

# IMDB\_network\_final

```
devtools::install_github("briatte/ggnet")
```

```
## Skipping install of 'ggnet' from a github remote, the SHA1 (da9a7cf2) has not changed since last ins
##   Use `force = TRUE` to force installation
```

```
library(ggnet)
```

```
##
## Attaching package: 'ggnet'
```

```
## The following objects are masked from 'package:GGally':
##
##   ggnet, ggnet2
```

```
library(network)
library(sna)
```

## Cleaning the Data

```
imdb = read.csv("imdb.csv", header = T, na.strings=c("", "NA"))
awards = read.csv("awards.csv", header = T, na.strings = c("", "NA"))
cpi = read.csv("cpi.csv", header = T)
colnames(awards) = tolower(colnames(awards))

names(cpi) = c("year", "cpi")
colSums(is.na(imdb))
```

```
##           color           director_name
##           19             104
## num_critic_for_reviews      duration
##           50             15
## director_facebook_likes  actor_3_facebook_likes
##           104             23
## actor_2_name      actor_1_facebook_likes
##           13             7
## gross           genres
##           884             0
## actor_1_name      movie_title
##           7             0
## num_voted_users  cast_total_facebook_likes
##           0             0
## actor_3_name      facenumber_in_poster
##           23             13
## plot_keywords      movie_imdb_link
##           153             0
```

```
##      num_user_for_reviews      language
##              21              12
##      country      content_rating
##              5              303
##      budget      title_year
##      492              108
##      actor_2_facebook_likes      imdb_score
##              13              0
##      aspect_ratio      movie_facebook_likes
##      329              0
```

Clearly there are a significant number of missing observations in this data set. Because this analysis is focused only on creating a network plot, we can omit these rows with NAs.

```
imdb = na.omit(imdb)
imdb = imdb %>% select(gross, genres, movie_title, country, movie_imdb_link, budget, title_year, imdb_score)

names(imdb) = c("gross", "genres", "title", "country", "links", "budget", "year", "score", "rating")

#simplify the genres by taking the first entry
imdb$genres = as.character(imdb$genres)
imdb$genres_simple = strsplit(imdb$genres, split = "|", fixed = TRUE)
imdb$genres_simple = as.character(imdb$genres_simple)
imdb$genres_simple = str_extract(imdb$genres, pattern = "[A-Za-z]{1,20}")
imdb$genres_simple = as.factor(imdb$genres_simple)
imdb$title = gsub(imdb$title, pattern = "?", replacement = "")
imdb$links = as.character(imdb$links)

#Convert data types
imdb$genres = as.factor(imdb$genres)
imdb$budget = as.numeric(imdb$budget)
imdb$gross = as.numeric(imdb$gross)
imdb$score = as.numeric(imdb$score)
imdb$rating = as.factor(imdb$rating)

levels(imdb$genres_simple)
```

```
## [1] "Action"      "Adventure"   "Animation"   "Biography"   "Comedy"
## [6] "Crime"       "Documentary" "Drama"       "Family"      "Fantasy"
## [11] "Horror"      "Musical"     "Mystery"     "Romance"     "Sci"
## [16] "Thriller"    "Western"
```

```
link_pat = '(.){35}'
imdb$links = str_extract(imdb$links, pattern = link_pat)
```

## Using CPI data to convert to adjust for inflation

This uses data from the US census to adjust for inflation. I also filter out movies not made in the USA.

```

imdb = inner_join(imdb, cpi, by = "year")
reference_year_cpi = filter(imdb, year == 2016)$cpi[1]
imdb$gross_adj = reference_year_cpi / imdb$cpi

imdb = imdb %>%
  group_by(year) %>%
  mutate(cpi_ratio = reference_year_cpi/cpi) %>%
  mutate(gross_adj = gross*cpi_ratio) %>%
  mutate(budget_adj = budget*cpi_ratio) %>%
  select(-gross, -budget) %>%
  ungroup() %>%
  filter(country == 'USA')

```

## Network Analysis of Linked Movies

On imdb.com, for each movie, there are 12 recommended movies. For each movie in this imdb data set, I went to imdb.com and pulled down the 12 recommended movie links. That's what this below function does.

```

get_links <- function(address) {
  # read the movie page
  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 = "")

  recs = paste("http://www.imdb.com/title/", recs, sep = "")
  return(recs)
}

#Example

get_links(imdb$links[1])

```

```

## [1] "http://www.imdb.com/title/tt1392170"
## [2] "http://www.imdb.com/title/tt0120338"
## [3] "http://www.imdb.com/title/tt1454468"
## [4] "http://www.imdb.com/title/tt0416449"
## [5] "http://www.imdb.com/title/tt0454876"
## [6] "http://www.imdb.com/title/tt3659388"
## [7] "http://www.imdb.com/title/tt1010048"
## [8] "http://www.imdb.com/title/tt1951264"
## [9] "http://www.imdb.com/title/tt0480249"
## [10] "http://www.imdb.com/title/tt0371746"
## [11] "http://www.imdb.com/title/tt0848228"
## [12] "http://www.imdb.com/title/tt0903624"

```

Here I take a small sample at first to test the system.

```
mydata = filter(imdb, year >= 2016)
#View(mydata2)
dim(mydata)
```

```
## [1] 45 12
```

45 movies is a good starting point. This code below creates an adjacency matrix that is used for the network plot. The  $i,j$ th entry is 1 if movie  $i$  is connected to movie  $j$  and zero otherwise.

Here is the upper 1-3th quadrant of one such matrix.

```
m = matrix( c( 0, 1, 0, 1, 0, 1, 0, 1, 0), nrow = 3)

rownames(m) = c("Star Wars", "Toy Story", "Fiddler on the Roof")
colnames(m) = rownames(m)
m
```

```
##              Star Wars Toy Story Fiddler on the Roof
## Star Wars           0         1             0
## Toy Story           1         0             1
## Fiddler on the Roof 0         1             0
```

Here Star Wars is connected to Toy Story but not Fiddler on the Roof, and Fiddler on the Roof is connected to Toy Story but not Star Wars.

## Creating a Connections Matrix

```
summary(imdb$genres_simple)
```

```
##      Action  Adventure  Animation  Biography  Comedy  Crime
##       751      291      36      137      850      202
## Documentary  Drama      Family  Fantasy  Horror  Musical
##       22      499      3       31      138      2
##      Mystery  Romance      Sci  Thriller  Western
##       16       1       6       1       1
```

```
make_network = function(cur_data, cur_size = 10, add.label = T){
  #Due to a restriction on RcolorBrewer, we can only use a max of 9 genres.
  #Here I filter out all but the 9 most popular. I do so only when necessary

  if( length(unique( cur_data$genres_simple)) > 9) {
    cur_data = filter( cur_data, ( genres_simple %in% c("Action",
                                                         "Adventure",
                                                         "Animation",
                                                         "Comedy",
                                                         "Crime",
                                                         "Drama",
                                                         "Fantasy",
                                                         "Horror",
```

```

    "Biography"))))
  }

  n = nrow(cur_data)
  #Need an index number for each link to match
  vectorize = function(input_links){
    out = c(rep(0, n))
    for(i in input_links){
      index = match(i, cur_data$links)
      out[index] = 1
    }
    return(out)
  }

each_links = sapply(cur_data$links, get_links)
links_matrix = ldply(each_links, vectorize) %>%
  select(-1) %>%
  as.matrix()

#Add movie titles
rownames(links_matrix) = cur_data$title
colnames(links_matrix) = cur_data$title

#Remove empty connections, ie, movies with no links
# links_matrix = links_matrix[,which(!apply(links_matrix,2,FUN = function(x){all(x == 0)}))]
# links_matrix = links_matrix[which(!apply(links_matrix,2,FUN = function(x){all(x == 0)})), ]
#
net1 = network( links_matrix, directed = F, na.rm = F)

# cur_data = filter(cur_data, (title %in% rownames(links_matrix)))

network.vertex.names(net1) = rownames(links_matrix)

length = nrow(links_matrix)

ggnet2(net1,
  color = cur_data$genres_simple,
  palette = "Set1",
  label = add.label,
  size = cur_size)
}

```

## Movies in the US released after 2016

You can find interesting patterns between the linked movies over different years.

```
make_network(mydata)
```

```
## Loading required package: scales
```





## Movies from Prior to 1975

From this graph we see there are far fewer connections, and it appears movies that are released at the same time are more likely to be related, as are sequels.

```
mydata3 = filter(imdb, year > 1900, year < 1975, country == "USA")
dim(mydata3)
```

```
## [1] 45 12
```

```
make_network(mydata3, cur_size = 9)
```



## Movies from 2010 - 2014

Here there are too many movies to be able to read each title, so I just remove it.

Does it look like movies are generally grouped by genre?

```
mydata4 = filter(imdb, year > 2005, year < 2008, country == "USA")
dim(mydata4)
```

```
## [1] 253 12
```

```
make_network(mydata4, cur_size = 3, add.label = F)
```



