#Script to calculate and display a temporal phenome map using MIMIC2 data

#

#Author: Jeremy L. Warner M.D., M.S.

# Assistant Professor of Medicine and Biomedical Informatics

# jeremy.warner@vanderbilt.edu

#

#Version date: 07/12/2013

#

#Disclaimer: This script is open-source and you are free to distribute, modify, and

# use without limitations. The original author is not responsible for any errors

# associated with the script and offers no guarantees as to its functionality.

# Attribution is appreciated.

#

#Note: This is a script file and is not functionalized. Individual sections

# are delineated by headers.

#RPostgreSQL library for SQL database interface with MIMIC2

library(RPostgreSQL)

drv <- dbDriver("PostgreSQL")

#Password is in clear text; you must have MIMIC2 access.

conn <- dbConnect(drv, dbname="MIMIC2", user="mimic2", password="XXXXXX")

#1 Run algorithm to determine length of hospital stay for all adult patients

#1.1 All adult admissions with admit and discharge dates

all.hadm <- dbGetQuery(conn, "SELECT distinct admissions.subject\_id as subject\_id, hadm\_id, admit\_dt, disch\_dt

 FROM mimic2v26.admissions

 WHERE admissions.subject\_id in

 (SELECT subject\_id

 FROM mimic2v26.icustay\_detail

 WHERE icustay\_age\_group = 'adult')")

hadm <- paste(all.hadm$hadm\_id, ',', sep='', collapse=' ')

hadm <- substr(hadm, start = 1, stop = nchar(hadm) - 1)

#1.1.1 All adult admissions with admission and discharge dates and at least one POE order

all.hadm.poe <- dbGetQuery(conn, "

 SELECT hadm\_id, min(enter\_dt) as first\_poe, max(stop\_dt) as last\_poe

 FROM mimic2v26.admissions JOIN mimic2v26.poe\_order using(hadm\_id)

 WHERE admissions.subject\_id in

 (SELECT subject\_id

 FROM mimic2v26.icustay\_detail

 WHERE icustay\_age\_group = 'adult')

 GROUP BY admissions.subject\_id, hadm\_id, admit\_dt, disch\_dt")

#1.1.2 All adult admissions with at least one lab order within t -48 hours of admission

all.hadm.lab <- dbGetQuery(conn, paste("

 SELECT admissions.hadm\_id as hadm\_id, min(charttime) as first\_lab

 FROM mimic2v26.labevents JOIN mimic2v26.admissions using(subject\_id)

 WHERE admissions.hadm\_id in (", hadm, ") AND

 charttime - admit\_dt > interval '-48 hours'

 GROUP BY admissions.hadm\_id"))

#1.1.3 All adult discharges from the ICU

all.hadm.icu <- dbGetQuery(conn, "SELECT subject\_id, hadm\_id, icustay\_outtime as icu\_out

 FROM mimic2v26.icustay\_detail

 WHERE icustay\_age\_group = 'adult'")

#1.2 The following code determines the first event for each hadm\_id

all.hadm <- merge(all.hadm, all.hadm.lab, by='hadm\_id')

all.hadm <- merge(all.hadm, all.hadm.poe, by='hadm\_id', all.x = TRUE)

all.hadm$first.event <- as.POSIXct(NA)

all.hadm$first.event[difftime(all.hadm$first\_lab, all.hadm$admit\_dt, units = 'h') <= 24] <-

 all.hadm$first\_lab[difftime(all.hadm$first\_lab, all.hadm$admit\_dt, units = 'h') <= 24]

foo <- which(is.na(all.hadm$first.event))

#1.2.1 Some hadm have enter dates more than 24 hours after admit\_dt; fall back to admit\_dt/disch\_dt

all.hadm$first.event[foo] <- all.hadm$first\_poe[foo]

all.hadm$first.event[foo][difftime(all.hadm$first.event[foo], all.hadm$admit\_dt[foo], units = 'h') > 24] <- NA

all.hadm$first.event[is.na(all.hadm$first.event)] <- all.hadm$admit\_dt[is.na(all.hadm$first.event)]

#1.2.2 For patients who went from hospital to ICU and then discharged in the same day, use ICU discharge as out

#and first POE order as in.

all.temp <- merge(all.hadm.icu, all.hadm, by='subject\_id', all.x = TRUE)

all.temp$diff <- difftime(all.temp$icu\_out, all.temp$disch\_dt, units = 'days')

all.temp <- all.temp[which(all.temp$diff >= 0),]

all.temp <- all.temp[which(all.temp$diff <= 1),]

all.temp$diff <- difftime(all.temp$icu\_out, all.temp$first.event, units = 'hours')

all.temp <- all.temp[order(all.temp$hadm\_id.y),]

#1.3 The following code determines the last event for each hadm\_id

all.hadm$last.event <- as.POSIXct(NA)

#1.3.1 Adds the times to the all.hadm table

all.hadm$last.event[which(all.hadm$hadm\_id %in% all.temp$hadm\_id.y)] = all.temp$icu\_out

#1.3.2 Some hadm have stop dates less than 24 hours after disch\_dt

foo <- which(is.na(all.hadm$last.event))

#1.3.3 Some hadm have enter dates more than 24 hours after admit\_dt; fall back to admit\_dt/disch\_dt

all.hadm$last.event[foo] <- all.hadm$last\_poe[foo]

all.hadm$last.event[foo][difftime(all.hadm$last.event[foo], all.hadm$disch\_dt[foo], units = 'h') > 24] <- NA

#1.3.4 Some have labs less than 24 hours after disch\_dt

hadm <- all.hadm$hadm\_id[is.na(all.hadm$last.event)]

hadm <- paste(hadm, ',', sep='', collapse=' ')

hadm <- substr(hadm, start = 1, stop = nchar(hadm) - 1)

last.lab <- dbGetQuery(conn, paste("

 SELECT admissions.hadm\_id as hadm\_id, max(charttime) as last\_lab

 FROM mimic2v26.labevents JOIN mimic2v26.admissions using(subject\_id)

 WHERE admissions.hadm\_id in (", hadm, ") AND

 charttime - disch\_dt <= interval '24 hours' AND

 charttime - disch\_dt > interval '0 hours'

 GROUP BY admissions.hadm\_id"))

last.lab <- last.lab[order(last.lab$hadm\_id),]

all.hadm$last.event[which(all.hadm$hadm\_id %in% last.lab$hadm\_id)] = last.lab$last\_lab

#1.3.5 Some will just default to the disch\_dt

all.hadm$last.event[is.na(all.hadm$last.event)] <- all.hadm$disch\_dt[is.na(all.hadm$last.event)]

#1.4 Calculate the difference, in hours, between discharge and admit times

all.hadm$diff <- difftime(all.hadm$last.event,all.hadm$first.event, units = 'hours')

all.hadm$diff <- as.numeric(all.hadm$diff)

#1.4.1 If the difference is a negative number, truncate to 0 hours

all.hadm$diff[all.hadm$diff < 0] <- 0

#2 Specify bin size and determine interval cutoffs and subpopulations

#2.1 Determine the optimal number of bins with goal ~n patients per bin

#Example: n = 100

n <- 100

bin.n <- ceiling(dim(all.hadm)[1]/n)

#2.2 Prepare to create cutoffs

t.cuts <- quantile(as.numeric(all.hadm$diff), probs=seq(from=0, to=1, length.out=bin.n))

#2.3 Calculate subpopulations based on intervals

sub.hadm <- as.list(rep(0,(length(t.cuts)-1)))

for (a in 1:(length(t.cuts)-1)) {

 hadm <- all.hadm$hadm\_id[which(all.hadm$diff > t.cuts[a] & all.hadm$diff <= t.cuts[a+1])]

 hadm.n <- length(hadm)

 hadm <- paste(hadm, ',', sep='', collapse=' ')

 hadm <- substr(hadm, start = 1, stop = nchar(hadm) - 1)

 sub.hadm[[a]][1] <- hadm.n

 sub.hadm[[a]][2] <- hadm

 }

#3 Obtain ICD-9 counts for the subpopulations

#3.1 Initialize table with all possible ICD-9 in MIMIC

icd9.table <- dbGetQuery(conn, "SELECT distinct code FROM mimic2v26.icd9")

for (a in 1:(length(t.cuts)-1))

 {

 temp.colname <- paste("t",a,sep="\_")

 temp.icd9 <- dbGetQuery(conn, paste(

 "SELECT code, count(\*) AS", temp.colname,

 "FROM mimic2v26.icd9",

 "WHERE hadm\_id in (",

 sub.hadm[[a]][2],

 ") GROUP BY code"))

 icd9.table <- merge(icd9.table,temp.icd9,all.x=T)

 }

#3.2 Replace all NA created in this process with 0's

icd9.table[is.na(icd9.table)] <- 0

#4 Determine p-values for the developed table with Fisher's Exact Test

p.table <- data.frame(matrix(nrow=dim(icd9.table)[1],ncol=dim(icd9.table)[2]),stringsAsFactors=F)

colnames(p.table) <- colnames(icd9.table)

p.table[,1] <- icd9.table[,1]

icd9.table.r <- rowSums(icd9.table[,2:dim(icd9.table)[2]])

for (j in 2:dim(p.table)[2])

 {

 # Number in subgroup, rounded up

 n3 <- as.integer(sub.hadm[[j-1]][[1]])

 if(n3 > 0) {

 temp <- rep(0,dim(p.table)[1])

 for (i in 1:dim(p.table)[1])

 temp[i] <- fisher.test(matrix(c(icd9.table[i,j],

 icd9.table.r[i]-icd9.table[i,j],

 n3,

 dim(all.hadm)[1]-n3),

 nrow = 2, ncol = 2))[['p.value']]

 p.table[,j] <- temp } else p.table[,j] <- 1

 }

#4.1 Replace p=0 with p=1.5e-322 (smallest size reported in R)

p.table[p.table==0] <- 1.5e-322

#4.2 Adjust P-value for FDR on each bin individually

p.table.adj <- p.table

for (i in 2:dim(p.table)[2])

 p.table.adj[,i] <- p.adjust(p.table[,i], method='BH')

#5 Data preparation for visualization

#5.1 Create numeric correlates to V-codes and E-codes

code <- data.frame(codec=as.character(icd9.table$code),

 coden=as.character(icd9.table$code),stringsAsFactors=F)

temp.index <- grep("V", code$codec)

v.rep <- code$codec[temp.index]

code$coden[temp.index] <- as.numeric(sub('V','',v.rep))+1050

rm(v.rep)

temp.index <- grep("E", code$codec)

e.rep <- code$codec[temp.index]

code$coden[temp.index] <- as.numeric(sub('E','',e.rep))+400

rm(e.rep)

code$coden <- as.numeric(code$coden)

#5.2 Create an ICD-9 chapter color mapping table

icd9.col <- data.frame(cbind(

 lower=c(0,140,240,280,290,320,360,390,460,520,580,630,680,710,740,760,780,

 800,1025,1175),

 color=c('magenta','cyan','blue','green3','black','orange',

 'magenta','cyan','blue','green3','black','orange',

 'magenta','cyan','blue','green3','black','orange',

 'purple','gray')

 ),

 stringsAsFactors = FALSE)

icd9.col$lower <- as.numeric(icd9.col$lower)

#5.3 Create data frames containing localization, point sizing, and color information

#5.3.1 First data frame, all adjusted p not equal to 1

temp.df1 <- data.frame(codec=NA,coden=NA,y1=NA,y2=NA,p=NA)

for (i in 2:dim(p.table.adj)[2]) {

 temp <- which(p.table.adj[,i] != 1)

 if (length(temp) > 0)

 for (j in 1:length(temp))

 temp.df1 <- rbind(temp.df1,

 cbind(code[temp[j],],y1=i-1,y2=i,p=p.table.adj[temp[j],i]))

 }

temp.df1 <- na.omit(temp.df1)

temp.df1 <- temp.df1[temp.df1$p < 0.99,]

temp.df1 <- temp.df1[order(temp.df1$coden),]

temp.df1$t1 <- round(t.cuts[temp.df1$y1]/24,digits=1)

temp.df1$t2 <- round(t.cuts[temp.df1$y2]/24,digits=1)

temp.df1$color <- ''

for (j in 1:dim(temp.df1)[1])

 temp.df1$color[j] <- icd9.col$color[max(which(icd9.col$lower <= temp.df1$coden[j]))]

#5.3.2 Second data frame, only adjusted p <= 0.05

temp.df2 <- temp.df1[temp.df1$p <= 0.05,]

#6 Temporal phenome maps based on temp.df1 and temp.df2

# If you wish to generate a TIFF, run the entire section.

# If you wish to view the graphic within the R environment, exclude the tiff() and dev.off()

# If you wish to "zoom" into parts of the map, change the x.min, x.max, y.min, y.max parameters

#6.1 Set the borders of the map

#6.1.1 Default x-axis values range from 0 to 1400

x.min <- 0

x.max <- 1400

#6.1.2 Default y-axis values range from 1st to last time interval

y.min <- 0

y.max <- dim(p.table)[2]

#6.2 Open pipe to TIFF

tiff(filename = paste('temporal.phenome.',n,'.',Sys.Date(),'.tif',sep=''),

 width=10, height=8, res=300, units='in', compression='lzw')

#6.3 Set the environment to show two graphs in one TIFF/window

par(mfrow=c(2,1))

#6.4 Create the first temporal phenome map

#6.4.1 Create temporal phenome map borders and labels, first

plot(x=-100,

 xlim=c(x.min, x.max),

 xaxt='n',

 ylim=c(y.min, y.max),

 yaxt='n',

 xlab="ICD-9-CM Code",

 ylab="Time, days")

#6.4.2 Add line segments

for (i in 1:dim(temp.df2)[1])

 segments(x0=temp.df2$coden[i],

 y0=temp.df2$y1[i],

 y1=temp.df2$y2[i]+1,

 lwd=3\*log(-log10(temp.df2$p[i])),

 col=temp.df2$color[i])

#6.4.3 Place tick marks on the x-axis

axis(side=1,

 labels=c('V Codes','E Codes'),

 at=c(1100,1300)

 )

#6.4.4 Place tick marks on the y-axis

axis(side=2,

 labels=round(quantile(x=t.cuts, probs = seq(0, 1, 0.1))/24),

 at=quantile(x=1:length(t.cuts), probs = seq(0, 1, 0.1)))

#6.4.5 Next, add title (default title is "A")

mtext(text = "A", font=2, side = 3, adj = 0, padj = -1, cex=1.5)

#6.4.6 Add light gray dashed vertical lines to divide by ICD-9 chapter

for (i in c(140,240,280,290,320,360,390,460,520,580,630,680,710,740,760,780,

 800,1025,1175))

 abline(v = i, lty = 2, col = "light gray")

#6.4.7 Adds horizontal lines at 25th, median, 75th percentile

# Default text placement will have to be changed if borders are changed

foo <- quantile(1:length(t.cuts), probs=seq(from=0.25, to=.75, length.out=3))

abline(h = foo[1], lty = 2, col = "black")

text(1200, foo[1], "25th percentile", pos = 3)

abline(h = foo[2], lty = 2, col = "black")

text(1200, foo[2], "50th percentile", pos = 3)

abline(h = foo[3], lty = 2, col = "black")

text(1200, foo[3], "75th percentile", pos = 3)

#6.5 Create the second temporal phenome map

#6.5.1 Create temporal phenome map borders and labels, first

plot(x=-100,

 xlim=c(x.min, x.max),

 xaxt='n',

 ylim=c(y.min, y.max),

 yaxt='n',

 xlab="ICD-9-CM Code",

 ylab="Time, days")

#6.5.2 Add line segments

for (i in 1:dim(temp.df1)[1])

 segments(x0=temp.df1$coden[i],

 y0=temp.df1$y1[i],

 y1=temp.df1$y2[i]+1,

 lwd=3\*log(-log10(temp.df1$p[i])),

 col=temp.df1$color[i])

#6.5.3 Place tick marks on the x-axis

axis(side=1,

 labels=c('V Codes','E Codes'),

 at=c(1100,1300)

 )

#6.5.4 Place tick marks on the y-axis

axis(side=2,

 labels=round(quantile(x=t.cuts, probs = seq(0, 1, 0.1))/24),

 at=quantile(x=1:length(t.cuts), probs = seq(0, 1, 0.1)))

#6.5.5 Next, add title (default title is "B")

mtext(text = "B", font=2, side = 3, adj = 0, padj = -1, cex=1.5)

#6.5.6 Add light gray dashed vertical lines to divide by ICD-9 chapter

for (i in c(140,240,280,290,320,360,390,460,520,580,630,680,710,740,760,780,

 800,1025,1175))

 abline(v = i, lty = 2, col = "light gray")

#6.5.7 Adds horizontal lines at 25th, median, 75th percentile

# Default text placement will have to be changed if borders are changed

foo <- quantile(1:length(t.cuts), probs=seq(from=0.25, to=.75, length.out=3))

abline(h = foo[1], lty = 2, col = "black")

text(1200, foo[1], "25th percentile", pos = 3)

abline(h = foo[2], lty = 2, col = "black")

text(1200, foo[2], "50th percentile", pos = 3)

abline(h = foo[3], lty = 2, col = "black")

text(1200, foo[3], "75th percentile", pos = 3)

#7 Output pertinent data into a RData file with a specified naming convention:

# temporal.phenome.[bin size].[date of creation].RData

save(sub.hadm,all.hadm,icd9.table,p.table,

 file=paste('temporal.phenome.',n,'.',Sys.Date(),'.RData',sep=''))