# Bios 6301: Assignment 5

*Ying Ji*

*Due Tuesday, 15 November, 1:00 PM*

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

50 points total.

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)

```
haart<-read.csv('https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/haart.csv',string
haart[,'last.visit']<-as.Date(haart[,'last.visit'],"%m/%d/%y")
haart[,'init.date']<-as.Date(haart[,'init.date'],"%m/%d/%y")
haart[,'date.death']<-as.Date(haart[,'date.death'],"%m/%d/%y")
```

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

```
#1. counts of year from 'init.date'
table(format(haart[,'init.date'],"%Y"))
```

```
##
## 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?

```
haart[,'deathinyear']<-rep(0,nrow(haart))

        for (i in 1:nrow(haart)) {
        if (!is.na(haart[i,'date.death'])){
                if ( (haart[i,'date.death'] - haart[i,'init.date']) <= 365) {
                haart[i,'deathinyear']<-1
                }
        }
        }
#92 died in 1 year
sum(haart[,'deathinyear'])
```

```
## [1] 92
```

3. Use the `init.date`, `last.visit` and `date.death` 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.

```
    haart[,'followup']<-numeric(nrow(haart))
for (i in 1:nrow(haart)) {
        if (!is.na(haart[i,'date.death']) && !is.na(haart[i,'last.visit']) ){
                haart[i,'followup']<-min( (haart[i,'date.death']-haart[i,'init.date']), (haart[i,'last.
                }
        else if ( is.na(haart[i,'date.death']) && !is.na(haart[i,'last.visit']) ){
                haart[i,'followup']<-(haart[i,'last.visit']-haart[i,'init.date'])
                }
        else if ( !is.na(haart[i,'date.death']) && is.na(haart[i,'last.visit']) ){
                haart[i,'followup']<-(haart[i,'date.death']-haart[i,'init.date'])
                }
}

haart[,'followup'][haart[,'followup']>365]<-365
#see the quantile
quantile(haart[,'followup'])
```

```
##     0%    25%    50%    75%   100%
##   0.00 320.75 365.00 365.00 365.00
```

4. 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?

```
    haart[,'loss']<-numeric(nrow(haart))
for (i in 1:nrow(haart)){
        if ( (haart[i,'death']==0) && (haart[i,'last.visit']-haart[i,'init.date'])<=365 ){
                haart[i,'loss']=1
        }
}
```

5. 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?

```
#write a function "splitdrug"
    row.reg<-strsplit(haart[,'init.reg'],',')
all.reg<-unique( unlist(strsplit(haart[,'init.reg'],',')))
user.reg<-sapply(all.reg,function(j) sapply(row.reg,function(i) j %in% i) )

haart<-cbind(haart,+user.reg)
# drug found over 100 times: 3TC,AZT,EFV,NVP,D4T
colSums(user.reg)>100
```

```
##   3TC   AZT   EFV   NVP   D4T   ABC   DDI   IDV   LPV   RTV   SQV   FTC
##  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##   TDF   DDC   NFV   T20   ATV   FPV
## FALSE FALSE FALSE FALSE FALSE FALSE
```

6. 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.

```
haart2<-read.csv('https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/haart2.csv',str
haart1<-read.csv('https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/haart.csv',strin
haart3<-rbind(haart1,haart2)
haart3[,'last.visit']<-as.Date(haart3[,'last.visit'],"%m/%d/%y")
haart3[,'init.date']<-as.Date(haart3[,'init.date'],"%m/%d/%y")
haart3[,'date.death']<-as.Date(haart3[,'date.death'],"%m/%d/%y")
haart3[,'deathinyear']<-rep(0,nrow(haart3))

        for (i in 1:nrow(haart3)) {
        if (!is.na(haart3[i,'date.death'])){
                if ( (haart3[i,'date.death'] - haart3[i,'init.date']) <= 365) {
                haart3[i,'deathinyear']<-1
                }
        }
        }

haart3[,'followup']<-numeric(nrow(haart3))
for (i in 1:nrow(haart3)) {
        if (!is.na(haart3[i,'date.death']) && !is.na(haart3[i,'last.visit']) ){
                haart3[i,'followup']<-min( (haart3[i,'date.death']-haart3[i,'init.date']), (haart3[i,'la
                }
        else if ( is.na(haart3[i,'date.death']) && !is.na(haart3[i,'last.visit']) ){
                haart3[i,'followup']<-(haart3[i,'last.visit']-haart3[i,'init.date'])
                }
        else if ( !is.na(haart3[i,'date.death']) && is.na(haart3[i,'last.visit']) ){
                haart3[i,'followup']<-(haart3[i,'date.death']-haart3[i,'init.date'])
                }
}

haart3[,'followup'][haart3[,'followup']>365]<-365

haart3[,'loss']<-numeric(nrow(haart3))
for (i in 1:nrow(haart3)){
        if ( (haart3[i,'death']==0) && (haart3[i,'last.visit']-haart3[i,'init.date'])<=365 ){
                haart3[i,'loss']=1
        }
}

 row.reg<-strsplit(haart3[,'init.reg'],',')
all.reg<-unique( unlist(strsplit(haart3[,'init.reg'],',')))
user.reg<-sapply(all.reg,function(j) sapply(row.reg,function(i) j %in% i) )
haart3<-cbind(haart3,+user.reg)

head(haart3,5)
```

```
##   male age aids cd4baseline logvl  weight hemoglobin    init.reg
## 1    1  25    0          NA    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 deathinyear followup loss 3TC AZT
## 1 2003-07-01 2007-02-26     0       <NA>           0      365    0   1   1
## 2 2004-11-23 2008-02-22     0       <NA>           0      365    0   1   1
## 3 2003-04-30 2005-11-21     1 2006-01-11           0      365    0   1   1
## 4 2006-03-25 2006-05-05     1 2006-05-07           1       41    0   1   1
## 5 2004-09-01 2007-11-13     0       <NA>           0      365    0   1   0
##   EFV NVP D4T ABC DDI IDV LPV RTV SQV FTC TDF DDC NFV T20 ATV FPV
## 1   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 2   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 3   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 4   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 5   1   0   1   0   0   0   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 deathinyear
## 1000 3TC,D4T,NVP 2003-07-03 2008-02-29     0       <NA>           0
## 1001 3TC,AZT,NVP 2003-12-01 2004-01-05     0       <NA>           0
## 1002 3TC,AZT,NVP 2002-09-26 2004-03-29     0       <NA>           0
## 1003 3TC,DDI,EFV 2007-01-31 2007-04-16     0       <NA>           0
## 1004 3TC,D4T,NVP 2003-12-03 2007-10-11     0       <NA>           0
##      followup loss 3TC AZT EFV NVP D4T ABC DDI IDV LPV RTV SQV FTC TDF DDC
## 1000      365    0   1   0   0   1   1   0   0   0   0   0   0   0   0   0
## 1001       35    1   1   1   0   1   0   0   0   0   0   0   0   0   0   0
## 1002      365    0   1   1   0   1   0   0   0   0   0   0   0   0   0   0
## 1003       75    1   1   0   1   0   0   0   1   0   0   0   0   0   0   0
## 1004      365    0   1   0   0   1   1   0   0   0   0   0   0   0   0   0
##      NFV T20 ATV FPV
## 1000   0   0   0   0
## 1001   0   0   0   0
## 1002   0   0   0   0
## 1003   0   0   0   0
## 1004   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_sorted<-x[order(x[,'id'],x[,'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
```

```
numid<-as.numeric(levels(factor(x_sorted[,'id'])))
d<-data.frame(id=numeric(),dt=numeric(),a1c=numeric())
for (i in seq_along(numid) ){
observe<-subset(x_sorted,id==i)

        for (j in 2:nrow(observe)){
                year<-numeric()
                #use floor to get 0 if less than 1 year interval
                year[j]<-floor( (observe$dt[j]-observe$dt[j-1])/dyears(1) )

                if ( year[j] >0) {

                        id=rep(observe$id[j-1],year[j])
                        a1c=rep(NA,year[j])
```

```
                        dt<-numeric()
                        #deal with different years
                        for (k in 1:year[j]){
                                dt<-append(dt,observe$dt[j-1]+dyears(k))
                        }
        d1<-data.frame(id=id,dt=dt,a1c=a1c)


        d<-rbind(d,d1)
        }
}
}
y<-rbind(x_sorted,d)
#change dt to POSIXct form
y$dt<-as.POSIXct(y$dt, format = "%y-%m-%d %H:%M:%S")
#reorder
y<-y[order(y[,'id'],y[,'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.

```
num<-as.numeric(levels(factor(y[,'id'])))
y<-cbind(y,visit=numeric(nrow(y)))
for (i in seq_along(num) ) {
observe<-subset(y,id==i)

y[y$id==i,]$visit<-seq(nrow(observe))

 }
```

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

```
for (i in 1:nrow(y)) {
  if(is.na(y$a1c[i])) {
    y$a1c[i] <- mean(y$a1c[which(y$id == y$id[i])], na.rm = TRUE)
  }
}
```

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

```
num<-as.numeric(levels(factor(y[,'id'])))
for (i in seq_along(num)) {

  print(cbind(i,mean(y$a1c[y$id == i])))
}
```

```
##      i
## [1,] 1 4.063372
##      i
## [1,] 2 7.544643
```

```
##      i
## [1,] 3 6.75764
##      i
## [1,] 4 3.892127
##      i
## [1,] 5 9.512311
##      i
## [1,] 6 7.555965
##      i
## [1,] 7 9.161686
##      i
## [1,] 8 7.189064
##      i
## [1,] 9 9.283873
##       i
## [1,] 10 7.975217
##       i
## [1,] 11 6.917562
##       i
## [1,] 12 7.034021
##       i
## [1,] 13 9.145282
##       i
## [1,] 14 6.623756
##       i
## [1,] 15 8.012406
##       i
## [1,] 16 4.222158
##       i
## [1,] 17 3.996034
##       i
## [1,] 18 9.164873
##       i
## [1,] 19 5.50721
##       i
## [1,] 20 3.726675
##       i
## [1,] 21 8.140939
##       i
## [1,] 22 5.637501
##       i
## [1,] 23 7.366889
##       i
## [1,] 24 7.439316
##       i
## [1,] 25 6.877135
##       i
## [1,] 26 6.556759
##       i
## [1,] 27 4.926457
##       i
## [1,] 28 7.433917
##       i
## [1,] 29 4.508086
```

```
##       i
## [1,] 30 6.045577
##       i
## [1,] 31 7.116586
##       i
## [1,] 32 6.568791
##       i
## [1,] 33 6.494069
##       i
## [1,] 34 6.768615
##       i
## [1,] 35 8.4767
##       i
## [1,] 36 9.60441
##       i
## [1,] 37 9.606253
##       i
## [1,] 38 5.355979
##       i
## [1,] 39 6.917013
##       i
## [1,] 40 9.530136
##       i
## [1,] 41 9.802424
##       i
## [1,] 42 3.89177
##       i
## [1,] 43 6.095849
##       i
## [1,] 44 9.09167
##       i
## [1,] 45 6.737204
##       i
## [1,] 46 9.621763
##       i
## [1,] 47 9.231489
##       i
## [1,] 48 6.4046
##       i
## [1,] 49 6.096076
##       i
## [1,] 50 8.962319
```

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

```
num<-as.numeric(levels(factor(y[,'id'])))
for (i in num) {

  print(cbind(i,max(y$visit[y$id == i])))
}
```

```
##      i
## [1,] 1 11
```

```
##      i
## [1,] 2 20
##      i
## [1,] 3 14
##      i
## [1,] 4 12
##      i
## [1,] 5 14
##      i
## [1,] 6 10
##      i
## [1,] 7 9
##      i
## [1,] 8 12
##      i
## [1,] 9 11
##       i
## [1,] 10 12
##       i
## [1,] 11 10
##       i
## [1,] 12 10
##       i
## [1,] 13 8
##       i
## [1,] 14 12
##       i
## [1,] 15 8
##       i
## [1,] 16 9
##       i
## [1,] 17 12
##       i
## [1,] 18 10
##       i
## [1,] 19 10
##       i
## [1,] 20 9
##       i
## [1,] 21 10
##       i
## [1,] 22 8
##       i
## [1,] 23 8
##       i
## [1,] 24 15
##       i
## [1,] 25 12
##       i
## [1,] 26 14
##       i
## [1,] 27 11
##       i
## [1,] 28 14
```

```
##       i
## [1,] 29 10
##       i
## [1,] 30 7
##       i
## [1,] 31 11
##       i
## [1,] 32 5
##       i
## [1,] 33 8
##       i
## [1,] 34 12
##       i
## [1,] 35 11
##       i
## [1,] 36 9
##       i
## [1,] 37 17
##       i
## [1,] 38 15
##       i
## [1,] 39 8
##       i
## [1,] 40 7
##       i
## [1,] 41 17
##       i
## [1,] 42 14
##       i
## [1,] 43 11
##       i
## [1,] 44 11
##       i
## [1,] 45 14
##       i
## [1,] 46 9
##       i
## [1,] 47 12
##       i
## [1,] 48 11
##       i
## [1,] 49 12
##       i
## [1,] 50 10
```

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

```
print(y[y$id == 15, ])
```

```
##      id                  dt      a1c visit
## 11   15 2000-04-30 00:34:50 7.527105     1
## 406  15 2001-01-17 21:11:02 5.898371     2
## 306  15 2001-04-25 06:23:05 8.566593     3
```

```
## 1810 15 2002-04-25 06:23:05 8.012406     4
## 1910 15 2003-04-25 06:23:05 8.012406     5
## 484  15 2003-06-06 14:06:00 9.133769     6
## 2010 15 2004-06-05 14:06:00 8.012406     7
## 263  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.

```
addr<-read.table('https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/addr.txt',sep="'
temp<-unlist(strsplit(addr[,1],"  "))
#delete the space at the begining or at the end of each string
delspace <- function (x) gsub("^\\s+|\\s+$", "", x)
temp<-delspace(temp)
#save only non space items,make into matrix
temp<-temp[temp!=""]
address<-matrix(temp,ncol=6,byrow=T)
streetno=sub("^(\\w+)\\s?(.*)$","\\1",address[,3])
streetname=sub("^(\\w+)\\s?(.*)$","\\2",address[,3])
address<-cbind(address,streetno,streetname)
address<-address[,-3]
colnames(address)<-c("lastname", "firstname", "city", "state", "zip","streetno", "streetname")
address<-address[,c("lastname", "firstname","streetno", "streetname", "city", "state", "zip")]
print(address)
```

```
##       lastname      firstname    streetno streetname
##  [1,] "Bania"       "Thomas M."  "725"    "Commonwealth Ave."
##  [2,] "Barnaby"     "David"      "373"    "W. Geneva St."
##  [3,] "Bausch"      "Judy"       "373"    "W. Geneva St."
##  [4,] "Bolatto"     "Alberto"    "725"    "Commonwealth Ave."
##  [5,] "Carlstrom"   "John"       "933"    "E. 56th St."
##  [6,] "Chamberlin"  "Richard A." "111"    "Nowelo St."
##  [7,] "Chuss"       "Dave"       "2145"   "Sheridan Rd"
##  [8,] "Davis"       "E. J."      "933"    "E. 56th St."
##  [9,] "Depoy"       "Darren"     "174"    "W. 18th Ave."
## [10,] "Griffin"     "Greg"       "5000"   "Forbes Ave."
## [11,] "Halvorsen"   "Nils"       "933"    "E. 56th St."
## [12,] "Harper"      "Al"         "373"    "W. Geneva St."
## [13,] "Huang"       "Maohai"     "725"    "W. Commonwealth Ave."
## [14,] "Ingalls"     "James G."   "725"    "W. Commonwealth Ave."
## [15,] "Jackson"     "James M."   "725"    "W. Commonwealth Ave."
## [16,] "Knudsen"     "Scott"      "373"    "W. Geneva St."
## [17,] "Kovac"       "John"       "5640"   "S. Ellis Ave."
## [18,] "Landsberg"   "Randy"      "5640"   "S. Ellis Ave."
## [19,] "Lo"          "Kwok-Yung"  "1002"   "W. Green St."
## [20,] "Loewenstein" "Robert F."  "373"    "W. Geneva St."
## [21,] "Lynch"       "John"       "4201"   "Wilson Blvd"
```

```
## [22,] "Martini"     "Paul"       "174"    "W. 18th Ave."
## [23,] "Meyer"       "Stephan"    "933"    "E. 56th St."
## [24,] "Mrozek"      "Fred"       "373"    "W. Geneva St."
## [25,] "Newcomb"     "Matt"       "5000"   "Forbes Ave."
## [26,] "Novak"       "Giles"      "2145"   "Sheridan Rd"
## [27,] "Odalen"      "Nancy"      "373"    "W. Geneva St."
## [28,] "Pernic"      "Dave"       "373"    "W. Geneva St."
## [29,] "Pernic"      "Bob"        "373"    "W. Geneva St."
## [30,] "Peterson"    "Jeffrey"    "5000"   "Forbes Ave."
## [31,] "Pryke"       "Clem"       "933"    "E. 56th St."
## [32,] "Rebull"      "Luisa"      "5640"   "S. Ellis Ave."
## [33,] "Renbarger"   "Thomas"     "2145"   "Sheridan Rd"
## [34,] "Rottman"     "Joe"        "8730"   "W. Mountain View Ln"
## [35,] "Schartman"   "Ethan"      "933"    "E. 56th St."
## [36,] "Spotz"       "Bob"        "373"    "W. Geneva St."
## [37,] "Thoma"       "Mark"       "373"    "W. Geneva St."
## [38,] "Walker"      "Chris"      "933"    "N. Cherry St."
## [39,] "Wehrer"      "Cheryl"     "5000"   "Forbes Ave."
## [40,] "Wirth"       "Jesse"      "373"    "W. Geneva St."
## [41,] "Wright"      "Greg"       "791"    "Holmdel-Keyport Rd."
## [42,] "Zingale"     "Michael"    "5640"   "S. Ellis Ave."
##       city         state zip
##  [1,] "Boston"     "MA"  "02215"
##  [2,] "Wms. Bay"   "WI"  "53191"
##  [3,] "Wms. Bay"   "WI"  "53191"
##  [4,] "Boston"     "MA"  "02215"
##  [5,] "Chicago"    "IL"  "60637"
##  [6,] "Hilo"       "HI"  "96720"
##  [7,] "Evanston"   "IL"  "60208-3112"
##  [8,] "Chicago"    "IL"  "60637"
##  [9,] "Columbus"   "OH"  "43210"
## [10,] "Pittsburgh" "PA"  "15213"
## [11,] "Chicago"    "IL"  "60637"
## [12,] "Wms. Bay"   "WI"  "53191"
## [13,] "Boston"     "MA"  "02215"
## [14,] "Boston"     "MA"  "02215"
## [15,] "Boston"     "MA"  "02215"
## [16,] "Wms. Bay"   "WI"  "53191"
## [17,] "Chicago"    "IL"  "60637"
## [18,] "Chicago"    "IL"  "60637"
## [19,] "Urbana"     "IL"  "61801"
## [20,] "Wms. Bay"   "WI"  "53191"
## [21,] "Arlington"  "VA"  "22230"
## [22,] "Columbus"   "OH"  "43210"
## [23,] "Chicago"    "IL"  "60637"
## [24,] "Wms. Bay"   "WI"  "53191"
## [25,] "Pittsburgh" "PA"  "15213"
## [26,] "Evanston"   "IL"  "60208-3112"
## [27,] "Wms. Bay"   "WI"  "53191"
## [28,] "Wms. Bay"   "WI"  "53191"
## [29,] "Wms. Bay"   "WI"  "53191"
## [30,] "Pittsburgh" "PA"  "15213"
## [31,] "Chicago"    "IL"  "60637"
## [32,] "Chicago"    "IL"  "60637"
```

```
## [33,] "Evanston"   "IL"  "60208-3112"
## [34,] "Littleton"  "CO"  "80125"
## [35,] "Chicago"    "IL"  "60637"
## [36,] "Wms. Bay"   "WI"  "53191"
## [37,] "Wms. Bay"   "WI"  "53191"
## [38,] "Tucson"     "AZ"  "85721"
## [39,] "Pittsburgh" "PA"  "15213"
## [40,] "Wms. Bay"   "WI"  "53191"
## [41,] "Holmdel"    "NY"  "07733-1988"
## [42,] "Chicago"    "IL"  "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.

The problem is that function "as.formula"" requires a object variable in "character" form, we can convert 'death' into character form by "paste" to solve the problem.

**5 bonus points**

Create a working function.

```r
myfun_1 <- function(dat, response) {
  form <- as.formula (paste(response, "~."))
  coef(summary(glm(form, data=dat, family=binomial(logit))))
 }
myfun_1(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
```