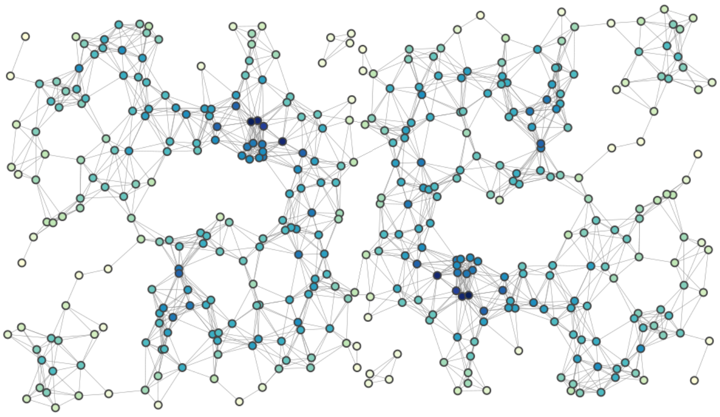

Old Dominion University

CS 432 Web Science

Assignment Seven

Derek Goddeau

Professor

Michael L. Nelson

April 7, 2017

1 Find a substitute you

Because of the structure of the data I chose to do the entire assignment in R. It is much easier to deal with so many data frames with R as long as there is no need for anything other than data manipulation. After reading each dataset into a data frame with the same name I took a subset of u.user for users similar to me.

```
df <- u.user[ u.user$age == 29
             & u.user$gender == 'M'
             & u.user$occupation == 'programmer', ]
```

This only resulted in two users.

	user.id	age	gender	occupation	zip.code
45	45	29	M	programmer	50233
222	222	29	M	programmer	27502

But another try with `u.user$occupation == 'scientist'` got one more hit.

	user.id	age	gender	occupation	zip.code
483	483	29	M	scientist	43212

Then for each of the three resulting users I created a list of all of their ratings by subsetting the `u.data` data frame and then matching each item ID to each movie ID and keeping the movie names.

```
u45.data <- u.data[ u.data$user.id == 45, ][c("item.id", "rating")]
u45.data$item.id <- u.item$movie.title[match(u45.data$item.id,
                                              u.item$movie.id
                                             )
                                       ]
```

This resulted in a data frame as like the following table.

item.id	rating
Birdcage, The (1996)	4
Mystery Science Theater 3000: The Movie (1996)	5
Twister (1996)	4
Dragonheart (1996)	3
Godfather, The (1972)	5
Independence Day (ID4) (1996)	4

From here I counted each good rate and bad rate for each of the three users. I generated a score for each by subtracting the total number of bad rates from the total number of good rates. None had a good score, but User 45 and User 222 were close and User 222 got disqualified by giving The Fifth Element a bad rating. The chosen substitute me, User 45, ended up with a score of -3 with three more ratings I considered bad than good. So not very representative, the biggest outliers were rating Twister a 4 and Willy Wonka and the Chocolate Factory 2.

2 Find top and bottom five correlated users to you

To find correlations in the data I used the `cor()` function throughout. First I gathered a list of data frames for each user's ratings.

```
user.list <- list()
for(n in 1:dim(u.user)) {
        user.list[[n]] <- u.data[ u.data$user.id == n, ][c("item.id", "rating")]
}
```

Then I define the `cordf` function which correlates two dataframes passed to it and use `sapply()` over the list to compare each user to the subsitute me.

```
cordf <- function(df.one, df.two) {
    df.one <- df.one[ df.one$item.id %in% df.two$item.id, ]
    df.two <- df.two[ df.two$item.id %in% df.one$item.id, ]
    df.one <- df.one[order(df.one[,1]), ]
    df.two <- df.two[order(df.two[,1]), ]

    # pearson is the default
    cor(df.one$rating, df.two$rating)
}

cors <- list()
cors <- sapply(user.list, cordf, df.one=sub.me)
```

I use the same exact pattern to create a vector of the number of items each correlation calculation was performed with and merge them into a single data frame, and removing substitute me from the data. The `cor()` function requires at least three data points to calculate the correlation but there was a large number of ties for most and least correlated, and with only three data points it is possible that some were lucky collisions. I filtered the data frame removing all users that did not have more than 5 movie ratings in common with substitute me.

```
cor.data.filtered <- cor.data[cor.data$incommon > 5, ]
```

From here the top five correlated users can easily be found using `order()`.

```
head(cor.data.filtered[ order(cor.data.filtered[,1], decreasing=TRUE), ])
```

	correlation	incommon
728	1.0000000	6
210	0.9338430	12
871	0.9284767	6
409	0.8931977	6
71	0.8876254	7
610	0.8750000	6

Also, the least five correlated.

```
head(cor.data.filtered[ order(cor.data.filtered[,1], decreasing=FALSE), ])
```

	correlation	incommon
647	-0.8783101	6
196	-0.8212037	8
217	-0.7083333	7
677	-0.5519432	8
199	-0.5510141	6
636	-0.5477226	7

3 Get your top and bottom 5 recommendations

To get the top and bottom five recommendations I start by creating a correlation matrix for all of the movies.

```r
# Slice only the genre data
item.data <- u.item[c(-1:-5)]
rownames <- rownames(item.data)

# Invert and replace rownames
t.item.data <- as.data.frame(t(item.data))
colnames(t.item.data) <- rownames
item.cors <- cor(t.item.data)
```

This results in a symmetric matrix so it is true that,

$$\begin{aligned} A &= A^{\mathrm{T}} \\ A_{ij} &= A_{ji} \end{aligned}$$

I then create 6 helper functions which consist of `top.5.items`, `top.5.users`, `top.5.user.items`, and their bottom counterparts. each one returns only a numeric vector of item or user IDs.

The helper functions are used in the `get.ratings` function to return a data frame with item IDs and a score. To calculate the score first the top five most correlated users to substitue me are found using the `top.5.users` function and each of their top five movies are found. Then I get the five most correlated movies to each of their favorite movies. The item IDs for each is added to vector including duplicates.

```
# Top 5's favorites
top.users <- top.5.user()
item.id <- unlist(lapply(top.users, top.5.user.items, target.id=u.id))

# Get movies similar to my top 5 users top 5
item.id <- append(item.id,
                  unlist(lapply(unique(item.id), top.5.items,
                                u.id=45,
                                i.cors=item.cors
                               )
                        )
                 )
```

Then I get my top five but only add items similar to the vector. Last the items in the vector are counted and the results returned as a data frame using `as.data.frame(table(item.id))`.

4 Get films related to favorite and least favorite

My favorite movie on the list is *The Fifth Elemet* and the most correlated movies to it are:

correlation	title
1.00000000	Timecop
1.00000000	No Escape
1.00000000	Highlander III
1.00000000	Barb Wire
1.00000000	Demolition Man

I would say the only one totally out of place is *Barb Wire.* The bottom 5 are:

correlation	title
-0.204980	Space Jam
-0.204980	Hercules
-0.177123	Legends of the Fall
-0.177123	Professional, The
-0.177123	Bound

Besides *Space Jam* this is pretty accurate.

My least favorite film from the list is *Mystery Science Theater 3000*:

correlation	title
1.00000000	Sleeper
1.00000000	Delicatessen
1.00000000	Back to the Future
1.00000000	Coneheads
1.00000000	Junior

I have no idea about *Sleeper*, *Delicatessen*, or *Junior*, and *Back to the Future* is definetely not similar. *Coneheads* is spot on but that shows how this isn't always a good metric because I do like that movie.

correlation	title
-0.204980	Diva
-0.204980	Pagemaster, The
-0.177123	Legends of the Fall
-0.177123	Professional, The
-0.177123	Bound

I have no idea about *Diva* or *The Pagemaster*, but they do not sound like they are similar.