Midterm

You midterm project involves building and analyzing datasets from the internet movie database (http://www.imdb.com).

1. The page http://www.imdb.com/chart/top?ref_=ft_250 has the top 250 movies as rated by IMDB users. Our first goal is to build a dataframe that has one row for each of these movies. The columns should be Title (character), Year (numeric), Content.rating (character), User.rating (numeric), Num.Raters (numeric), US Gross ($US) (numeric), Budget ($US) (numeric), US Opening ($US) (numeric), and Genre (character). For instance, the current top movie's webpage is http://www.imdb.com/title/tt0111161/ and information about its finances is at http://www.imdb.com/title/tt0111161/business. The resulting row for the data frame would include

Title: The Shawshank Redemption Year: 1994 Content.Rating: R User.Rating: 9.3 Num.Raters: 1,538,647 Gross: 28,341,469 Budget: 25,000,000 Opening: 727,327 Genre: Crime, Drama.

(Code below will help you get started.)

a. IMDB movie pages have addresses that start with "www.imdb.com/title/" and are followed by "tt" and a 7-digit number ("tt codes"). (The page about finances is the same address, followed by "/business".) Our first task is to get those "tt codes" for the top 250 movies. Some R commands to do that are below. Please add a comment before each line to explain what the line does.

```r
top250.page <- readLines("http://www.imdb.com/chart/top?ref_=ft_250")
top250.codes <- top250.page[grep("/title/tt", top250.page)]
top250.codes <- unlist(strsplit(top250.codes, split = "//?"))[seq(from = 3,
    to = 1999, by = 4)]
top250.codes <- top250.codes[seq(from = 2, to = 500, by = 2)]
head(top250.codes)
```

```
## [1] "tt0111161" "tt0068646" "tt0071562" "tt0468569" "tt0050083" "tt0108052"
```

b. Now that we have the "tt codes," our next task is to write a function that reads a movie's main page and business page and extracts the information we need. Below is code to write an example of the main page html and the business page html to your computer. It may be helpful to look at parts of those pages in a text editor to do the rest of this problem.

```r
address <- "http://www.imdb.com/title/tt0245429/"
page <- readLines(address)
# you may have to change the path below
write(page, file = "~/example.main.txt")

address <- "http://www.imdb.com/title/tt0245429/business"
page <- readLines(address)
# you may have to change the path below
write(page, file = "~/example.finance.txt")
```

Below is partial code for a function to read the pages and extract the information we want. Your job is to finish the code and return a list with the nine pieces of information above (Title, Year, etc). SUGGESTIONs are in the comments in the code. All this can be done with grep() and strsplit(). gsub() may make things easier too. That said, I am sure there are more efficient ways. If you see a better way, that is great!

```r
get.stuff <- function(imdb.code) {
    # Make the movie's main page address
    address <- paste("http://www.imdb.com/title/", imdb.code, sep = "")
    # Read the .html for the page
    page <- readLines(address)

    # extract the line that has the movie title and the year
    title.year <- page[grep("<title>", page)]
    # SUGGESTION: use a series of strsplit() commands to remove the extraneous
    # characters until you can put the title into a variable called Title and
    # year into a variable called Year.

    # get content rating
    Content.Rating <- page[grep("contentRating", page)[1]]
    # SUGGESTION: use strsplit() to get rid of extaneous text

    # get user ratings and number of raters
    User.Rating <- page[grep("Users rated this", page)[1]]
    User.Rating <- strsplit(User.Rating, "/")[[1]][1]
    User.Rating <- as.numeric(strsplit(User.Rating, "this ")[[1]][2])

    # SUGGESTION: use similar commands to above to get Num.Raters

    # Get genre information
    inds <- grep("ref_=tt_stry_gnr", page)
    Genre <- unlist(strsplit(page[inds], "/genre/"))[seq(from = 2, to = length(inds) *
        2, by = 2)]
    Genre <- unlist(strsplit(Genre, "\\?"))[seq(from = 1, to = length(inds) *
        2, by = 2)]
    Genre <- paste(Genre, collapse = " ")

    # Make the movie's finance page address
    address <- paste("http://www.imdb.com/title/", imdb.code, "/business", sep = "")
    business.page <- readLines(address)

    budget <- NA
    opening <- NA
    gross <- NA
    temp <- grep("Budget", business.page)
    if (length(temp) > 0) {
        budget <- business.page[temp + 1]
        budget <- strsplit(budget, " \\(")[[1]][1]
        budget <- gsub("\\$", "", budget)
        budget <- as.numeric(gsub("\\,", "", budget))
    }

    # SUGGESTION: repeat for the gross revenue and the opening

    return(list(Title = Title, Year = Year, Content.Rating = Content.Rating,
        User.Rating = User.Rating, Num.Naters = Num.Raters, Genre = Genre, Budget = budget,
        Opening = opening, Gross = gross))
}
```

Test your function on one movie. You should get something similar to what is below.

as.vector(get.stuff("tt0407887")) $Title [1] "The Departed" $Year [1] 2006 $Content.Rating [1] "R" $User.Rating [1] 8.5 $Num.Raters [1] "786280" $Genre [1] "Crime Drama Thriller" $Budget [1] 9e+07 $Opening [1] 26887467 $Gross [1] 132384315

c. Our next task is to apply the function to all 250 movies. One way to do that is to apply the function to each element of top250.codes with a loop. A better way is to use a function from the **apply** family. We will learn more about those functions later, but code is below.

```
  # The next command takes a few minutes.
  data <- sapply(top250.codes,get.stuff)
  # Please explain what the next line does.
  data <- as.data.frame(t(data))
  # sapply() leave each column of data in an odd format (a vector of lists!)
  # The next line fixes that.
  for (i in 1:dim(data)[2])
      data[,i] <- unlist(data[,i])
```

2. Now that we have the dataset, let's do some descriptive analyses. Pleast note that if you could not finish 1 completely, you may download the data from the course website to do this question.

a. Write a function to list the names and revenues of the **x** movies with the largest grosses. Evaluate your function at **x=5
b. Make a plot that shows mean gross revenue for each content rating category. Please make sure to appropriately label your plot and organize the data to make it easy to read. Please label with the number of movies in each category too.
c. Is there a strong relationship between user ratings (or number of user ratings) and gross revenue? Support your answer.
d. Use a t-test to compare the mean gross revenue for movies whose genre includes Drama to those whose genre includes Comedy.
e. Please compare gross revenue and opening revenue. (Gross means total revenue, and opening means opening weekend revenue.) Compare gross revenue and opening revenue. Do you see any problems? For which movies? Why?

3. The main page for each movie lists six other movies that were also liked by users. These "People who liked this also liked…" links can be used to define a correspondence network among the 250 top movies. The code below finds the "tt codes" for the movies that are linked to a movie's main page. (I'll call these linked movies "recommendations" below.)

```
linked.movies <- function(imdb.code) {
    # read the movie page
    address <- paste("http://www.imdb.com/title/", imdb.code, sep = "")
    page <- readLines(address)

    # find the lines with the recommendations and strip the unneeded stuff
    recs <- page[grep("rec_item", page)]
    recs <- unlist(strsplit(recs, "data-tconst="))[seq(from = 2, to = 24, by = 2)]

    # return the codes
    recs <- paste("tt", gsub("[^0-9]", "", recs), sep = "")
    return(recs)
}
linked.movies("tt0245429")
```

```
##  [1] "tt0347149" "tt0096283" "tt0092067" "tt0110357" "tt0910970"
##  [6] "tt0876563" "tt0097814" "tt1049413" "tt0435761" "tt0892769"
## [11] "tt1568921" "tt0266543"
```

a. Your next task is to write a function to build a matrix with 250 rows and 250 columns where each row and column corresonds to one of the top 250 movies, and label each with the movie's name. The (i, j) the entry in the matrix should be 1 if movie i recommends movie j and 0 otherwise. Once you have the matix, use the code below to plot the corresponding graph. Each movie is a node and two nodes are connected by an edge if one movie recommends the other.

b. (If you want to show off, fill in this matrix without a loop!)

c. One way to extend the functio you built in (a) is to include another input, number of recommendations. This input takes values between 1 and 6. If it is 1, the link is only included if the movie is the top recommendation, etc.

d. Which movies are linked to the most other movies?

e. Finally, you can plot your network with the following code. (You will need the statnet library to run this.)

```
library(statnet)
  # links is the matrix with the links in it.
  g <- network(links)
  plot(g)
```