

Bios 6301: Assignment 5

Zi Ye

Due Tuesday, 15 November, 1:00 PM

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

50 points total.

Grade: 50/50

Submit a single knitr file (named `homework5.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 `homework5.rmd` or include author name may result in 5 points taken off.

Question 1

24 points

Import the HAART dataset (`haart.csv`) from the GitHub repository into R, and perform the following manipulations: (4 points each)

1. Convert date columns into a usable (for analysis) format. Use the `table` command to display the counts of the year from `init.date`.

```
#setwd('~Downloads/Biostat/Bios6301/datasets/')
haart <- read.csv('haart.csv', stringsAsFactors = F)
haart[, 'init.date'] <- as.POSIXct(haart[, 'init.date'], format='%m/%d/%y')
haart[, 'last.visit'] <- as.POSIXct(haart[, 'last.visit'], format='%m/%d/%y')
haart[, 'date.death'] <- as.POSIXct(haart[, 'date.death'], format='%m/%d/%y')
year <- format(haart[, 'init.date'], format = '%Y')
table(year)
```

```
## year
## 1998 2000 2001 2002 2003 2004 2005 2006 2007
##    1    5   17   60  270  292  207  104   44
```

2. Create an indicator variable (one which takes the values 0 or 1 only) to represent death within 1 year of the initial visit. How many observations died in year 1?

```
diff <- difftime(haart[, 'date.death'], haart[, 'init.date'], units = 'days')
diff[is.na(diff)] <- 366
deathinone <- c()
for (i in seq(nrow(haart))) {
  if (diff[i] < 365) {
    deathinone[i] <- 1} else {deathinone[i] <- 0}
}
table(deathinone)
```

```
## deathinone
##    0    1
##  908   92
```

- Use the `init.date`, `last.visit` and `death.date` columns to calculate a followup time (in days), which is the difference between the first and either the last visit or a death event (whichever comes first). If these times are longer than 1 year, censor them (this means if the value is above 365, set followup to 365). Print the quantile for this new variable.

```
diff_last <- difftime(haart[, 'last.visit'], haart[, 'init.date'], units = 'days')
diff_death <- difftime(haart[, 'date.death'], haart[, 'init.date'], units = 'days')
diff_death[is.na(diff_death)] <- 10000
diff_last[is.na(diff_last)] <- 10000
diff_sum <- c()
for (i in seq(nrow(haart))) {
  if (diff_last[i] < diff_death[i]) {
    diff_sum[i] <- diff_last[i]
  } else {
    diff_sum[i] <- diff_death[i]
  }
}
for (i in seq(length(diff_sum))) {
  if (diff_sum[i] > 365) diff_sum[i] <- 365
}
quantile(diff_sum)
```

```
##      0%      25%      50%      75%     100%
## 0.0000 320.7188 365.0000 365.0000 365.0000
```

- Create another indicator variable representing loss to followup; this means the observation is not known to be dead but does not have any followup visits after the first year. How many records are lost-to-followup?

```
count <- c()
for (i in seq(nrow(haart))) {
  if (diff_last[i] < 365 & diff_death[i] != 10000) {count[i] <- 1} else {count[i] <- 0}
}
sum(count==1)
```

```
## [1] 85
```

- Recall our work in class, which separated the `init.reg` field into a set of indicator variables, one for each unique drug. Create these fields and append them to the database as new columns. Which drug regimen are found over 100 times?

```
all.reg <- strsplit(haart[, 'init.reg'], ',')
all.reg <- unlist(all.reg)
all.reg <- unique(all.reg)
row.reg <- strsplit(haart[, 'init.reg'], ',')
user.reg <- +sapply(all.reg, function(j) sapply(row.reg, function(i) j %in% i))
haart <- cbind(haart, user.reg)
table(unlist(row.reg))[table(unlist(row.reg))>100]
```

```
##
## 3TC AZT D4T EFV NVP
## 973 794 146 516 358
```

- The dataset `haart2.csv` contains a few additional observations for the same study. Import these and append them to your master dataset (if you were smart about how you coded the previous steps, cleaning the additional observations should be easy!). Show the first five records and the last five records of the complete (and clean) data set.

```
#setwd('~Downloads/Biostat/Bios6301/datasets/')
haart2 <- read.csv('haart2.csv', stringsAsFactors = F)
haart2[, 'init.date'] <- as.POSIXct(haart2[, 'init.date'], format='%m/%d/%y')
haart2[, 'last.visit'] <- as.POSIXct(haart2[, 'last.visit'], format='%m/%d/%y')
haart2[, 'date.death'] <- as.POSIXct(haart2[, 'date.death'], format='%m/%d/%y')
row.reg2 <- strsplit(haart2[, 'init.reg'], ',')
user.reg2 <- +sapply(all.reg, function(j) sapply(row.reg2, function(i) j %in% i))
haart2 <- cbind(haart2, user.reg2)
haart3 <- rbind(haart, haart2)
head(haart3, 5)
```

```
##   male age aids cd4baseline logvl weight hemoglobin   init.reg
## 1    1  25    0          NA    NA          NA    3TC,AZT,EFV
## 2    1  49    0         143    NA 58.0608        11 3TC,AZT,EFV
## 3    1  42    1         102    NA 48.0816         1 3TC,AZT,EFV
## 4    0  33    0         107    NA 46.0000        NA 3TC,AZT,NVP
## 5    1  27    0          52     4     NA        NA 3TC,D4T,EFV
##   init.date last.visit death date.death 3TC AZT EFV NVP D4T ABC DDI IDV
## 1 2003-07-01 2007-02-26     0      <NA>  1  1  1  0  0  0  0  0
## 2 2004-11-23 2008-02-22     0      <NA>  1  1  1  0  0  0  0  0
## 3 2003-04-30 2005-11-21     1 2006-01-11  1  1  1  0  0  0  0  0
## 4 2006-03-25 2006-05-05     1 2006-05-07  1  1  0  1  0  0  0  0
## 5 2004-09-01 2007-11-13     0      <NA>  1  0  1  0  1  0  0  0
##   LPV RTV SQV FTC TDF DDC NFV T20 ATV FPV
## 1    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0
```

```
tail(haart3, 5)
```

```
##   male age aids cd4baseline logvl weight hemoglobin
## 1000  0 40.00000    1         131    NA 46.2672         8
## 1001  0 27.00000    0         232    NA     NA        NA
## 1002  1 38.72142    0         170    NA 84.0000        NA
## 1003  1 23.00000   NA         154 3.995635 65.5000        14
## 1004  0 31.00000    0         236    NA 45.8136        NA
##   init.reg init.date last.visit death date.death 3TC AZT EFV NVP
## 1000 3TC,D4T,NVP 2003-07-03 2008-02-29     0      <NA>  1  0  0  1
## 1001 3TC,AZT,NVP 2003-12-01 2004-01-05     0      <NA>  1  1  0  1
## 1002 3TC,AZT,NVP 2002-09-26 2004-03-29     0      <NA>  1  1  0  1
## 1003 3TC,DDI,EFV 2007-01-31 2007-04-16     0      <NA>  1  0  1  0
## 1004 3TC,D4T,NVP 2003-12-03 2007-10-11     0      <NA>  1  0  0  1
##   D4T ABC DDI IDV LPV RTV SQV FTC TDF DDC NFV T20 ATV FPV
## 1000  1  0  0  0  0  0  0  0  0  0  0  0  0  0
## 1001  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 1002  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 1003  0  0  1  0  0  0  0  0  0  0  0  0  0  0
## 1004  1  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Question 2

14 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: (2 points each)

1. Order the data set by id and dt.

```
x <- x[order(x$id),]
for (i in seq(max(x$id))) {
  x[x$id==i,] <- x[x$id==i,][order(x[x$id==i,]$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.

```
library(lubridate)

##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##      date

for (i in seq(max(x$id))) {
  for (j in seq(nrow(x[x$id==i,])-1)) {
    y <- as.numeric(difftime(x[x$id==i,][j+1,'dt'], x[x$id==i,][j,'dt'], units = 'days'))
    if (floor(y/365)>=1) {for (t in seq(floor(y/365))) {
      new <- data.frame('id' = i, 'dt' = (ymd_hms(x[x$id==i,][j,'dt'], tz='CST')+days(365*t)), 'a1c'=NA)
      x <- rbind(x, new)
    }}
  }
}

## Warning in as.POSIXct.POSIXlt(lt): unknown timezone 'CST'
## Warning in as.POSIXlt.POSIXct(ct): unknown timezone 'CST'
## Warning in as.POSIXlt.POSIXct(date): unknown timezone 'CST'
## Warning in as.POSIXct.POSIXlt(lt): unknown timezone 'CST'
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
## Warning in as.POSIXlt.POSIXct(ct): unknown timezone 'CST'
## Warning in as.POSIXct.POSIXlt(new): unknown timezone 'CST'
## Warning in as.POSIXct.POSIXlt(lt): unknown timezone 'CST'
## Warning in as.POSIXlt.POSIXct(ct): unknown timezone 'CST'
## Warning in as.POSIXlt.POSIXct(date): unknown timezone 'CST'
## Warning in as.POSIXct.POSIXlt(lt): unknown timezone 'CST'
## Warning in as.POSIXlt.POSIXct(ct): unknown timezone 'CST'
## Warning in as.POSIXct.POSIXlt(new): unknown timezone 'CST'
x <- x[order(x$id),]
for (i in seq(max(x$id))) {
  x[x$id==i,] <- x[x$id==i,][order(x[x$id==i,]$dt),]
}
```

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.

```
for (i in seq(max(x$id))) {
  visit <- nrow(x[x$id==i,])
  x[x$id==i, 'visit'] <- 1:visit
}
```

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

```
for (i in seq(max(x$id))) {
  x[x$id==i,][is.na(x[x$id==i,])] <- mean(x[x$id==i,]$a1c, na.rm = T)
}
```

5. Print mean a1c for each id.

```
a1cmean <- data.frame('id' = NA, 'mean' = NA)
for (i in seq(max(x$id))) {
  a1cmean[i,] <- data.frame('id' = i, 'mean' = mean(x[x$id==i,]$a1c, na.rm = T))
}
a1cmean
```

```
##      id      mean
## 1     1 4.063372
## 2     2 7.544643
## 3     3 6.757640
## 4     4 3.892127
## 5     5 9.512311
## 6     6 7.555965
## 7     7 9.161686
## 8     8 7.189064
## 9     9 9.283873
## 10    10 7.975217
## 11    11 6.917562
## 12    12 7.034021
## 13    13 9.145282
## 14    14 6.623756
## 15    15 8.012406
## 16    16 4.222158
```

```
## 17 17 3.996034
## 18 18 9.164873
## 19 19 5.507210
## 20 20 3.726675
## 21 21 8.140939
## 22 22 5.637501
## 23 23 7.366889
## 24 24 7.439316
## 25 25 6.877135
## 26 26 6.556759
## 27 27 4.926457
## 28 28 7.433917
## 29 29 4.508086
## 30 30 6.045577
## 31 31 7.116586
## 32 32 6.568791
## 33 33 6.494069
## 34 34 6.768615
## 35 35 8.476700
## 36 36 9.604410
## 37 37 9.606253
## 38 38 5.355979
## 39 39 6.917013
## 40 40 9.530136
## 41 41 9.802424
## 42 42 3.891770
## 43 43 6.095849
## 44 44 9.091670
## 45 45 6.737204
## 46 46 9.621763
## 47 47 9.231489
## 48 48 6.404600
## 49 49 6.096076
## 50 50 8.962319
```

6. Print total number of visits for each id.

```
tolvisit <- data.frame('id'= NA, 'visit'=NA)
for (i in seq(max(x$id))) {
  tolvisit[i,] <- data.frame('id'=i, 'visit'=max(x[x$id==i,]$visit))
}
tolvisit
```

```
##      id visit
## 1     1     11
## 2     2     20
## 3     3     14
## 4     4     12
## 5     5     14
## 6     6     10
## 7     7      9
## 8     8     12
## 9     9     11
## 10    10     12
## 11    11     10
```

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

7. Print the observations for id = 15.

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

```
##      id      dt      a1c visit
## 11   15 2000-04-30 00:34:50 7.527105    1
## 263  15 2001-01-17 21:11:02 5.898371    2
## 306  15 2001-04-25 06:23:05 8.566593    3
## 406  15 2002-04-25 01:23:05 8.012406    4
## 484  15 2003-04-25 01:23:05 8.012406    5
## 1117 15 2003-06-06 14:06:00 9.133769    6
## 1118 15 2004-06-05 09:06:00 8.012406    7
## 1119 15 2004-08-20 17:47:11 8.936190    8
```


Question 3

10 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('~Downloads/Biostat/Bios6301/datasets/')
addr <- read.table('addr.txt', sep='\t', stringsAsFactors = F)
addr_new <- data.frame()
for (i in seq(nrow(addr))) {
  lil <- unlist(strsplit(addr[i,], ' '))
  temp <- data.frame('lastname' = lil[1], 'firstname' = lil[2], 'streetno' = unlist(strsplit(lil[3], ' ')),
    addr_new <- rbind(addr_new, temp)
}
```

##	lastname	firstname	streetno	streetname	city	state
## 1	Bania	Thomas M.	725	Commonwealth Ave.	Boston	MA
## 2	Barnaby	David	373	W. Geneva	Wms. Bay	WI
## 3	Bausch	Judy	373	W. Geneva	Wms. Bay	WI
## 4	Bolatto	Alberto	725	Commonwealth Ave.	Boston	MA
## 5	Carlstrom	John	933	E. 56th	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	Chicago	IL
## 9	Depoy	Darren	174	W. 18th	Columbus	OH
## 10	Griffin	Greg	5000	Forbes Ave.	Pittsburgh	PA
## 11	Halvorsen	Nils	933	E. 56th	Chicago	IL
## 12	Harper	Al	373	W. Geneva	Wms. Bay	WI
## 13	Huang	Maohai	725	W. Commonwealth	Boston	MA
## 14	Ingalls	James G.	725	W. Commonwealth	Boston	MA
## 15	Jackson	James M.	725	W. Commonwealth	Boston	MA
## 16	Knudsen	Scott	373	W. Geneva	Wms. Bay	WI
## 17	Kovac	John	5640	S. Ellis	Chicago	IL
## 18	Landsberg	Randy	5640	S. Ellis	Chicago	IL
## 19	Lo	Kwok-Yung	1002	W. Green	Urbana	IL
## 20	Loewenstein	Robert F.	373	W. Geneva	Wms. Bay	WI
## 21	Lynch	John	4201	Wilson Blvd	Arlington	VA
## 22	Martini	Paul	174	W. 18th	Columbus	OH
## 23	Meyer	Stephan	933	E. 56th	Chicago	IL
## 24	Mrozek	Fred	373	W. Geneva	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	Wms. Bay	WI
## 28	Pernic	Dave	373	W. Geneva	Wms. Bay	WI
## 29	Pernic	Bob	373	W. Geneva	Wms. Bay	WI
## 30	Peterson	Jeffrey	5000	Forbes Ave.	Pittsburgh	PA
## 31	Pryke	Clem	933	E. 56th	Chicago	IL
## 32	Rebull	Luisa	5640	S. Ellis	Chicago	IL
## 33	Renbarger	Thomas	2145	Sheridan Rd	Evanston	IL
## 34	Rottman	Joe	8730	W. Mountain	Littleton	CO
## 35	Schartman	Ethan	933	E. 56th	Chicago	IL

## 36	Spotz	Bob	373	W. Geneva	Wms. Bay	WI
## 37	Thoma	Mark	373	W. Geneva	Wms. Bay	WI
## 38	Walker	Chris	933	N. Cherry	Tucson	AZ
## 39	Wehrer	Cheryl	5000	Forbes Ave.	Pittsburgh	PA
## 40	Wirth	Jesse	373	W. Geneva	Wms. Bay	WI
## 41	Wright	Greg	791	Holmdel-Keyport Rd.	Holmdel	NY
## 42	Zingale	Michael	5640	S. Ellis	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 4

2 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(expr, envir, enclos): object 'death' not found>
```

What do you think is going on? Consider using `debug` to trace the problem. **Looks like there is a problem in using the function `glm`. Deleting the argument `'family=binomial(logit)'` will make the function work again**

5 bonus points

Create a working function.

```
myfun1 <- function(dat, response) {
  form <- as.formula(response ~ .)
  coef(summary(glm(form, data=dat)))
}
```

JC Grading +0