Module 7: exercise solutions

BIOSTAT 213: Introduction to Computing in the R Software Environment

Wednesday, February 13, 2019

1. If you have not already, convert `start`, `end`, and `dob` to Date objects.

```
> pp<-read.csv('presidencies.csv')
> pp$start<-as.Date(pp$start,'%d/%m/%Y')
> pp$end<-as.Date(pp$end,'%d/%m/%Y')
> pp$dob<-as.Date(pp$dob,'%B %d, %Y')
```

2. Find the president who was youngest at the start of his term. How old was he? Find the president who was oldest at the start of his term. How old was he?

```
> pp$age_start<-as.numeric(pp$start-pp$dob)/365.25
> # using indexing:
> pp[pp$age_start==min(pp$age_start),c('name','age_start')]
                name age_start
26 Theodore Roosevelt  42.88022
> pp[pp$age_start==max(pp$age_start),c('name','age_start')]
            name age_start
40 Ronald Reagan  69.95483
> # using subset():
> subset(pp,age_start==min(age_start),select=c(name,age_start))
                name age_start
26 Theodore Roosevelt  42.88022
> subset(pp,age_start==max(age_start),select=c(name,age_start))
            name age_start
40 Ronald Reagan  69.95483
```

3. Find the president who was youngest at the end of his term. How old was he? Find the president who was oldest at the end of his term. How old was he?

```
> pp$age_end<-as.numeric(pp$end-pp$dob)/365.25
> # using indexing:
> pp[pp$age_end==min(pp$age_end),c('name','age_end')]
              name  age_end
35 John F. Kennedy 46.48323
> pp[pp$age_end==max(pp$age_end),c('name','age_end')]
            name  age_end
40 Ronald Reagan 77.95483
> # using subset():
> subset(pp,age_end==min(age_end),select=c(name,age_end))
              name  age_end
35 John F. Kennedy 46.48323
> subset(pp,age_end==max(age_end),select=c(name,age_end))
            name  age_end
40 Ronald Reagan 77.95483
```

4. Find the president with the longest term. How long was that term? Find the president with the shortest term. How long was that term?

```
> pp$duration<-as.numeric(pp$end-pp$start)
> # using indexing:
```

```
> pp[pp$duration==max(pp$duration),c('name','duration')]
                    name duration
32 Franklin D. Roosevelt     4422
> pp[pp$duration==min(pp$duration),c('name','duration')]
                    name duration
9 William Henry Harrison       31
> # using subset():
> subset(pp,duration==max(duration),select=c(name,duration))
                    name duration
32 Franklin D. Roosevelt     4422
> subset(pp,duration==min(duration),select=c(name,duration))
                    name duration
9 William Henry Harrison       31
```

5. Download and import the 2 data files `DEMO_I.XPT and DR1TOT_I`. Subset the demographic data set to individuals 50 years of age or younger. Use this reduced demographic data set for the subsequent problems.

```
> library(foreign)
> tt1<-read.xport('DEMO_I.XPT')
> tt2<-read.xport('DR1TOT_I.XPT')
> names(tt1)<-tolower(names(tt1))
> names(tt2)<-tolower(names(tt2))
> tt1<-subset(tt1,ridageyr<=50)
```

6. Check for duplicated ID (`SEQN`) in each data set.

```
> sum(duplicated(tt1$seqn))
[1] 0
> sum(duplicated(tt2$seqn))
[1] 0
```

7. Merge the dietary data into the demographic data by ID (`SEQN`), using `all=TRUE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all=TRUE` means X *OR* Y:

```
> length(unique(c(tt1$seqn,tt2$seqn)))
[1] 9847
```

```
> tt<-merge(tt1,tt2,by='seqn',all=TRUE)
> nrow(tt)
[1] 9847
```

8. Merge the dietary data into the demographic data by ID (`SEQN`), using `all=FALSE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all=FALSE` means X *AND* Y:

```
> sum(is.element(tt1$seqn,tt2$seqn))
[1] 6930
> sum(is.element(tt2$seqn,tt1$seqn))
[1] 6930
```

```
> tt<-merge(tt1,tt2,by='seqn',all=FALSE)
> nrow(tt)
[1] 6930
```

9. Merge the dietary data into the demographic data by ID (`SEQN`), using `all.x=TRUE, all.y=FALSE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match

the following, since `all.x=TRUE, all.y=FALSE` means simply X:

```
> length(unique(tt1$seqn))
[1] 7233
```

```
> tt<-merge(tt1,tt2,by='seqn',all.x=TRUE,all.y=FALSE)
> nrow(tt)
[1] 7233
```

10. Merge the dietary data into the demographic data by ID (`SEQN`), using `all.x=FALSE, all.y=TRUE`, and giving the new data frame a new arbitrary name. Check the number of rows. This should match the following, since `all.x=FALSE, all.y=TRUE` means simply Y:

```
> length(unique(tt2$seqn))
[1] 9544
```

```
> tt<-merge(tt1,tt2,by='seqn',all.x=FALSE,all.y=TRUE)
> nrow(tt)
[1] 9544
```