Homework 5

Angela Huynh

11/15/2016

JC Grade: 51/50

Nice, job. Also, check out how Cole approached Question 2 using lapply and tapply.

Question 1:

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

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

```
# conversions of date columns to usable format
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")

# counts of year from init.date
table(year(haart$init.date))
```

```
##
## 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?

```
# 0 = death after 1 year of initial visit/not died within one year 1 = death
# within one year of initial visit

ind <- 1 * (haart$date.death <= haart$init.date + years(1))  # 1* makes it numeric
ind[is.na(ind)] <- 0  # turns NA's into 0s because patients did not die
sum(ind)  # sum of those who died within one year
```

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

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.

```
followup <- vector()
for (i in 1:length(haart$init.date)) {
    if (is.na(haart$date.death[i])) {
        followup[i] <- difftime(haart$last.visit[i], haart$init.date[i], units = "days")
    } else {
        followup[i] <- difftime(haart$date.death[i], haart$init.date[i], units = "days")
    }
}

# censored
for (i in 1:length(followup)) {
```

```
    if (followup[i] >= 365) {
        followup[i] <- 365
    }
}

quantile(followup)
```

```
##    0%   25%   50%   75%  100%
##   0.0 329.5 365.0 365.0 365.0
```

JC Grading -2 The 25th percentile is slightly high of 320.75. The death date can be recorded as being later than the last visit date, and if logic written above uses death date unless missing. Instead, find the minimum of the two dates to determine follow-up time.

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?

```
indicate <- vector()

for (i in 1:length(haart$init.date)) {
    if ((is.na(haart$last.visit[i])) == TRUE & (is.na(haart$last.visit[i])) ==
        TRUE) {
        indicate[i] <- 1
    } else {
        indicate[i] <- 0
    }
}

haart$loss.ind <- indicate  # column of indicators into dataset

(sum(indicate))  # records lost to followup
```

```
## [1] 11
```

JC Grading -2 The indicate variable created needs to check whether the patient was dead at last visit (you'd think they wouldn't show up if dead, but sometimes that's recorded as the last visit) and whether the last visit was within 365 days.

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"
```

```
(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"
```

```
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)  # columns of medications

# make TRUE values into medicine names
for (i in 1:nrow(haart_merged)) {
    for (j in 15:32) {
        if (haart_merged[i, j] == TRUE) {
            haart_merged[i, j] <- colnames(haart_merged)[j]
        } else {
            haart_merged[i, j] <- NA
        }
    }
}

# put medicine names into dataset
for (i in 1:nrow(haart_merged)) {
    true.meds <- vector()
    for (j in 15:32) {
        if (is.na(haart_merged[i, j]) == FALSE) {
            true.meds <- c(true.meds, haart_merged[i, j])
        }
    }
    haart_merged$true.meds[i] <- paste(true.meds, collapse = ",")
}

# Which regimens occur more than 100 times
((table(haart_merged$true.meds)[which(table(haart_merged$true.meds) > 100)]))
```

```
##
## 3TC,AZT,EFV 3TC,AZT,NVP
##         421         284
```

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.

```
# import dataset use all same commands as previous questions haart2 <-
# read.csv('~/Documents/Vanderbilt 1/Semester 1/BIOS
# 6301/Assignments/haart2.csv', header=TRUE)
haart2 <- read.csv("haart2.csv", header = TRUE)

# matching date formats
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")

# indicator column for death within one year of visit
indicate1 <- vector()

for (i in 1:length(haart2$init.date)) {
```

```
    if ((is.na(haart2$last.visit[i])) == TRUE & (is.na(haart2$last.visit[i])) ==
        TRUE) {
        indicate1[i] <- 1
    } else {
        indicate1[i] <- 0
    }
}

haart2$loss.ind <- indicate1  # column of indicators into dataset
haart2[, "init.reg.factor"] <- factor(haart2[, "init.reg"])

# indicator for loss of followup

indicate.loss <- vector()

for (i in 1:length(haart2$init.date)) {
    if ((is.na(haart2$last.visit[i])) == TRUE & (is.na(haart2$last.visit[i])) ==
        TRUE) {
        indicate.loss[i] <- 1
    } else {
        indicate.loss[i] <- 0
    }
}

haart2$loss.ind <- indicate.loss

# init.reg_list column

init.reg2 <- as.character(haart2[, "init.reg"])
haart2[["init.reg_list"]] <- strsplit(init.reg2, ",")

# merging the datasets
haart.all <- merge(x = haart, y = haart2, all = TRUE)

# show first 5 records
(head(x = haart.all, n = 5))
```

```
##   male age aids cd4baseline    logvl weight hemoglobin    init.reg
## 1    0  18    0          89 5.184231     NA         NA 3TC,AZT,EFV
## 2    0  18    0         280       NA 52.164         11 3TC,AZT,EFV
## 3    0  18    0         431 5.342423 58.000         NA 3TC,AZT,NVP
## 4    0  19    0          51 5.618615 48.600         NA 3TC,AZT,NVP
## 5    0  19    0         180 4.121330     NA         NA 3TC,AZT,NVP
##    init.date last.visit death date.death loss.ind init.reg_list
## 1 2003-11-03 2006-04-12     0       <NA>        0 3TC, AZT, EFV
## 2 2004-02-19 2008-03-14     0       <NA>        0 3TC, AZT, EFV
## 3 2007-03-13 2007-03-13     0       <NA>        0 3TC, AZT, NVP
## 4 2005-12-07 2007-04-17     0       <NA>        0 3TC, AZT, NVP
## 5 2006-09-08 2006-10-15     0       <NA>        0 3TC, AZT, NVP
##   init.reg.factor
## 1            <NA>
## 2            <NA>
## 3            <NA>
## 4            <NA>
```

```
## 5            <NA>
```

```
# show last 5 records
(tail(x = haart.all, n = 5))
```

```
##      male age aids cd4baseline   logvl  weight hemoglobin        init.reg
## 1000    1  66    0         298 4.09496      NA         NA     3TC,AZT,EFV
## 1001    1  67    0          95      NA 66.6792         16     3TC,AZT,EFV
## 1002    1  69    0          NA      NA      NA         NA 3TC,AZT,RTV,SQV
## 1003    1  80    0         267      NA 53.0712         NA     3TC,AZT,NVP
## 1004    1  89    0           9      NA 43.5456         10     3TC,ABC,AZT
##       init.date last.visit death date.death loss.ind      init.reg_list
## 1000 2006-06-08 2007-02-12     0       <NA>        0      3TC, AZT, EFV
## 1001 2004-02-13 2008-02-21     0       <NA>        0      3TC, AZT, EFV
## 1002 2006-04-01 2007-09-13     0       <NA>        0 3TC, AZT, RTV, SQV
## 1003 2004-11-08 2006-11-20     1 2006-11-26        0      3TC, AZT, NVP
## 1004 2004-12-15 2006-04-11     0       <NA>        0      3TC, ABC, AZT
##      init.reg.factor
## 1000            <NA>
## 1001            <NA>
## 1002            <NA>
## 1003            <NA>
## 1004            <NA>
```

Question 2

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:

1. Order the data set by id and dt.

```
attach(x)
new.x <- x[order(id, 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.

```
diff <- vector()  # vector of time differences

for (i in 2:nrow(new.x) + 1) {
    diff[i - 1] <- difftime(new.x$dt[i], new.x$dt[i - 1], units = "days")
    diff[1] <- 0
    diff[500] <- 0
    if (diff[i - 1] < 0) {
        diff[i - 1] <- 0
    }
}

diff.floor <- floor(diff/365)

new.x$diff <- diff.floor  # create column in new.x for floored time differences in years

for (i in 500:2) {
    if (new.x$diff[i] == 1) {
        # time difference is 1 year
        newrow <- data.frame(0, 0, 0, 0)
        colnames(newrow) <- c("id", "dt", "a1c", "diff")
        newrow[1, "dt"] <- as.character.Date(new.x[i, "dt"] + years(1))
        newrow[1, "id"] <- new.x[i, "id"]
        newrow[1, "a1c"] <- NA
        newrow[1, "diff"] <- 0
        new.x <- rbind(new.x[1:i, ], newrow, new.x[-(1:i), ])
    } else if (new.x$diff[i] == 2) {
        # time difference is 2 years
        newrow <- data.frame(0, 0, 0, 0)
        colnames(newrow) <- c("id", "dt", "a1c", "diff")
        newrow[1, "dt"] <- as.character.Date(new.x[i, "dt"] + years(1))
        newrow[1, "id"] <- new.x[i, "id"]
        newrow[1, "a1c"] <- NA
        newrow[1, "diff"] <- 0
        newrow[2, "dt"] <- as.character.Date(new.x[i, "dt"] + years(2))
        newrow[2, "id"] <- new.x[i, "id"]
        newrow[2, "a1c"] <- NA
        newrow[2, "diff"] <- 0
        new.x <- rbind(new.x[1:i, ], newrow, new.x[-(1:i), ])
    }
}
```

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.

```
new.x$visit <- ave(new.x$diff, new.x$id, FUN = seq_along)
```

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

```
# using mean a1c values from dataset x to replace missing values with
mean.a1c <- vector()
for (i in 1:50) {
    mean.a1c[i] <- mean(x$a1c[which(x$id == i)])
}
```

```
# replace missing values in new.x with individual averages from old dataset
for (i in 1:nrow(new.x)) {
    if (is.na(new.x$a1c[i])) {
        new.x$a1c[i] <- mean.a1c[new.x$id[i]]
    }
}
```

5. Print mean a1c for each id.

```
new.means <- vector()
for (i in 1:50) {
    new.means[i] <- mean(new.x$a1c[which(new.x$id == i)])
}
(new.means)  # mean a1c for each id
```

```
##  [1] 4.063372 7.544643 6.757640 3.892127 9.512311 7.555965 9.161686
##  [8] 7.189064 9.283873 7.975217 6.917562 7.034021 9.145282 6.623756
## [15] 8.012406 4.222158 3.996034 9.164873 5.507210 3.726675 8.140939
## [22] 5.637501 7.366889 7.439316 6.877135 6.556759 4.926457 7.433917
## [29] 4.508086 6.045577 7.116586 6.568791 6.494069 6.768615 8.476700
## [36] 9.604410 9.606253 5.355979 6.917013 9.530136 9.802424 3.891770
## [43] 6.095849 9.091670 6.737204 9.621763 9.231489 6.404600 6.096076
## [50] 8.962319
```

6. Print total number of visits for each id.

```
num.visits <- vector()

for (i in 1:50) {
    num.visits[i] <- length(which(new.x$id == i))
}
(num.visits)  # visits per id
```

```
##  [1] 11 20 14 12 14 10  9 12 11 12 10 10  8 12  8  9 12 10 10  9 10  8  8
## [24] 15 12 14 11 14 10  7 11  5  8 12 11  9 17 15  8  7 17 14 11 11 14  9
## [47] 12 11 12 10
```

7. Print the observations for id = 15.

```
(new.x[which(new.x$id == 15), ])
```

```
##      id                  dt      a1c diff visit
## 11   15 2000-04-30 00:34:50 7.527105    0     1
## 406  15 2001-01-17 21:11:02 5.898371    0     2
## 306  15 2001-04-25 06:23:05 8.566593    2     3
## 1137 15 2002-04-25 06:23:05 8.012406    0     4
## 2    15 2003-04-25 06:23:05 8.012406    0     5
## 484  15 2003-06-06 14:06:00 9.133769    1     6
## 1136 15 2004-06-06 14:06:00 8.012406    0     7
## 263  15 2004-08-20 17:47:11 8.936190    0     8
```

Question 3

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
data.addr <- "https://github.com/fonnesbeck/Bios6301/raw/master/datasets/addr.txt"
addr <- readLines(data.addr)
x <- gsub("(\\s){2,}", ",", addr)
# strplit(x, ',')

addr.split <- do.call(rbind, strsplit(x, ","))

last.name <- character(42)
first.name <- character(42)
street.num <- character(42)
street.name <- character(42)
city <- character(42)
state <- character(42)
zip <- character(42)

street.num.name <- grep("^[0-9].*[A-Z]", addr.split, value = TRUE)
street.no <- sub("([^0-9].*[^0-9])", "", street.num.name)
street.names <- sub("[^A-Z].*[0-9]", "", street.num.name)

for (i in 1:42) {
    last.name[i] <- addr.split[i, 1]
    first.name[i] <- addr.split[i, 2]
    street.num[i] <- street.no[i]
    street.name[i] <- street.names[i]
    city[i] <- addr.split[i, 4]
    state[i] <- addr.split[i, 5]
    zip[i] <- addr.split[i, 6]
}

(addr.data <- data.frame(last.name, first.name, street.num, street.name, city,
    state, zip))
```

```
##      last.name first.name street.num          street.name       city
## 1        Bania  Thomas M.        725    Commonwealth Ave.     Boston
## 2      Barnaby      David        373        W. Geneva St.   Wms. Bay
## 3       Bausch       Judy        373        W. Geneva St.   Wms. Bay
## 4      Bolatto    Alberto        725    Commonwealth Ave.     Boston
## 5    Carlstrom       John        933                th St.    Chicago
## 6   Chamberlin Richard A.        111           Nowelo St.       Hilo
## 7        Chuss       Dave       2145          Sheridan Rd   Evanston
## 8        Davis      E. J.        933                th St.    Chicago
## 9        Depoy     Darren        174               th Ave.   Columbus
## 10     Griffin       Greg       5000          Forbes Ave. Pittsburgh
## 11   Halvorsen       Nils        933                th St.    Chicago
## 12      Harper         Al        373        W. Geneva St.   Wms. Bay
## 13       Huang     Maohai        725  W. Commonwealth Ave.     Boston
## 14     Ingalls   James G.        725  W. Commonwealth Ave.     Boston
## 15     Jackson   James M.        725  W. Commonwealth Ave.     Boston
## 16     Knudsen      Scott        373        W. Geneva St.   Wms. Bay
## 17       Kovac       John       5640        S. Ellis Ave.    Chicago
## 18   Landsberg      Randy       5640        S. Ellis Ave.    Chicago
## 19          Lo  Kwok-Yung       1002         W. Green St.     Urbana
## 20 Loewenstein  Robert F.        373        W. Geneva St.   Wms. Bay
## 21       Lynch       John       4201          Wilson Blvd  Arlington
```

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

```
## 33    IL 60208-3112
## 34    CO       80125
## 35    IL       60637
## 36    WI       53191
## 37    WI       53191
## 38    AZ       85721
## 39    PA       15213
## 40    WI       53191
## 41    NY 07733-1988
## 42    IL       60637
```

Question 4

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 variable "death" isn't attached to anything. You can't call out a column without giving context with the dataset.

Bonus Create a working function.

```
myfun <- function(dat, response) {
  #attach(dat, warn.conflicts = FALSE)
  response.name <- deparse(substitute(response))
  df.name <- deparse(substitute(dat))
  response.df <- paste(df.name, response.name, sep="$")
  form <- paste(response.df, " ~ .", sep = "")
  #form <- as.formula(response ~ .)
  #coef(summary(glm(form, data=dat, family=binomial(logit))))
  print(coef(summary(glm(form, data=dat, family=binomial(logit)))))
  #detach(dat)
}
```

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

```
##                 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
```

JC Grading +5