Assignment 10.2

2023-11-17

```
# Building the realratingmatrix
dfr_r2 <- dfr_r[,-4]
rmat2 <- as(as.data.frame(dfr_r2), 'realRatingMatrix')
```

Building the Matrix

In this step we have already imported the data and the packages necessary to create the recommender system and we are converting the ratings dataset in the given files to a matrix. The matrix conversion is necessary for a couple of reasons. The recommnder system uses matrices as the data format and a matrix allows us to reduce file size. Firstly, we remove timestamp since instead of converting a sparse matrix with the dcast() function we are skipping ahead to a dense matrix. Once the matrix is converted our file size decreases to 1.9mb. It's important to understand the value of this step as this reduces the file size to a 25th of its original size had we first converted to a sparse matrix.

```
# Building the recommender
m_sim <- similarity(rmat2, method = 'cosine', which = 'items')
m_sim.df <- as.data.frame(as(m_sim, 'matrix'))
# Building the empty dataframe for the recommender
# sim_movies <- data.frame(matrix(nrow = 9724, ncol = 51))
# colnames(sim_movies)[1] <- 'movieId'
# Building a for loop that iterates on each row or each movie title
#for(i in 1:nrow(m_sim.df))
#{
  #get main product name for which to get similar products
  #sim_movies[i,1]<-rownames(m_sim.df[i,1:52])

  #get the names of top 50 similar products
  #sim_movies[i,2:51] <-
names(head(n=50,(m_sim.df[,order(as.numeric(m_sim.df[i,]),decreasing=TRUE)])[
i,-1]))[1:50]
#}
```

Building the Recommender System

As you can see we have quite a bit happening in the above code. Firstly, we create the recommender system that is an item based recommender that will recommend items of similar ratings. We use the cosine similarity because that is the recommended method from both sources and it is often used in recommendation systems just like this one. Lastly, we convert it to a dataframe because we are going to create a recommendations dataframe

from the output of the recommender system which we will then use to pull our recommendations from. After the recommender system is built and we have the contents in a dataframe format, we then go about pulling the top 50 recommendations for each movie, where each row is a unique movieID, and the columns 2:51 represent the rankings of the 50 most similar titles in order. The reason the code is commented out, is because this step actually takes forever. For the 9000+ rows that I had it took about 30 minutes for the for loop to complete its looping process. I commented the code out for the RMarkdown file so I didn't have to wait an extra 30 minutes for the document to knit together, and instead I saved the sim_movies dataframe that is built from the for loop as a csv and imported it to make sure that my code afterwards would work.

```
# Creating a top 10 dataframe
t10 <- sim_movies[,1:11]
colnames(t10) <- c('movieId', '1st', '2nd', '3rd', '4th', '5th', '6th',
'7th', '8th', '9th', '10th')
```

Extracting the top10 dataframe

In this step, while it's a very simple step, I kept it separate because it is a part of the building of the recommender system rather than the function that I created to intake a movie title. As mentioned above it uses a saved version of the recommender dataframe that I imported at the beginning of the document.

```
# Building a function to recall the 10 recommendations for a movie given in
the function.
mymovies <- function (movie) {
  z <- dfr_m$movieId[dfr_m$title == movie]
  top_10 <- as(t10[t10$movieId == z,2:11], "list")
  top_10_df <- data.frame(top_10)
  top_10_df <- gather(top_10_df, value, movieId)
  top_10_df <- as.data.frame(top_10_df[,-1])
  colnames(top_10_df) <- 'movieId'
  top_10_df$movieId=as.numeric(top_10_df$movieId)
  names=left_join(top_10_df, dfr_m, by="movieId")

  return(names[,2:3])
}
```

```
> mymovies("Othello (1995)")
                                    title                                genres
1                        Sudden Death (1995)                                Action
2         Dracula: Dead and Loving It (1995)                        Comedy|Horror
3                               Balto (1995)       Adventure|Animation|Children
4                         Money Train (1995) Action|Comedy|Crime|Drama|Thriller
5                        It Takes Two (1995)                   Children|Comedy
6                     Dead Presidents (1995)                Action|Crime|Drama
7                      Big Green, The (1995)                   Children|Comedy
8                      Eye for an Eye (1996)                    Drama|Thriller
9  Lawnmower Man 2: Beyond Cyberspace (1996)            Action|Sci-Fi|Thriller
10                           Big Bully (1996)                    Comedy|Drama
```

output using the mymovies() function and Othello (1995)

Conclusion

Finally, in the code above, we create a function that takes a movie title given in the function, passes it as a filter in the dfr_m(movies) dataframe, and extracts the movieID. The movieid is then passed to the dataframe t10, which then creates a list of the top 10 movies for that movie id. The list is then converted to a dataframe, gathered to create a skinny rather than a long dataframe, the repeat column containing the original column names is removed, the remaining column is then labeled movieId so it can be joined to the dfr_m dataframe, and then we can extract the titles and genres for the top 10 movies for the recommendation given. We then pass 'Othello (1995)' into the function to check that it works and there we have it. Based off of this recommender system, we should watch

Sources

https://www.data-mania.com/blog/how-to-build-a-recommendation-engine-in-r/
https://towardsdatascience.com/find-similar-products-to-recommend-to-users-8c2f4308c2e4