ...
##  $ age            : int  NA NA 145 NA 150 104 168 168 197 197 ...
##  $ sex            : Factor w/ 22 levels "alfa-romero",..: 1 1 8 8 18 18 20 20 20 20 ...
##  $ steroid        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
##  $ antivirals     : Factor w/ 2 levels "std","turbo": 1 1 1 1 2 2 1 1 1 1 ...
##  $ fatigue        : Factor w/ 2 levels "four","two": 2 2 1 1 2 1 2 2 2 2 ...
##  $ malaise        : Factor w/ 5 levels "convertible",..: 1 1 4 4 3 4 4 3 3 3 ...
##  $ anorexia       : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 3 2 2 3 3 3 3 ...
##  $ liver big      : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
##  $ liver firm     : num  88.6 88.6 113 113 99.1 ...
##  $ spleen palpable: num  169 169 200 200 187 ...
##  $ spiders        : num  64.1 64.1 69.6 69.6 66.5 66.5 64 64 67.7 67.7 ...
##  $ ascites        : num  48.8 48.8 52.8 52.8 56.1 56.1 52.6 52.6 52 52 ...
##  $ varices        : int  2548 2548 4066 4066 2808 2847 2265 2300 2976 3016 ...
##  $ bilirubin      : Factor w/ 7 levels "dohc","dohcv",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ alk phosphate  : Factor w/ 7 levels "eight","five",..: 3 3 4 4 3 3 3 3 4 4 ...
##  $ sgot           : int  130 130 258 258 121 121 98 98 171 171 ...
##  $ albumin        : Factor w/ 8 levels "1bbl","2bbl",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ protime        : num  3.47 3.47 3.63 3.63 3.54 3.54 3.24 3.24 3.27 3.27 ...
##  $ histology      : num  2.68 2.68 4.17 4.17 3.07 3.07 3.08 3.08 3.35 3.35 ...
##  $ NA             : num  9 9 8.1 8.1 9 9 9.4 9.4 9.3 9.3 ...
##  $ NA             : int  111 111 176 176 160 160 112 112 161 161 ...
##  $ NA             : int  5000 5000 4750 4750 5500 5500 6600 6600 5200 5200 ...
##  $ NA             : int  21 21 15 15 19 19 26 26 20 19 ...
##  $ NA             : int  27 27 19 19 26 26 29 29 24 24 ...
##  $ NA             : int  13495 16500 32250 35550 18150 18620 9298 9538 16558 15998 ...
```

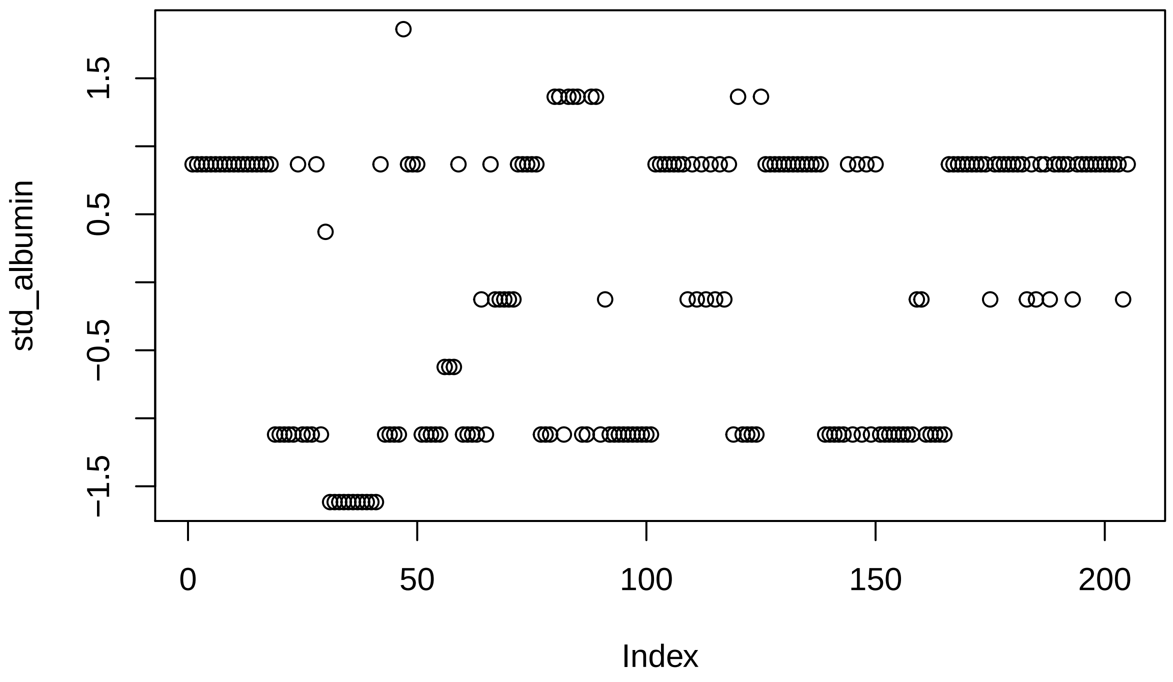
10. **Standardize Bilirubin and Albumin and plot the outcome as a scatterplot.**

    So this depends on what you mean by standardize. If you mean something like a z-score, then we can compute it thusly:

```
std_bilirubin <- scale(tmp$bilirubin)
std_albumin   <- scale(tmp$albumin)
plot(std_bilirubin)
```



```
plot(std_albumin)
```



11. **Consider the data frame consisting of the complete cases for the variables Bilirubin, ALK, SGOT and Albumin. What fraction of the variance does the first principal component account for?**

```
tmp2 <- tmp[,c("bilirubin", "alk phosphate", "sgot", "albumin")]
a <- prcomp(tmp2[complete.cases(tmp2),], center=TRUE, scale=TRUE)
a
```

```
## Standard deviations:
## [1] 1.2359724 1.1065479 0.8966278 0.6663200
##
## Rotation:
##                      PC1       PC2        PC3        PC4
## bilirubin     -0.1385276 0.6991067 -0.6535952  0.2547024
## alk phosphate -0.1585509 0.6830605  0.6762888 -0.2256623
## sgot           0.6898680 0.1660335 -0.2283903 -0.6665980
## albumin        0.6926427 0.1308095  0.2515645  0.6632121
```

```
b <- princomp(tmp2[complete.cases(tmp2),], scores = TRUE, cor = TRUE)
b
```

```
## Call:
## princomp(x = tmp2[complete.cases(tmp2), ], cor = TRUE, scores = TRUE)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4
## 1.2359724 1.1065479 0.8966278 0.6663200
##
##  4  variables and  205 observations.
```

```
c <- princomp(tmp2[complete.cases(tmp2),])
c
```

```
## Call:
## princomp(x = tmp2[complete.cases(tmp2), ])
##
## Standard deviations:
##     Comp.1     Comp.2     Comp.3     Comp.4
## 41.5539282  1.7312645  1.0762467  0.7353964
##
##  4  variables and  205 observations.
```

```
summary(a)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4
## Standard deviation     1.2360 1.1065 0.8966 0.6663
## Proportion of Variance 0.3819 0.3061 0.2010 0.1110
## Cumulative Proportion  0.3819 0.6880 0.8890 1.0000
```

```
summary(b)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4
## Standard deviation     1.2359724 1.1065479 0.8966278 0.6663200
## Proportion of Variance 0.3819069 0.3061121 0.2009854 0.1109956
## Cumulative Proportion  0.3819069 0.6880190 0.8890044 1.0000000
```

```
summary(c)
```

```
## Importance of components:
##                            Comp.1      Comp.2       Comp.3       Comp.4
## Standard deviation     41.5539282 1.731264487 1.0762466993 0.7353964418
## Proportion of Variance  0.9972876 0.001731104 0.0006689904 0.0003123484
## Cumulative Proportion   0.9972876 0.999018661 0.9996876516 1.0000000000
```

```
b["loadings"]
```

```
## $loadings
##
## Loadings:
##               Comp.1 Comp.2 Comp.3 Comp.4
## bilirubin     -0.139  0.699  0.654  0.255
## alk phosphate -0.159  0.683 -0.676 -0.226
## sgot           0.690  0.166  0.228 -0.667
## albumin        0.693  0.131 -0.252  0.663
##
##                Comp.1 Comp.2 Comp.3 Comp.4
## SS loadings      1.00   1.00   1.00   1.00
## Proportion Var   0.25   0.25   0.25   0.25
## Cumulative Var   0.25   0.50   0.75   1.00
```

12. **Subsetting the data on Age, Sex, Steroid and Antivirals columns and join the resulting data frame with the data frame of complete cases for Age, Sex, Bilirubin, ALK, SGOT and Albumin. What are the dimensions of the resulting frame?**

```r
tmpA <- hep_data[,c("sex", "age", "steroid", "antivirals")]
tmpB <- hep_data[,c("sex", "age", "bilirubin", "alk phosphate", "sgot", "albumin")]
tmpC <- merge.data.frame(tmpA[complete.cases(tmpA),],
                         tmpB[complete.cases(tmpB),])
dim(tmpC)
```

```
## [1] 574   8
```