Bios 6301: Assignment 7

Yi Zuo

Due Thursday, 08 November, 1:00 PM

$5^{n=day}$ points taken off for each day late.

40 points total.

Submit a single knitr file (named `homework7.rmd`), along with a valid PDF output file. Inside the file, clearly indicate which parts of your responses go with which problems (you may use the original homework document as a template). Add your name as `author` to the file's metadata section. Raw R code/output or word processor files are not acceptable.

Failure to name file `homework7.rmd` or include author name may result in 5 points taken off.

Question 1

21 points

Use the following code to generate data for patients with repeated measures of A1C (a test for levels of blood glucose).

```
genData <- function(n) {
    if(exists(".Random.seed", envir = .GlobalEnv)) {
        save.seed <- get(".Random.seed", envir= .GlobalEnv)
        on.exit(assign(".Random.seed", save.seed, envir = .GlobalEnv))
    } else {
        on.exit(rm(".Random.seed", envir = .GlobalEnv))
    }
    set.seed(n)
    subj <- ceiling(n / 10)
    id <- sample(subj, n, replace=TRUE)
    times <- as.integer(difftime(as.POSIXct("2005-01-01"), as.POSIXct("2000-01-01"), units='secs'))
    dt <- as.POSIXct(sample(times, n), origin='2000-01-01')
    mu <- runif(subj, 4, 10)
    a1c <- unsplit(mapply(rnorm, tabulate(id), mu, SIMPLIFY=FALSE), id)
    data.frame(id, dt, a1c)
}
x <- genData(500)
```

Perform the following manipulations: (3 points each)

1. Order the data set by `id` and `dt`.

```
x1 <- x[order(x$id,x$dt),]
head(x1)
```

```
##     id                  dt      a1c
## 154  1 2000-03-14 07:44:28 5.452057
## 349  1 2000-05-04 10:29:31 4.609312
## 464  1 2000-11-13 03:32:47 4.219065
## 147  1 2001-09-05 21:00:24 2.234179
## 317  1 2001-11-20 01:57:31 2.748069
## 453  1 2002-08-28 04:44:21 4.904277
```

x1 is the data set ordered by id and dt.

2. For each id, determine if there is more than a one year gap in between observations. Add a new row at the one year mark, with the a1c value set to missing. A two year gap would require two new rows, and so forth.

```
f2 <- function(x1){

  x1_first_v_time <- x1[!duplicated(x1$id),'dt']

  x1_last_v_time <- as.POSIXct(as.vector(with(x1, tapply(dt, id, function(x) x[length(x)]))),origin='19

  x1_gap <- floor(as.numeric(difftime(x1_last_v_time,x1_first_v_time,units="days")/365))

  # create the appended data set
  x1_mark_id <- unlist(sapply(unique(x1$id),function(x){rep(x,x1_gap[x])}))
  x1_mark_index <- unlist(sapply(unique(x1$id),function(x){seq(x1_gap[x])}))

  x1_mark_dt<-numeric(length(x1_mark_id))
  for(i in seq_along(x1_mark_id) ){
    x1_mark_dt[i] <- x1_first_v_time[x1_mark_id[i]] + 365 * 86400 * x1_mark_index[i]
  }

  x1_mark_dt <- as.POSIXct(x1_mark_dt,origin='1970-01-01')

  x1_mark <- data.frame(id=x1_mark_id, dt=x1_mark_dt, a1c=NA)

  x2 <- rbind(x1,x1_mark)
  x2 <- x2[order(x2$id, x2$dt),]
  rownames(x2) <- NULL
  return(x2)
}

x2 <- f2(x1)
```

Data set x2 is the new data set with added rows.

3. Create a new column visit. For each id, add the visit number. This should be 1 to n where n is the number of observations for an individual. This should include the observations created with missing a1c values.

```
f3 <- function(x2){
  x2$visit <- unlist(sapply(tabulate(x2$id),function(x){seq(x)}))
  return(x2)
}

x3 <- f3(x2)
```

x3 is the new data set with new variable viist.

4. For each id, replace missing values with the mean a1c value for that individual.

```
f4 <- function(x3){
  x3_mean_a1c <- as.numeric(tapply(x3$a1c, x3$id, function(x) mean(x,na.rm=T)))

  for(i in seq_along(x3_mean_a1c)){
   index <- which(x3$id == i)
```

```
   x3[index[is.na(x3[index,'a1c'])], 'a1c'] <- x3_mean_a1c[i]
  }

  return(x3)
}

x4 <- f4(x3)
```

x4 is the new data set without missing a1c.

5. Print mean `a1c` for each `id`.

```
tapply(x4$a1c, x4$id, mean)
```

```
##        1        2        3        4        5        6        7        8
## 4.063372 7.544643 6.757640 3.892127 9.512311 7.555965 9.161686 7.189064
##        9       10       11       12       13       14       15       16
## 9.283873 7.975217 6.917562 7.034021 9.145282 6.623756 8.012406 4.222158
##       17       18       19       20       21       22       23       24
## 3.996034 9.164873 5.507210 3.726675 8.140939 5.637501 7.366889 7.439316
##       25       26       27       28       29       30       31       32
## 6.877135 6.556759 4.926457 7.433917 4.508086 6.045577 7.116586 6.568791
##       33       34       35       36       37       38       39       40
## 6.494069 6.768615 8.476700 9.604410 9.606253 5.355979 6.917013 9.530136
##       41       42       43       44       45       46       47       48
## 9.802424 3.891770 6.095849 9.091670 6.737204 9.621763 9.231489 6.404600
##       49       50
## 6.096076 8.962319
```

6. Print total number of visits for each `id`.

```
tapply(x4$visit, x4$id, max)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## 15 24 16 14 16 12 12 15 15 15 12 11  9 15  9 11 16 14 12 11 13 10  9 19 15
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
## 17 14 17 12  9 13  6 10 14 14 10 19 18  9  9 21 17 13 13 17 10 16 13 15 13
```

7. Print the observations for `id = 15`.

```
x4[x4$id == 15,]
```

```
##     id                  dt      a1c visit
## 202 15 2000-04-30 00:34:50 7.527105     1
## 203 15 2001-01-17 21:11:02 5.898371     2
## 204 15 2001-04-25 06:23:05 8.566593     3
## 205 15 2001-04-30 00:34:50 8.012406     4
## 206 15 2002-04-30 00:34:50 8.012406     5
## 207 15 2003-04-30 00:34:50 8.012406     6
## 208 15 2003-06-06 14:06:00 9.133769     7
## 209 15 2004-04-29 00:34:50 8.012406     8
## 210 15 2004-08-20 17:47:11 8.936190     9
```

Question 2

16 points

Import the `addr.txt` file from the GitHub repository. This file contains a listing of names and addresses (thanks google). Parse each line to create a data.frame with the following columns: lastname, firstname, streetno, streetname, city, state, zip. Keep middle initials or abbreviated names in the firstname column. Print out the entire data.frame.

```
setwd("/Volumes/GoogleDrive/My Drive/Vanderbilt/1st Semester 2018-2019/BIOS6301 IntroStatComp/Homework_

library(readr)
addr <- readLines("addr.txt")

head(addr)
```

```
## [1] "Bania   Thomas M.       725 Commonwealth Ave.   Boston  MA      02215 "
## [2] "Barnaby         David   373 W. Geneva St.       Wms. Bay        WI      53191"
## [3] "Bausch  Judy    373 W. Geneva St.       Wms. Bay        WI      53191"
## [4] "Bolatto         Alberto         725 Commonwealth Ave.   Boston  MA      02215 "
## [5] "Carlstrom       John    933 E. 56th St.         Chicago         IL      60637"
## [6] "Chamberlin      Richard A.      111 Nowelo St.  Hilo    HI      96720"
```

```
length(addr)
```

```
## [1] 42
```

```
# split the string
addr_split <- strsplit(addr,"  ")

# get the index of the empty string
addr_list_index <- lapply(addr_split,function(x) x != "")

# remove the empty string in the list
addr_noempty_list <- sapply(1:length(addr_list_index), function(x) addr_split[[x]] <- addr_split[[x]][a

# remove the space in the string
addr_nospace_list <- trimws(addr_noempty_list)

# extract the elements
lastname <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[1,x])
firstname <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[2,x])
street <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[3,x])
streetno <- sub(" .*","",street)
streetname <- sub("^\\S+\\s+","",street)
city <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[4,x])
state <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[5,x])
zip <- sapply(1:length(addr_list_index), function(x) addr_nospace_list[6,x])

(data <- data.frame(lastname, firstname, streetno, streetname, city, state, zip))
```

```
##       lastname  firstname streetno           streetname       city state
## 1        Bania  Thomas M.      725    Commonwealth Ave.     Boston    MA
## 2      Barnaby      David      373        W. Geneva St.   Wms. Bay    WI
## 3       Bausch       Judy      373        W. Geneva St.   Wms. Bay    WI
## 4      Bolatto    Alberto      725    Commonwealth Ave.     Boston    MA
## 5    Carlstrom       John      933          E. 56th St.    Chicago    IL
## 6   Chamberlin Richard A.      111           Nowelo St.       Hilo    HI
## 7        Chuss       Dave     2145          Sheridan Rd   Evanston    IL
## 8        Davis      E. J.      933          E. 56th St.    Chicago    IL
```

```
## 9        Depoy     Darren      174         W. 18th Ave.   Columbus    OH
## 10     Griffin       Greg     5000           Forbes Ave. Pittsburgh    PA
## 11   Halvorsen       Nils      933           E. 56th St.    Chicago    IL
## 12      Harper         Al      373        W. Geneva St.   Wms. Bay    WI
## 13       Huang     Maohai      725 W. Commonwealth Ave.     Boston    MA
## 14     Ingalls   James G.      725 W. Commonwealth Ave.     Boston    MA
## 15     Jackson   James M.      725 W. Commonwealth Ave.     Boston    MA
## 16     Knudsen      Scott      373        W. Geneva St.   Wms. Bay    WI
## 17       Kovac       John     5640        S. Ellis Ave.    Chicago    IL
## 18   Landsberg      Randy     5640        S. Ellis Ave.    Chicago    IL
## 19          Lo  Kwok-Yung     1002         W. Green St.     Urbana    IL
## 20 Loewenstein  Robert F.      373        W. Geneva St.   Wms. Bay    WI
## 21       Lynch       John     4201          Wilson Blvd  Arlington    VA
## 22     Martini       Paul      174         W. 18th Ave.   Columbus    OH
## 23       Meyer    Stephan      933           E. 56th St.    Chicago    IL
## 24      Mrozek       Fred      373        W. Geneva St.   Wms. Bay    WI
## 25     Newcomb       Matt     5000           Forbes Ave. Pittsburgh    PA
## 26       Novak      Giles     2145           Sheridan Rd   Evanston    IL
## 27      Odalen      Nancy      373        W. Geneva St.   Wms. Bay    WI
## 28      Pernic       Dave      373        W. Geneva St.   Wms. Bay    WI
## 29      Pernic        Bob      373        W. Geneva St.   Wms. Bay    WI
## 30    Peterson    Jeffrey     5000           Forbes Ave. Pittsburgh    PA
## 31       Pryke       Clem      933           E. 56th St.    Chicago    IL
## 32      Rebull      Luisa     5640        S. Ellis Ave.    Chicago    IL
## 33   Renbarger     Thomas     2145           Sheridan Rd   Evanston    IL
## 34     Rottman        Joe     8730  W. Mountain View Ln  Littleton    CO
## 35   Schartman      Ethan      933           E. 56th St.    Chicago    IL
## 36       Spotz        Bob      373        W. Geneva St.   Wms. Bay    WI
## 37       Thoma       Mark      373        W. Geneva St.   Wms. Bay    WI
## 38      Walker      Chris      933        N. Cherry St.     Tucson    AZ
## 39      Wehrer     Cheryl     5000           Forbes Ave. Pittsburgh    PA
## 40       Wirth      Jesse      373        W. Geneva St.   Wms. Bay    WI
## 41      Wright       Greg      791  Holmdel-Keyport Rd.    Holmdel    NY
## 42     Zingale    Michael     5640        S. Ellis Ave.    Chicago    IL
##           zip
## 1       02215
## 2       53191
## 3       53191
## 4       02215
## 5       60637
## 6       96720
## 7  60208-3112
## 8       60637
## 9       43210
## 10      15213
## 11      60637
## 12      53191
## 13      02215
## 14      02215
## 15      02215
## 16      53191
## 17      60637
## 18      60637
## 19      61801
```

```
## 20      53191
## 21      22230
## 22      43210
## 23      60637
## 24      53191
## 25      15213
## 26 60208-3112
## 27      53191
## 28      53191
## 29      53191
## 30      15213
## 31      60637
## 32      60637
## 33 60208-3112
## 34      80125
## 35      60637
## 36      53191
## 37      53191
## 38      85721
## 39      15213
## 40      53191
## 41 07733-1988
## 42      60637
```

Question 3

3 points

The first argument to most functions that fit linear models are formulas. The following example defines the response variable `death` and allows the model to incorporate all other variables as terms. `.` is used to mean all columns not otherwise in the formula.

```
url <- "https://github.com/fonnesbeck/Bios6301/raw/master/datasets/haart.csv"
haart_df <- read.csv(url)[,c('death','weight','hemoglobin','cd4baseline')]
coef(summary(glm(death ~ ., data=haart_df, family=binomial(logit))))
```

```
##                 Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept)  3.576411744 1.226870535  2.915069 0.0035561039
## weight      -0.046210552 0.022556001 -2.048703 0.0404911395
## hemoglobin  -0.350642786 0.105064078 -3.337418 0.0008456055
## cd4baseline  0.002092582 0.001811959  1.154872 0.2481427160
```

Now imagine running the above several times, but with a different response and data set each time. Here's a function:

```
myfun <- function(dat, response) {
  form <- as.formula(response ~ .)
  coef(summary(glm(form, data=dat, family=binomial(logit))))
}
```

Unfortunately, it doesn't work. `tryCatch` is "catching" the error so that this file can be knit to PDF.

```
tryCatch(myfun(haart_df, death), error = function(e) e)
```

```
## <simpleError in eval(predvars, data, env): object 'death' not found>
```

What do you think is going on? Consider using `debug` to trace the problem.

```
debugonce(myfun)
myfun(haart_df, death)
traceback()
```

The reason why the function didn't work is that the response variable `death` couldn't be passed to the function since it's not defined in the global environment.

5 bonus points

Create a working function.

```
myfun <- function(dat, response) {
  form <- as.formula(paste(response, "~","." ))
  coef(summary(glm(form, data=dat, family=binomial(logit))))
}

myfun(haart_df, "death")
```

```
##                 Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept)  3.576411744 1.226870535  2.915069 0.0035561039
## weight      -0.046210552 0.022556001 -2.048703 0.0404911395
## hemoglobin  -0.350642786 0.105064078 -3.337418 0.0008456055
## cd4baseline  0.002092582 0.001811959  1.154872 0.2481427160
```