Bios 6301: Assignment 5

Elizabeth Sigworth

Tuesday 15 November

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)

```
url1 <- "https://github.com/fonnesbeck/Bios6301/raw/master/datasets/haart.csv"
haart <- read.csv(url1,stringsAsFactors=FALSE)
```

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

```
haart$init.date <- as.Date(haart$init.date, format="%m/%d/%y")
haart$last.visit <- as.Date(haart$last.visit, format="%m/%d/%y")
haart$date.death <- as.Date(haart$date.death, format="%m/%d/%y")
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?

```
for (i in 1:nrow(haart)) {
  haart$one.year[i] <- ifelse(abs(unclass(difftime(haart$init.date[i],
                                        haart$date.death[i], units='days'))[1]) > 365, 0, 1)
}
year.one <- sum(haart$one.year,na.rm = TRUE)
```

In this data, 92 observations died in year 1.

3. 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.

```
for (i in 1:nrow(haart)){
  if(is.na(haart$date.death[i]) == TRUE ) {
    difference <- unclass(difftime(haart$last.visit[i], haart$init.date[i], 'days'))[1]
    haart$follow.up[i] <- min(365,difference)
  }
  else if(is.na(haart$date.death[i]) == FALSE && is.na(haart$last.visit[i])==TRUE){
    difference <- unclass(difftime(haart$date.death[i], haart$init.date[i], 'days'))[1]
    haart$follow.up[i] <- min(365,difference)
  }
  else {
    first <- min(haart$last.visit[i],haart$date.death[i])
    difference <- unclass(difftime(first, haart$init.date[i], 'days'))[1]
    haart$follow.up[i] <- min(365,difference)
  }
}
quantile(haart$follow.up)
```

```
##     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?

```
for (i in 1:nrow(haart)){
  if(is.na(haart$date.death[i]) && haart$follow.up[i] < 365){
    haart$lost[i] <- 1
  }
  else {
    haart$lost[i] <- 0
  }
}
lost.to <- sum(haart$lost, na.rm=TRUE)
```

There were 173 records lost-to-followup.

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?

```
init.reg <- as.character(haart[,'init.reg'])
haart[['init.reg_list']] <- strsplit(init.reg, ",")
(all_drugs <- unique(unlist(haart$init.reg_list)))
```

```
##  [1] "3TC" "AZT" "EFV" "NVP" "D4T" "ABC" "DDI" "IDV" "LPV" "RTV" "SQV"
## [12] "FTC" "TDF" "DDC" "NFV" "T20" "ATV" "FPV"
```

```r
(unique_drugs <- unique(unlist(haart$init.reg_list)))
```

```
##  [1] "3TC" "AZT" "EFV" "NVP" "D4T" "ABC" "DDI" "IDV" "LPV" "RTV" "SQV"
## [12] "FTC" "TDF" "DDC" "NFV" "T20" "ATV" "FPV"
```

```r
reg_drugs <- matrix(FALSE, nrow=nrow(haart), ncol=length(all_drugs))
for(i in seq_along(all_drugs)) {
  reg_drugs[,i] <- sapply(haart$init.reg_list, function(x) all_drugs[i] %in% x)
}
reg_drugs <- data.frame(reg_drugs)
names(reg_drugs) <- all_drugs
haart_merged <- cbind(haart, reg_drugs)
for (i in 17:34){
  for (j in 1:nrow(haart_merged)){
    if(haart_merged[j,i]==TRUE){
      haart_merged[j,i] <- colnames(haart_merged)[i]
    }
    else {
      haart_merged[j,i] <- NA
    }
  }
}

apply(X = haart_merged[,17:34],2,table)
```

```
## 3TC AZT EFV NVP D4T ABC DDI IDV LPV RTV SQV FTC TDF DDC NFV T20 ATV FPV
## 973 794 516 358 146  56  38  27  31  79  29   8  10   1   8   1   2   2
```

The individual drugs 3TC, AZT, EFV, NVP, and D4T are each found over 100 times.

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.

```r
url2 <- "https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/haart2.csv"
haart2 <- read.csv(url2, stringsAsFactors=FALSE)
haart2$init.date <- as.Date(haart2$init.date, format="%m/%d/%y")
haart2$last.visit <- as.Date(haart2$last.visit, format="%m/%d/%y")
haart2$date.death <- as.Date(haart2$date.death, format="%m/%d/%y")
for (i in 1:nrow(haart2)) {
  haart2$one.year[i] <- ifelse(abs(unclass(difftime(haart2$init.date[i], haart2$date.death[i],
                                                    units='days'))[1]) > 365, 0, 1)
}
for (i in 1:nrow(haart2)){
  if(is.na(haart2$date.death[i]) == TRUE) {
    difference <- unclass(difftime(haart2$last.visit[i], haart2$init.date[i], 'days'))[1]
    haart2$follow.up[i] <- min(365,difference)
  }
  else {
    difference <- unclass(difftime(haart2$date.death[i], haart2$init.date[i], 'days'))[1]
```

```r
    haart2$follow.up[i] <- min(365,difference)
  }
}
for (i in 1:nrow(haart2)){
  if(is.na(haart2$date.death[i]) && unclass(difftime(haart2$last.visit[i],
                                                     haart2$init.date[i], 'days'))[1] < 365){
    haart2$lost[i] <- 1
  }
  else {
    haart2$lost[i] <- 0
  }
}

init.reg <- as.character(haart2[,'init.reg'])
haart2[['init.reg_list']] <- strsplit(init.reg, ",")

reg_drugs <- matrix(FALSE, nrow=nrow(haart2), ncol=length(all_drugs))
for(i in seq_along(all_drugs)) {
  reg_drugs[,i] <- sapply(haart2$init.reg_list, function(x) all_drugs[i] %in% x)
}
reg_drugs <- data.frame(reg_drugs)
names(reg_drugs) <- all_drugs
haart2_merged <- cbind(haart2, reg_drugs)

for (i in 17:34){
  for (j in 1:nrow(haart2_merged)){
    if(haart2_merged[j,i]==TRUE){
      haart2_merged[j,i] <- colnames(haart2_merged)[i]
    }
    else {
      haart2_merged[j,i] <- NA
    }
  }
}


haart_final <- rbind(haart_merged,haart2_merged)
haart_final[c(1:5,1000:1004),]
```

```
##      male      age aids cd4baseline    logvl  weight hemoglobin
## 1       1 25.00000    0          NA       NA      NA         NA
## 2       1 49.00000    0         143       NA 58.0608         11
## 3       1 42.00000    1         102       NA 48.0816          1
## 4       0 33.00000    0         107       NA 46.0000         NA
## 5       1 27.00000    0          52 4.000000      NA         NA
## 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 one.year follow.up
## 1    3TC,AZT,EFV 2003-07-01 2007-02-26     0       <NA>       NA       365
## 2    3TC,AZT,EFV 2004-11-23 2008-02-22     0       <NA>       NA       365
```

```
## 3    3TC,AZT,EFV 2003-04-30 2005-11-21     1 2006-01-11        0       365
## 4    3TC,AZT,NVP 2006-03-25 2006-05-05     1 2006-05-07        1        41
## 5    3TC,D4T,EFV 2004-09-01 2007-11-13     0       <NA>       NA       365
## 1000 3TC,D4T,NVP 2003-07-03 2008-02-29     0       <NA>       NA       365
## 1001 3TC,AZT,NVP 2003-12-01 2004-01-05     0       <NA>       NA        35
## 1002 3TC,AZT,NVP 2002-09-26 2004-03-29     0       <NA>       NA       365
## 1003 3TC,DDI,EFV 2007-01-31 2007-04-16     0       <NA>       NA        75
## 1004 3TC,D4T,NVP 2003-12-03 2007-10-11     0       <NA>       NA       365
##      lost init.reg_list 3TC  AZT  EFV  NVP  D4T  ABC  DDI  IDV  LPV  RTV
## 1       0 3TC, AZT, EFV 3TC  AZT  EFV <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 2       0 3TC, AZT, EFV 3TC  AZT  EFV <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 3       0 3TC, AZT, EFV 3TC  AZT  EFV <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 4       0 3TC, AZT, NVP 3TC  AZT <NA>  NVP <NA> <NA> <NA> <NA> <NA> <NA>
## 5       0 3TC, D4T, EFV 3TC <NA>  EFV <NA>  D4T <NA> <NA> <NA> <NA> <NA>
## 1000    0 3TC, D4T, NVP 3TC <NA> <NA>  NVP  D4T <NA> <NA> <NA> <NA> <NA>
## 1001    1 3TC, AZT, NVP 3TC  AZT <NA>  NVP <NA> <NA> <NA> <NA> <NA> <NA>
## 1002    0 3TC, AZT, NVP 3TC  AZT <NA>  NVP <NA> <NA> <NA> <NA> <NA> <NA>
## 1003    1 3TC, DDI, EFV 3TC <NA>  EFV <NA> <NA> <NA>  DDI <NA> <NA> <NA>
## 1004    0 3TC, D4T, NVP 3TC <NA> <NA>  NVP  D4T <NA> <NA> <NA> <NA> <NA>
##       SQV  FTC  TDF  DDC  NFV  T20  ATV  FPV
## 1    <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 2    <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 3    <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 4    <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 5    <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 1000 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 1001 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 1002 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 1003 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
## 1004 <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
```

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$id,x$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.

```
#Write a function that finds gaps
check.dates <- function(identity,date){
  insert.at <- vector()
  rows.fin <- vector()
  for (i in unique(identity)){
    rows <- which(identity==i)[1:length(which(identity==i))-1]
    for (j in rows){
      rows.fin <- c(rows.fin, j)
      if(unclass(difftime(date[j+1], date[j], "days"))[1] > 366){
        insert.at <- c(insert.at,j+1)
      }
    }
  }
  return(insert.at)
}

#Write a function that fills gaps
add.gap <- function(df,insertion){
    df <- rbind(df[1:(insertion-1),],data.frame(id=df$id[insertion-1],
                                          dt=df$dt[insertion-1]+years(1),a1c=NA),
                df[insertion:nrow(df),])
  return(df)
}

p <- x
insert.at <- check.dates(p$id,p$dt)
lines <- insert.at+seq(from=0,by=1,length.out=length(insert.at))
for (i in 1:length(lines)){
  p <- add.gap(p,lines[i])
}

#Check again to fix 2-year gaps
insert.at <- check.dates(p$id,p$dt)
lines <- insert.at+seq(from=0,by=1,length.out=length(insert.at))
for (i in 1:length(lines)){
  p <- add.gap(p,lines[i])
}

#Check for any 3-year gaps
(insert.at <- check.dates(p$id,p$dt))
```

```
## logical(0)
```

```
x <- p
```

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 1:length(unique(x$id))){
  visits <- seq(1:table(x$id)[[i]])
  x$visit[x$id==i] <- visits
}
```

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

```
for (i in 1:length(unique(x$id))){
  rows <- which(x$id==i)
  meana1c <- mean(x$a1c[rows[1]:tail(rows,n=1)],na.rm = TRUE)
  for (j in rows){
    if(is.na(x$a1c[j])){
      x$a1c[j] <- meana1c
    }
  }
}
```

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

```
for (i in 1:length(unique(x$id))){
  rows <- which(x$id==i)
  meana1c <- mean(x$a1c[rows[1]:tail(rows,n=1)])
  print(c(as.integer(i),meana1c))
}
```

```
## [1] 1.000000 4.063372
## [1] 2.000000 7.544643
## [1] 3.00000 6.75764
## [1] 4.000000 3.892127
## [1] 5.000000 9.512311
## [1] 6.000000 7.555965
## [1] 7.000000 9.161686
## [1] 8.000000 7.189064
## [1] 9.000000 9.283873
## [1] 10.000000  7.975217
## [1] 11.000000  6.917562
## [1] 12.000000  7.034021
## [1] 13.000000  9.145282
## [1] 14.000000  6.623756
## [1] 15.000000  8.012406
## [1] 16.000000  4.222158
## [1] 17.000000  3.996034
## [1] 18.000000  9.164873
```

```
## [1] 19.00000  5.50721
## [1] 20.000000  3.726675
## [1] 21.000000  8.140939
## [1] 22.000000  5.637501
## [1] 23.000000  7.366889
## [1] 24.000000  7.439316
## [1] 25.000000  6.877135
## [1] 26.000000  6.556759
## [1] 27.000000  4.926457
## [1] 28.000000  7.433917
## [1] 29.000000  4.508086
## [1] 30.000000  6.045577
## [1] 31.000000  7.116586
## [1] 32.000000  6.568791
## [1] 33.000000  6.494069
## [1] 34.000000  6.768615
## [1] 35.0000  8.4767
## [1] 36.00000  9.60441
## [1] 37.000000  9.606253
## [1] 38.000000  5.355979
## [1] 39.000000  6.917013
## [1] 40.000000  9.530136
## [1] 41.000000  9.802424
## [1] 42.00000  3.89177
## [1] 43.000000  6.095849
## [1] 44.00000  9.09167
## [1] 45.000000  6.737204
## [1] 46.000000  9.621763
## [1] 47.000000  9.231489
## [1] 48.0000  6.4046
## [1] 49.000000  6.096076
## [1] 50.000000  8.962319
```

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

```r
table(x$id)
```

```
##
##  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
## 11 20 14 12 14 10  9 12 11 12 10 10  8 12  8  9 12 10 10  9 10  8  8 15 12
## 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
## 14 11 14 10  7 11  5  8 12 11  9 17 15  8  7 17 14 11 11 14  9 12 11 12 10
```

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

```r
x[which(x$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
## 1117 15 2002-04-25 06:23:05 8.012406     4
```

```
## 1154 15 2003-04-25 06:23:05 8.012406     5
## 484  15 2003-06-06 14:06:00 9.133769     6
## 1118 15 2004-06-06 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.

```r
url3 <- "https://raw.githubusercontent.com/fonnesbeck/Bios6301/master/datasets/addr.txt"
addr <- read.delim(url3, header=FALSE, stringsAsFactors=FALSE)
parsed.data <- data.frame(lastname=rep(NA,nrow(addr)),firstname=rep(NA,nrow(addr)),
                          streetno=rep(NA,nrow(addr)),streetname=rep(NA,nrow(addr)),
                          city=rep(NA,nrow(addr)),state=rep(NA,nrow(addr)),zip=rep(NA,nrow(addr)))

#Write function to trim leading and trailing whitespace to use after splitting strings into sections
trim <- function (x) gsub("^\\s+|\\s+$", "", x)

#Loop through addr file to parse into fields
for (i in 1:nrow(addr)){
  #Identify whitespace of 2 spaces or more
  cutpoints <- c(1,unlist(gregexpr(" {2,}", addr[i,])),nchar(addr[i,]))
  fields <- vector()
  #Loop through each line and cut into parts, then trim trailing and leading whitespace
  for (j in 1:(length(cutpoints)-1)){
    fields[j] <- substring(addr[i,],cutpoints[j],cutpoints[j+1])
    fields[j] <- trim(fields[j])
  }
  #Assign parts that don't need more splitting to the appropriate columns
  parsed.data[i,"lastname"] <- fields[1]
  parsed.data[i,"firstname"] <- fields[2]
  parsed.data[i,"city"] <- fields[4]
  parsed.data[i,"state"] <- fields[5]
  parsed.data[i,"zip"] <- fields[6]
  #Split street address into number and street name
  name.cut <- unlist(gregexpr("[[:alpha:]]", fields[3]))
  number <- substring(fields[3],1,name.cut[1]-1)
  parsed.data[i,"streetno"] <- trim(number)
  street <- substring(fields[3],name.cut[1],nchar(fields[3]))
  parsed.data[i,"streetname"] <- trim(street)
}
print(parsed.data)
```

```
##       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 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.

```r
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:

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

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

```r
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 function is failing because it cannot find the object 'death' in the function call. This occurs in the third line when we find the coefficients of the summary of the glm of death ~ . using `haart_df`. This is happening because the object "death" is defined inside of the data set `haart_df`, but we are trying to call the variable from outside of the data set itself. However, trying to call `haart_df$death` results in the model not properly using death as the outcome variable.

```r
myfun(haart_df, haart_df$death)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
##                  Estimate  Std. Error       z value  Pr(>|z|)
## (Intercept) -2.656607e+01 115935.1724 -2.291459e-04 0.9998172
## death        5.313214e+01  69028.4183  7.697140e-04 0.9993859
## weight      -4.499694e-15   1939.0571 -2.320558e-18 1.0000000
## hemoglobin   5.124642e-14   9774.8190  5.242697e-18 1.0000000
## cd4baseline  1.830771e-16    184.0846  9.945271e-19 1.0000000
```

5 bonus points

Create a working function.

```r
#We will use deparse(substitute(x)) to convert the inputs into strings, create the formula manually, an

myfun.2 <- function(dat, response) {
  c <- deparse(substitute(response))
  d <- deparse(substitute(dat))
  e <- paste(d,c,sep="$")
  f <- paste(e, " ~ .",sep="")
  print(f)
  print(coef(summary(glm(f, data=dat, family=binomial(logit)))))
}

myfun.2(haart_df,death)
```

```
## [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
```