Large-Scale Social and Complex Networks: Design and Algorithms
ECE 232E Summer 2018
Prof Vwani Roychowdhury
UCLA, Department of ECE

Project 5
IMDb Mining

Due on Monday, Sept. 3, 2018 by 11:59 PM PDT

Team Members

Jennifer MacDonald, UID: 604501712
Nguyen Nguyen, UID: 004870721
Sam Yang, UID: 604034791

Introduction

In this project, we studied the various properties of Internet Movie Database (IMDb). In the first part of the project, we explored and extracted the properties of a directed actor/actress network. In the second part, we explored the properties of undirected movie network.

1. Actor/Actress network

For this first part, we used two text files downloaded from Dropbox at https://ucla.box.com/s/z45q3g5zrpay8b8gtbql6ojaecb7kj2u to get the actors' data for our network:

- actor_movies.txt
- actress_movies.txt

In order to get the data ready for the following problems, we needed to preprocess and clean up the data. We did this in two steps:

1. We merged the two files we downloaded together and dropped any actors or actresses who had acted in less than 10 movies
2. We cleaned the data to remove multiples of movies that showed up with slightly different titles and merged the files

For example, there might be a movie that was listed twice as this:

- Movie X (voice)
- Movie X (as uncredited)

Since we don't want to count those as two different movies since that would create identical nodes, we made sure to remove all but one version.

QUESTION 1: Perform the preprocessing on the two text files and report the total number of actors and actresses and total number of unique movies that these actors and actresses have acted in.

In this question, we used python to preprocess all raw files. Moving forward, we will use python for all the preprocessing and R for all the analysis. The total number of unique actors and actresses are 113,121. The total number of unique movies that these actors and actresses have acted in are 463,219.

1. Directed actor/actress network creation

We used the file we just created to build a network of actors and actresses. The nodes are the actors/actresses and the edges between them are $w_{i\to j}$, given by the following equation:

$$w_{i\to j} = \frac{|S_i \cap S_j|}{|S_i|}$$

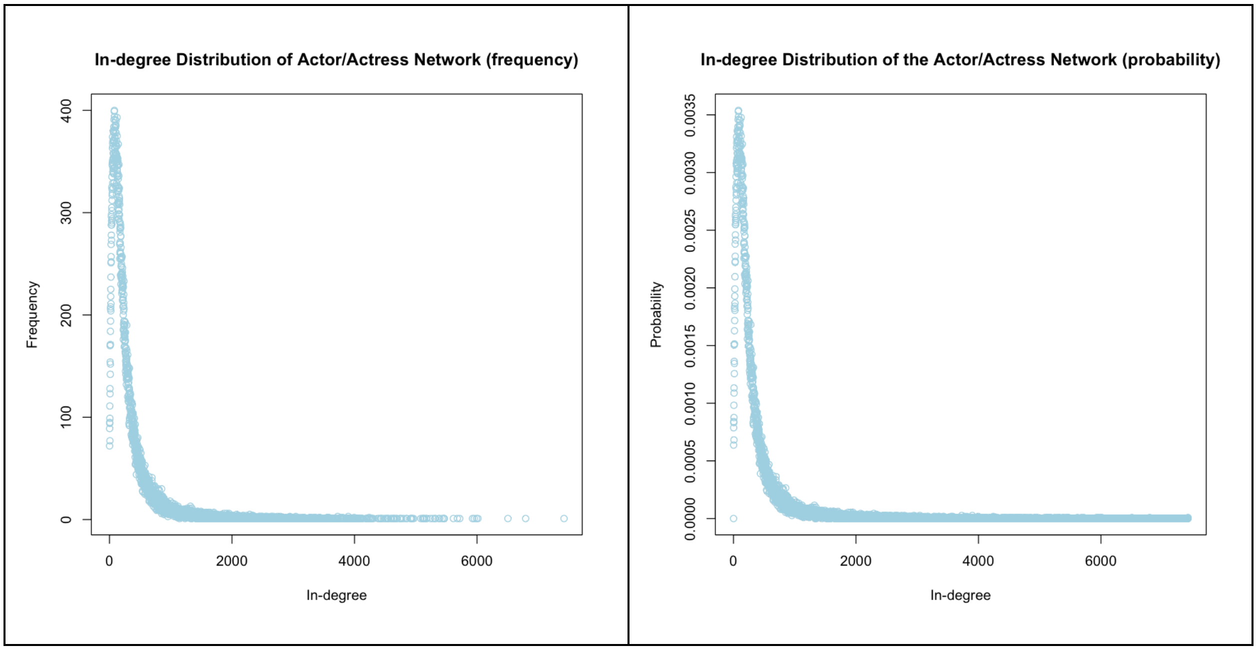
Where S_i is the set of movies in which the actor/actress i has acted in and S_j is the set of movies in which the actor/actress j has acted in.

QUESTION 2: Create a weighted directed actor/actress network using the processed text file and equation 1. Plot the in-degree distribution of the actor/actress network. Briefly comment on the in-degree distribution.

To create a weighted directed actor/actress network, we used the preprocessed filed in question 1. We sorted the in-degree and plotted the distribution as seen in Table 1 below. From our observation, we found that the in-degree distribution range is large. It ranges from 0 to 7000+ but the highest frequency is around 400-500. This indicates that most actors/actresses only collaborate in a smaller subset of the wide range of actors/actresses in the graph. This makes sense because we're only observing the distribution as a whole which includes different genres.

Thus, it is not intuitive to have an actor/actress who specialize in comedy to work with another actor/actress in drama.

Table 1: In-degree Distribution Plots



2. Actor pairings

In this section, we looked at the actor pairings between the following 10 actors:

- Tom Cruise
- Emma Watson (II)
- George Clooney
- Tom Hanks
- Dwayne Johnson (I)
- Johnny Depp
- Will Smith (I)
- Meryl Streep
- Leonardo DiCaprio
- Brad Pitt

QUESTION 3: Design a simple algorithm to find the actor pairings. To be specific, your algorithm should take as input one of the actors listed above and should return the name of the actor with whom the input actor prefers to work the most. Run your algorithm for the actors *listed above and report the actor names returned by your algorithm. Also for each pair, report the (input actor, output actor) edge weight. Does all the actor pairing make sense?*

In this question, we designed an simple algorithm to find the highest weighting between each pair of actor/actress as a measure of who each actor prefers to work with. We took the vertices that represented each of the actors/actresses listed for the input actor. To find the max weight, we took the neighbors of the actor we are examining and looked at the edge weight, saving the weight if it was larger than the previous highest weight. We would say that the actor pairing makes sense for the most part, although the Tom Cruise and Nicole Kidman relationship is surprising, given that they were married in 1990 and divorced in 2001, and haven't worked in any movies together (at least mainstream ones) since the divorce.

Table 2: Input Actors with Best Actor Paired and their Relationships

	Input Actor	**Best Actor Paired**	**Pairing Weight**	**Why Relationship Makes Sense**
1	Tom Cruise	Nicole Kidman	0.19298	Previously married
2	Emma Watson (II)	Daniel Radcliffe	0.56521	Starred in Harry Potter movies
3	George Clooney	Matt Damon	0.12121	Starred in Ocean's movies & others
4	Tom Hanks	Tim Allen (I)	0.10389	Starred in Toy Story movies & others
5	Dwayne Johnson (I)	Mark Calaway	0.23188	Wrestled in WWE movies
6	Johnny Depp	Helena Bonham Carter	0.08247	Always cast together in Tim Burton Movies
7	Will Smith (I)	Darrell Foster	0.13953	Life/Fitness coach Foster trains Smith
8	Meryl Streep	Kevin Kline (I)	0.06452	Starred in Ricki and the Flash movie and others
9	Leonardo DiCaprio	Martin Scorsese	0.11111	Scorsese directs movies with DiCaprio as lead

10	Brad Pitt	George Clooney	0.10294	Starred in Ocean's movies and others

3. Actor rankings

Then we extracted the top 10 actors/actresses from the network.

QUESTION 4: Use Google's PageRank algorithm to find the top 10 actor/actress in the network. Report the top 10 actor/actress and also the number of movies and the in-degree of each of the actor/actress in the top 10 list. Does the top 10 list have any actor/actress listed in the previous section? If it does not have any of the actor/actress listed in the previous section, please provide an explanation for this phenomenon.

To extract the top 10 actors/actresses from the network, we applied Google's PageRank algorithm and sorted from highest to lowest. We then used the actor and movie indexes we created to look up the name of the actor/actress and the number of movies they were credited in. The results are seen below in Table 3.

Table 3: Top 10 actor/actresses in the network and related statistics

Rank	Actor/Actress name	Number of movies in	In-degree	Why This Makes Sense
1	Bess Flowers	828	7420	Known as "Queen of the Extras"
2	Fred Tatasciore	353	3810	Very popular voice actor in animated films and games
3	Steve Blum (IX)	373	3187	Veteran voiceover actor
4	Sam Harris (II)	600	6793	A very popular extra in the early-to mid-1900s
5	Harold Miller (I)	561	6504	A very

				popular extra in the early- to mid-1900s
6	Yuri Lowenthal	316	2627	Very popular voice actor in animated films and games
7	Ron Jeremy	635	2821	Very well-known porn star
8	Lee Phelps (I)	647	5466	A very popular extra in the early- to mid-1900s
9	Robin Atkin Downes	267	2886	Very popular voice actor in games and films
10	Frank O'Connor (I)	623	5392	A director and very popular extra in the early- to mid-1900s

As seen in the table, there are no actors/actresses that were listed in the previous section, whose names are far more familiar to the average moviegoer. In fact, very few of the names of the top 10 actors/actresses are recognizable household names (with maybe the exception of Ron Jeremy, depending on your inclinations). This is because these actors are either extras, voice actors, or porn stars and thus not very recognizable (again, possible exceptions depending on your viewing preferences). These roles tend to give you a lot of acting credits without the star power. In the next section we show the number of movies very famous actors have been in. The actors with the highest PageRanks have acted in 5-10 times more movies than the famous actors. PageRank captures famous websites from Google searches because everyone links to famous websites. However, PageRank doesn't necessarily capture famous actors/actresses in this graph because fame doesn't lead to more connections. In fact, it appears that it's utility roles like being an extra or being a voice actor that help increase the PageRank score.

QUESTION 5: Report the PageRank scores of the actor/actress listed in the previous section. Also, report the number of movies each of these actor/actress have acted in and also their in-degree.

We also applied Google's PageRank algorithm to the actors/actresses listed in the previous section. For each actor, we used the actor and movie indexes to get the number of movies the actors started in. These results, as well as the PageRank scores and in-degree are seen below in Table 4.

Table 4: Input actors PageRank scores, number of acted movies, and in-degree

	Actor/Actress name	**PageRank scores**	**Number of movies in**	**In-degree**
1	Tom Cruise	3.9947e-05	57	1573
2	Emma Watson	1.7439e-05	23	411
3	George Clooney	4.0482e-05	66	1509
4	Tom Hanks	5.2725e-05	77	2023
5	Dwayne (I) Johnson	4.2513e-05	69	1306
6	Johnny Depp	5.5442e-05	97	2094
7	Will (I) Smith	3.2350e-05	43	1258
8	Meryl Streep	4.0765e-05	93	1556
9	Leonardo DiCaprio	3.2092e-05	45	1251
10	Brad Pitt	4.3945e-05	68	1689

2. Movie network

Next, we created a new network, this time with the movies as the nodes and the edges connecting other movies with common actors/actresses.

1. Undirected movie network creation

We took the processed file generated from the earlier questions to create the movie network, connecting the movie notes with weighted edges and eliminating any movies with less than five actors/actresses. To determine the edge weights $w_{i \rightarrow j}$, we used the following formula:

$$w_{i \to j} = \frac{|A_i \cap A_j|}{|A_i \cup A_j|}$$

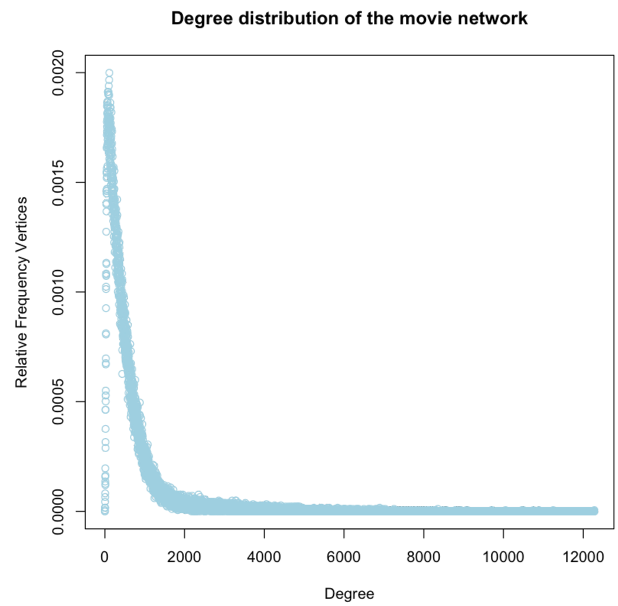
Where A_i is the set of actors in movie i and S_j is the set of actors in movie j. Since the relationship is bilateral, the network is undirected.

In order to construct the movie network, we took the combined and cleaned dataset we created from earlier and iterated through, taking the movie titles and adding them to a hashmap (we decided to do all preprocessing in Python for ease of coding). We then iterated through the list, removing any movies with less than five actors. We saved this as a text file with a number ID and the movie title that ID is supposed to represent. After that, we created another text file with the mapping of each movie node by ID to another movie node, as well as its calculated edge weight by dividing the intersection actors in common divided by the union of all actors between the two movies. We saved this as the first movie node as an ID, the second movie node as an ID, and the edge weight between the two nodes on each line in the text file.

QUESTION 6: Create a weighted undirected movie network using equation 2. Plot the degree distribution of the movie network. Briefly comment on the degree distribution.

The plot of the degree distribution of the movie network is seen below in Figure 1. Many of the plot points are clustered around a degree value of close to 0, and the degree distribution is again very large from 0 to over 12000. Out of the dataset, the largest clustering tends to be between 0 and 2000. This indicates that most movies have common actors/actresses with a *relatively* small amount of other movies, compared to the end of the spectrum in the several thousands. These far out movies most likely have actors/actresses from our top ten list who have a ton of acting credit.

Figure 1: Degree distribution of the movie network



2. Communities in the movie network

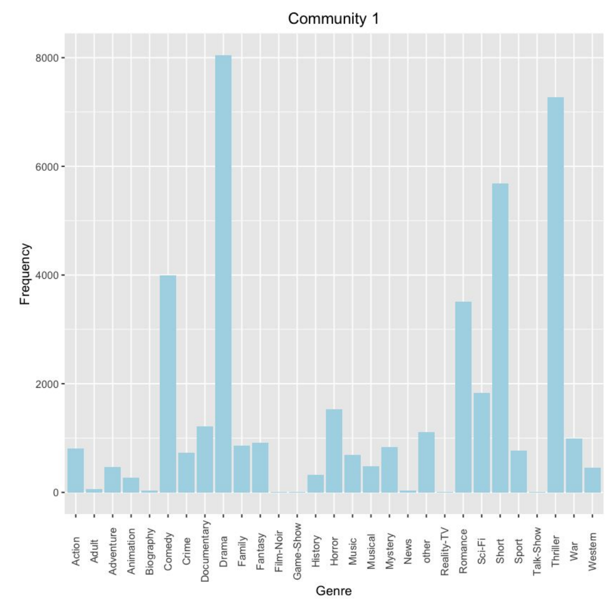
We will get the communities from our network and example the relationships of the movie genres from **movie_genre.txt**.

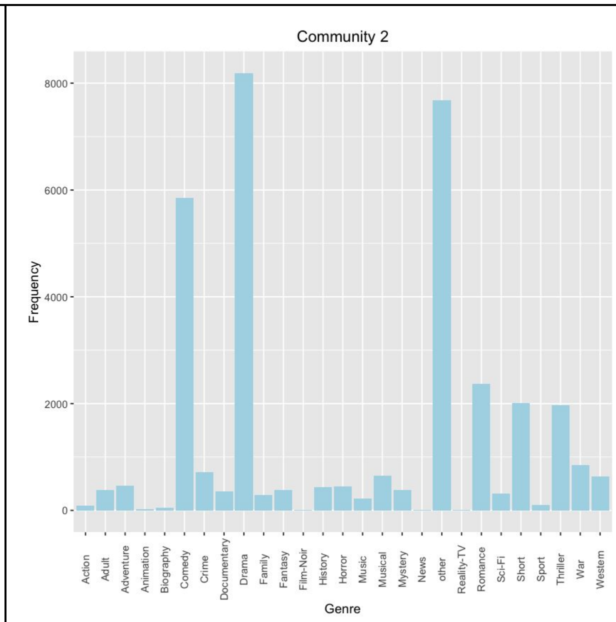
QUESTION 7: Use the Fast Greedy community detection algorithm to find the communities in the movie network. Pick 10 communities and for each community plot the distribution of the genres of the movies in the community.

We used our movie network used iGraph's Fast Greedy community detection algorithm to find the communities in the movie network. We then could use our movie index to map the movie IDs into the movies, and then look up the genre of each movie using the movie_genre.txt, which lists the genre for a subset of the movies in the network. We took ten of the communities and looped through each ID in that community, obtaining the genre using the id-to-movie-to-genre method (as long as the genre existed. Otherwise, we counted the movie as a genre of "other" or "unknown". Once we had the genres collected for each community, we plotted the results on a histogram. The results from the first ten communities can be seen below in Table 5.

Table 5: Distribution of movies by each genre in ten communities

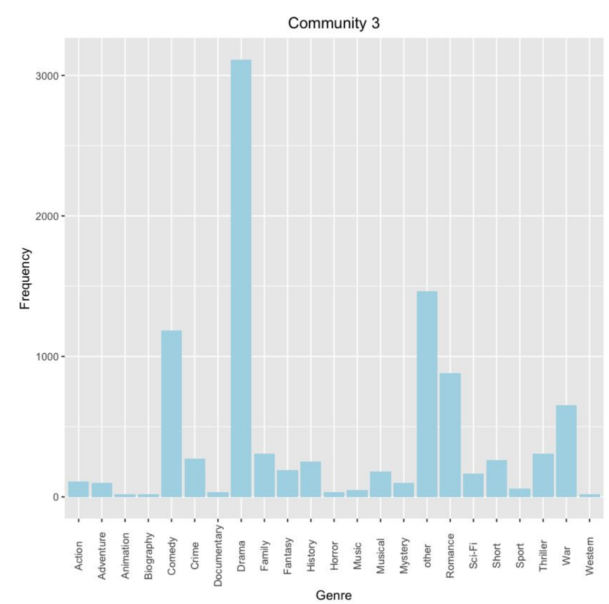
Community 1	**Community 2**

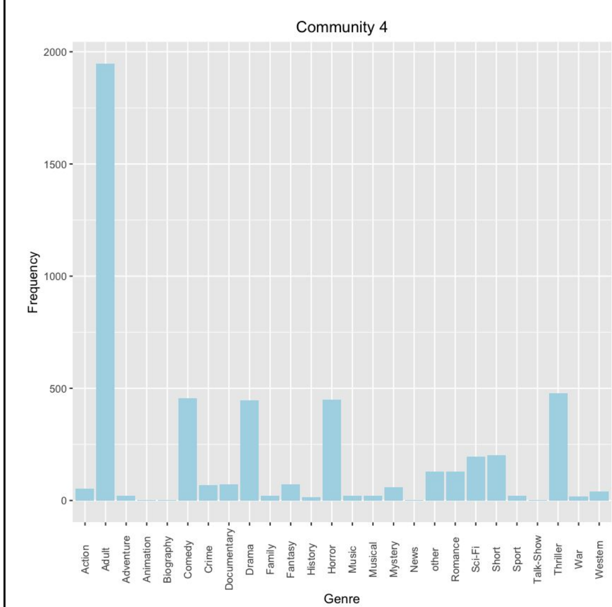




Community 3

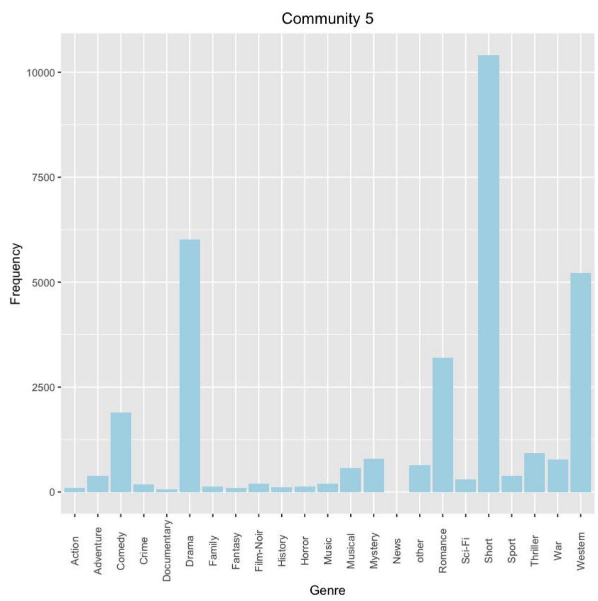
Community 4

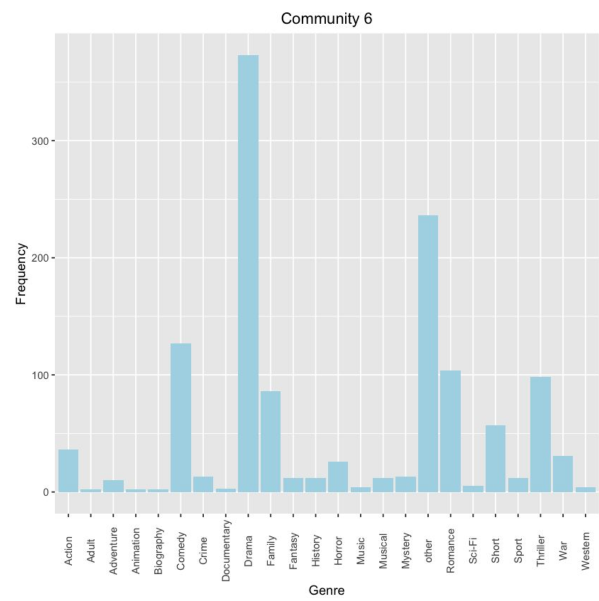




Community 5

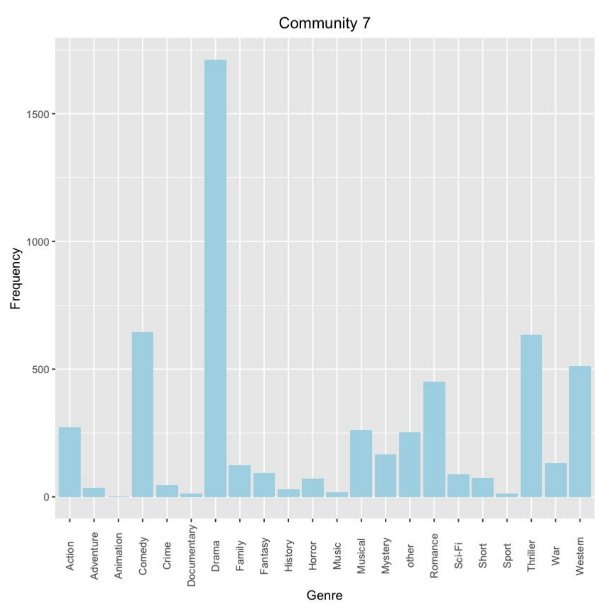
Community 6

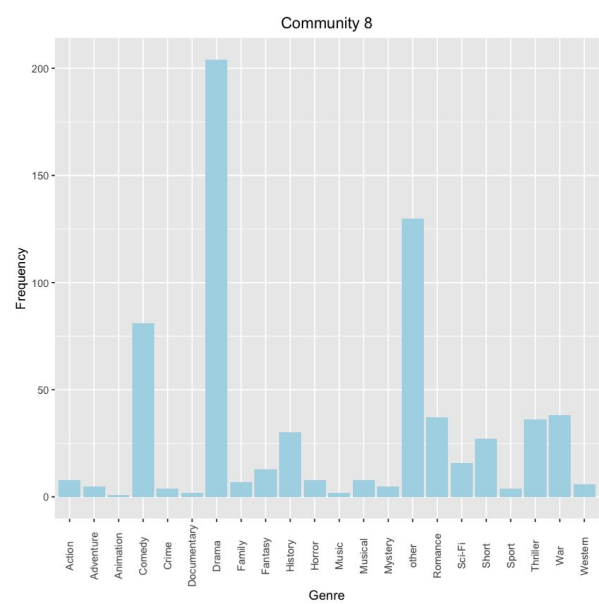




Community 7

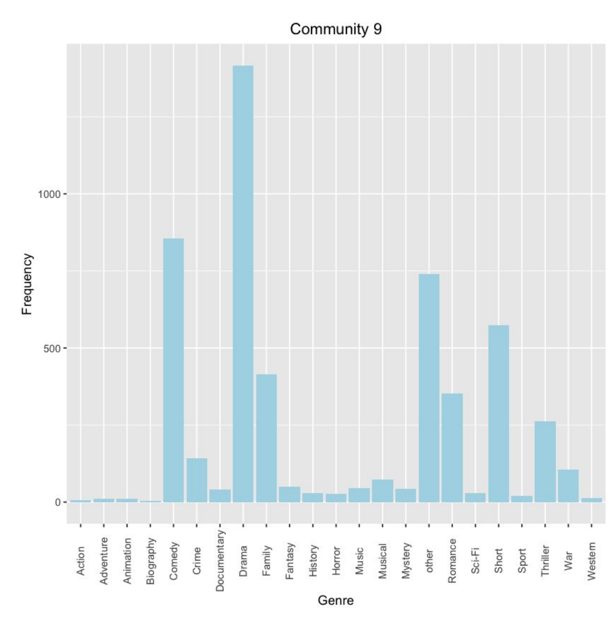
Community 8

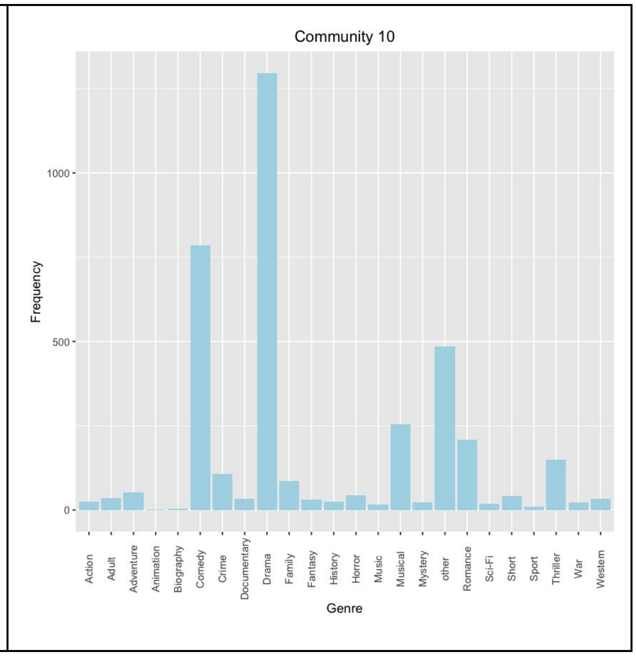




Community 9

Community 10





For many of the communities we plotted (eight in fact!), Drama was the most common genre. This makes sense because dramatic movies make up much of the movies that have been released since the movies were introduced, and made up a huge chunk of the films released in the 20th century. There are also some close categories in some of the communities, like the Thriller genre in community 1.

QUESTION 8:

8.1: In each community determine the most dominant genre based simply on frequency counts. Which genres tend to be the most frequent dominant ones across communities and why?

In the same manner as question 7, we used our movie index to map the movie IDs into the movies, and then looked up the genre of each movie using the movie_genre.txt, which lists the genre for a subset of the movies in the network. Then, for each community, we looped through each ID in that community, obtaining the genre using the id-to-movie-to-genre method (as long as the genre existed. Once we had the genres collected for each community, searched for the most frequent term that occured, which was our most dominant genre for that community. The most dominant genres for each community is seen below in Table 6.

Table 6: Most frequent genre in each community

Community	**Genre**	**Community**	**Genre**
1	Drama	2	Drama

3	Drama	4	Adult
5	Sport	6	Drama
7	Drama	8	Drama
9	Drama	10	Drama
11	Drama	12	Drama
13	Drama	14	Comedy
15	Drama	16	Drama
17	Drama	18	Drama
19	Drama	20	Romance
21	Drama	22	Short
23	Adult	24	Short
25	Short	26	Short
27	Thriller	28	Short

The "Drama" genre tends to dominate across the communities, with 17 out of the 28 communities dominated by the Drama genre. Other dominant genres in the communities include Sport, Adult, Short, Thriller, Comedy, and Romance. This mimics our the results in the previous problem, and tells us that there are a lot of drama movies, perhaps related to the earlier 20th century films where a large amount of films were dramatic, or even today with the emergence of Korean dramas becoming very popular.

8.2: In each community, for the i th genre assign a score of ln(c(i)) × p(i) q(i) where: c(i) is the number of movies belonging to genre i in the community; p(i) is the fraction of genre i movies in the community, and q(i) is the fraction of genre i movies in the entire data set. Now determine the most dominant genre in each community based on the modified scores. What are your findings and how do they differ from the results in 8.1.

For each genre in each community, we looped through in the same manner as we did previously to get the genre label and then calculated a dominance score according to the formula above once we had collected the exact percentage of that genre. The results are shown below in Table 7, with the most dominant genre in each community bolded. A cleaner summary of the dominant genre in each community is seen in Table 8.

Table 7: ln(c(i) x p(i)/q(i) scores for each genre in each community

Community	Modified Scores for Related Genres	Community	Modified Scores for Related Genres
1	Action : 0.00003316414 Adult : 0.000002322515 Adventure : 0.00002947751 Animation : 0.00009468099 Biography : 0.00001789651 Comedy : 0.00003941985 Crime : 0.00003868167 **Documentary : 0.0001041568** Drama : 0.00004052357 Family : 0.00004198199 Fantasy : 0.00005659115 Film-Noir : 0.0000003725511 Game-Show : 0.0000312306 History : 0.00002807181 Horror : 0.00007551286 Music : 0.00007821632 Musical : 0.00002116935 Mystery : 0.00004685857 News : 0.00006316321 Reality-TV : 0.0000208204 Romance : 0.00003919552 Sci-Fi : 0.00009684927 Short : 0.00005809422 Sport : 0.00007781598 Talk-Show : 0.00001918638 Thriller : 0.0001030472 War : 0.0000355996 Western : 0.00000925663	2	Action : 0.000002912199 Adult : 0.00002643657 Adventure : 0.0000355776 Animation : 0.000005831291 Biography : 0.00003504675 **Comedy : 0.00007450864** Crime : 0.00004635568 Documentary : 0.00003181378 Drama : 0.00005086603 Family : 0.00001462667 Fantasy : 0.00002555466 Film-Noir : 0.0000004586929 History : 0.00004868352 Horror : 0.00002247923 Music : 0.00002597589 Musical : 0.00003683591 Mystery : 0.00002315882 News : 0.000004416075 Reality-TV : 0.00001049897 Romance : 0.00003103165 Sci-Fi : 0.00001606081 Short : 0.0000223112 Sport : 0.000008910332 Thriller : 0.00002931432 War : 0.00003670966 Western : 0.00001659553
3	Action : 0.00001366535 Adventure : 0.0000208064 Animation : 0.0000174275 Biography : 0.00004189303 Comedy : 0.0000438308 Crime : 0.00005400591 Documentary : 0.000006136532 Drama : 0.00006154401 Family : 0.00005596103	4	Action : 0.00001140212 **Adult : 0.001212245** Adventure : 0.000005297348 Animation : 0 Biography : 0 Comedy : 0.00002897438 Crime : 0.00002032028 Documentary : 0.00003299395 Drama : 0.00001321639 Family : 0.000004000285

	Fantasy : 0.00004074681 History : 0.00009201059 Horror : 0.000003800237 Music : 0.00001449287 Musical : 0.0000293306 Mystery : 0.00001658964 Romance : 0.00003583911 Sci-Fi : 0.00002594628 Short : 0.000007574111 Sport : 0.00001634829 Thriller : 0.0000124446 **War : 0.00009580091** Western : 0.0000007553006		Fantasy : 0.00002404931 History : 0.000004832445 Horror : 0.000160825 Music : 0.000009624478 Musical : 0.000003748741 Mystery : 0.0000177449 News : 0 Romance : 0.000007379898 Sci-Fi : 0.00006395421 Short : 0.00001109149 Sport : 0.000007897739 Talk-Show : 0 Thriller : 0.00004077826 War : 0.000002512657 Western : 0.000004108608
5	Action : 0.000003239044 Adventure : 0.00003089779 Comedy : 0.00002224874 Crime : 0.000009563422 Documentary : 0.000004733599 Drama : 0.0000385539 Family : 0.00000609321 Fantasy : 0.00000499471 Film-Noir : 0.0001564316 History : 0.00001089734 Horror : 0.000005423839 Music : 0.00002267546 Musical : 0.00003440266 Mystery : 0.00005735984 News : 0 Romance : 0.0000465006 Sci-Fi : 0.00001545202 Short : 0.0001495934 Sport : 0.00004468124 Thriller : 0.00001314926 War : 0.0000352493 **Western : 0.0001941902**	6	Action : 0.0000263494 Adult : 0.0000004397249 Adventure : 0.000007855435 Animation : 0.000002888113 Biography : 0.00000694258 Comedy : 0.00002457356 Crime : 0.000008948907 Documentary : 0.000001339684 Drama : 0.00004150835 **Family : 0.00009248345** Fantasy : 0.000009142857 History : 0.0000150491 Horror : 0.00001907207 Music : 0.000003220944 Musical : 0.000007199011 Mystery : 0.000009293709 Romance : 0.00002214665 Sci-Fi : 0.000001908002 Short : 0.0000091443 Sport : 0.00001516667 Thriller : 0.00002403208 War : 0.00001844873 Western : 0.0000006152776
7	Action : 0.00007024925 Adventure : 0.000009266627 Animation : 0.000001555557 Comedy : 0.00003775883	8	Action : 0.000006471962 Adventure : 0.000005229251 Animation : 0 Comedy : 0.00002708155

	Crime : 0.00001070809 Documentary : 0.000003402319 Drama : 0.000054232 Family : 0.00003301023 Fantasy : 0.00002928077 History : 0.00001217141 Horror : 0.00001570579 Music : 0.000006846293 Musical : 0.00007979404 Mystery : 0.00005319718 Romance : 0.00002863054 Sci-Fi : 0.00002145856 Short : 0.000002815494 Sport : 0.000004257329 Thriller : 0.00004956527 War : 0.00002530525 **Western : 0.00008047081**		Crime : 0.00000283468 Documentary : 0.000001073329 Drama : 0.00003883453 Family : 0.000006263874 Fantasy : 0.00001947392 **History : 0.00009808732** Horror : 0.00000713408 Music : 0.000001533783 Musical : 0.000007649956 Mystery : 0.000004272203 Romance : 0.00001166826 Sci-Fi : 0.0000200346 Short : 0.000006725685 Sport : 0.000005372234 Thriller : 0.00001314262 War : 0.00004562929 Western : 0.000002272099
9	Action : 0.0000005342336 Adventure : 0.000001911225 Animation : 0.00001167122 Biography : 0.000006756512 Comedy : 0.00005609521 Crime : 0.00004634002 Documentary : 0.00001505754 Drama : 0.00004697501 **Family : 0.0001465355** Fantasy : 0.00001422595 History : 0.00001252893 Horror : 0.000005110835 Music : 0.00002420841 Musical : 0.00001808677 Mystery : 0.00001064532 Romance : 0.00002310144 Sci-Fi : 0.000005886124 Short : 0.00003513152 Sport : 0.000007414357 Thriller : 0.00001898445 War : 0.00002084299 Western : 0.0000008049867	10	Action : 0.000005262795 Adult : 0.00001333402 Adventure : 0.00002425148 Animation : 0.0000009756624 Biography : 0.00000938137 Comedy : 0.00007060624 Crime : 0.0000458459 Documentary : 0.00001646371 Drama : 0.00005896838 Family : 0.00003124276 Fantasy : 0.00001056888 History : 0.000013004 Horror : 0.00001266402 Music : 0.000009451077 **Musical : 0.0001152433** Mystery : 0.000006790256 Romance : 0.00001729452 Sci-Fi : 0.000004481012 Short : 0.000002040927 Sport : 0.000003956397 Thriller : 0.00001347142 War : 0.000003981247 Western : 0.000004494141
11	Action : 0.0001151497 Adult : 0	12	Action : 0.00003522631 Adult : 0.000002090186

	Adventure : 0.0001490231 Animation : 0.000001362198 Biography : 0.000005509301 Comedy : 0.00003988829 Crime : 0.00004226601 Documentary : 0.00000451137 Drama : 0.00004450982 Family : 0.000003062214 Fantasy : 0.00006637696 History : 0.00001337686 Horror : 0.00005846208 Music : 0.0000009273156 Musical : 0.00002286434 Mystery : 0.00001790503 Romance : 0.0000508864 Sci-Fi : 0.000005427485 Short : 0.00000008462778 Sport : 0.000006045712 Thriller : 0.0000272174 War : 0.00001525658 Western : 0.0000002366477		Adventure : 0.00000429285 Animation : 0.00002887369 Biography : 0.0000147108 Comedy : 0.00002286079 Crime : 0.00003775817 Documentary : 0.000002452666 **Drama : 0.00004741101** Family : 0.000004791315 Fantasy : 0.00001819487 Film-Noir : 0 History : 0.00002348008 Horror : 0.00003146812 Music : 0.00001073672 Musical : 0.000006838868 Mystery : 0.00002864311 Romance : 0.00001838549 Sci-Fi : 0.00003445275 Short : 0.000003523014 Sport : 0.00001044358 Thriller : 0.0000159082 War : 0.00001965816 Western : 0.0000003259316
13	**Action : 0.00005435904** Adventure : 0.000003207251 Comedy : 0.00001458688 Crime : 0.00002545464 Documentary : 0 Drama : 0.00005366406 Family : 0.000002269656 Fantasy : 0.000001713244 History : 0.000009198416 Horror : 0.00001006808 Music : 0 Musical : 0.000002269656 Mystery : 0.000007860798 Romance : 0.00005239537 Sci-Fi : 0.000002229015 Sport : 0.00001434493 Thriller : 0.00002696654 War : 0.00003925441	14	**Action : 0.00003932412** Adventure : 0 Biography : 0 Comedy : 0.0000182242 Crime : 0.000005167116 Documentary : 0.000003205225 Drama : 0.00001746925 Family : 0.000001903721 Fantasy : 0.000002417756 History : 0.00002387769 Horror : 0 Music : 0.000001926546 Musical : 0.000001903721 Mystery : 0.00001988449 Romance : 0.00000445605 Sci-Fi : 0 Thriller : 0.000006163301 War : 0.00001997057
15	Action : 0.000001893937	16	Action : 0.0000001968658

	Adventure : 0.000001631722 Animation : 0.000002491095 Biography : 0 Comedy : 0.00002630888 Crime : 0.00000690264 Documentary : 0.00001424166 Drama : 0.00006006022 Family : 0.00002402736 Fantasy : 0.00001798433 History : 0.00001931081 Horror : 0.000003230527 Music : 0.000008334519 Musical : 0.000005492632 Mystery : 0.0000007923348 Romance : 0.00001954943 Sci-Fi : 0.000002198577 Short : 0.00001126464 Sport : 0.000003530361 Thriller : 0.00001691965 **War : 0.0001812006** Western : 0.000001303625		Adult : 0.000003668753 Adventure : 0.00000323096 Biography : 0.000004826997 **Comedy : 0.00009983814** Crime : 0.00002407222 Documentary : 0.000003038252 Drama : 0.00005092356 Family : 0.000002792379 Fantasy : 0.0000002955305 History : 0.000005837299 Horror : 0 Music : 0.000003249886 Musical : 0.00001350772 Mystery : 0.000006461677 Romance : 0.00006975071 Sci-Fi : 0.000003795828 Short : 0.0000007276884 Sport : 0.000002845766 Thriller : 0.000009135258 War : 0.00003773888 Western : 0.0000006208063
17	**Action : 0.0002494481** Adult : 0.0000004117395 Adventure : 0.000005105926 Biography : 0.00005200587 Comedy : 0.00003637239 Crime : 0.000005055609 Documentary : 0.0000007657065 Drama : 0.00004910004 Family : 0.000004310578 Fantasy : 0.00006090739 History : 0.000004364931 Horror : 0.00003583654 Music : 0.000003130869 Musical : 0.00002195796 Mystery : 0.000001720962 Romance : 0.00004631205 Sci-Fi : 0.000009775041 Short : 0.0000003097205 Sport : 0.00001110887 Thriller : 0.000009636242	18	Action : 0.000006177426 **Adventure : 0.0002053774** Biography : 0.000006544969 Comedy : 0.00002541018 Crime : 0.00003389494 Documentary : 0.0000005312254 Drama : 0.00005082798 Family : 0.000008109522 Fantasy : 0.00002259197 History : 0.00002041966 Horror : 0.0000004950539 Musical : 0.000003192127 Mystery : 0.000002915186 Romance : 0.0001185686 Sci-Fi : 0.000004170772 Short : 0.00000008724856 Sport : 0.0000006647234 Thriller : 0.000004568372 War : 0.00002188735 Western : 0.000007536835

	War : 0.00002543295 Western : 0.000001766301		
19	Action : 0.000106009 Adventure : 0.000003102037 Animation : 0.000002288509 Biography : 0.000004117869 Comedy : 0.00001428362 Crime : 0.00001309098 Documentary : 0.000001238485 Drama : 0.00004173346 **Family : 0.0001215415** Fantasy : 0.00002582148 Game-Show : 0 History : 0.0000191931 Horror : 0.00001202608 Music : 0.00000263291 Musical : 0.00006432158 Mystery : 0.00001407803 News : 0 Reality-TV : 0 Romance : 0.00007591003 Sci-Fi : 0.000002579039 Short : 0.0000001351921 Sport : 0.000005302917 Thriller : 0.00004764796 War : 0.000004350154 Western : 0.0000001886059	20	Action : 0 Adventure : 0.000001012322 Comedy : 0.00001756667 Crime : 0.00002333483 Documentary : 0 Drama : 0.00001897611 Family : 0 Fantasy : 0.000002163043 History : 0.00001161338 Horror : 0.000002672297 Music : 0 Musical : 0.00001189888 Mystery : 0.00001373698 **Romance : 0.0000702356** Sci-Fi : 0.000001672666 Sport : 0.000001509257 Thriller : 0.00001689243 War : 0.000009784481
21	Comedy : 0.00001350049 Crime : 0 Drama : 0.00001905533 Fantasy : 0 Romance : 0 **Sport : 0.00004297788** Thriller : 0.000004519341 War : 0 Western : 0	22	**Short : 0.0001375413**
23	**Adult : 0.0007504639** Romance : 0	24	Comedy : 0.000004725171 Drama : 0.000002223122 Romance : 0.000005424005 **Short : 0.00005632767** Thriller : 0

25	**Short : 0.0002119772** Western : 0	26	**Short : 0**
27	Short : 0.000005867466 Sport : 0 **Thriller : 0.000112817**	28	**Short : 0.00005838786** Sport : 0

Table 8: Most frequent genre in each community with modified scores

Community	Genre	Community	Genre
1	Documentary	2	Comedy
3	War	4	Adult
5	Western	6	Family
7	Western	8	History
9	Family	10	Musical
11	Adventure	12	Drama
13	Action	14	Action
15	War	16	Comedy
17	Action	18	Adventure
19	Family	20	Romance
21	Sport	22	Short
23	Adult	24	Short
25	Short	26	Short
27	Thriller	28	Short

As we can see, there are significantly less Drama-dominant communities using this new, modified score. In fact, only one drama remained out of the originally-labeled 17! We can see the differences more easily below in Table 9, where the cell is bolded if a change occurred from the first to the second method .

Table 9: Comparison of frequent genre in each community using two methods

Community	Genre	Community	Genre
1	**Drama/Documentary**	2	**Drama/Comedy**
3	**Drama/War**	4	Adult/Adult
5	**Sport/Western**	6	**Drama/Family**
7	**Drama/Western**	8	**Drama/History**
9	**Drama/Family**	10	**Drama/Musical**
11	**Drama/Adventure**	12	Drama/Drama
13	**Drama/Action**	14	**Comedy/Action**
15	**Drama/War**	16	**Drama/Comedy**
17	**Drama/Action**	18	**Drama/Adventure**
19	**Drama/Family**	20	Romance/Romance
21	**Drama/Sport**	22	Short/Short
23	Adult/Adult	24	Short/Short
25	Short/Short	26	Short/Short
27	Thriller/Thriller	28	Short/Short

We can see that there is a significant difference in results from the first method to the second, with 18 out of the 28 communities' dominant genre being reclassified. In 16 out of those 18 instances, the switch was from Drama to another genre. This is because even though numerically the Drama classification was high in many of the communities, where when you are taking its fraction over the community and the dataset, the fraction between the community and the dataset is actually lower than that of other genres.

8.3: Find a community of movies that has size between 10 and 20. Determine all the actors who acted in these movies and plot the corresponding bipartite graph (i.e. restricted to these particular movies and actors). Determine three most important actors and explain how they help form the community. Is there a correlation between these actors and the dominant genres you found for this community in 8.1 and 8.2.

To make the problem as simple as possible, we decided to find the smallest community in the community of movies that still fulfilled the requirements. We looped through all the communities, checking to see if the community size was between 10 and 20 and smaller than the current "best sized" community. We found the optimal community size in community 27 to be 12.

After we obtained all the movie IDs, we used our movie index to obtain the movie titles, seen below in Table 10.

Table 10: Movies in community

Cent jours avant le lendemain (2015)	669: Escape the Reality (2011)	An Olimatsim adventure (2011)	L'affaire Hawkins (2014)
La peur anonyme (2014)	La Peur aux trousse (2015)	Les oiseaux se cachaient pour mourir (2015)	Midnight Stranger (2011)
New York Vengeance (2013)	Des humains bien tranquilles (2016)	Les années folles (2016)	Mocakoma (2013)

As you can see, many of the movies have French titles and don't seem to be widely-known movies. This makes sense, as a smaller community would not have famous titles (because a movie of that caliber would most likely belong to a larger, better-connected community).

Then, for each movie, we used our actors in movie index to find all the actors corresponding to each movie, and used our actor index to translate the actor IDs into names, as seen in Table 11 below.

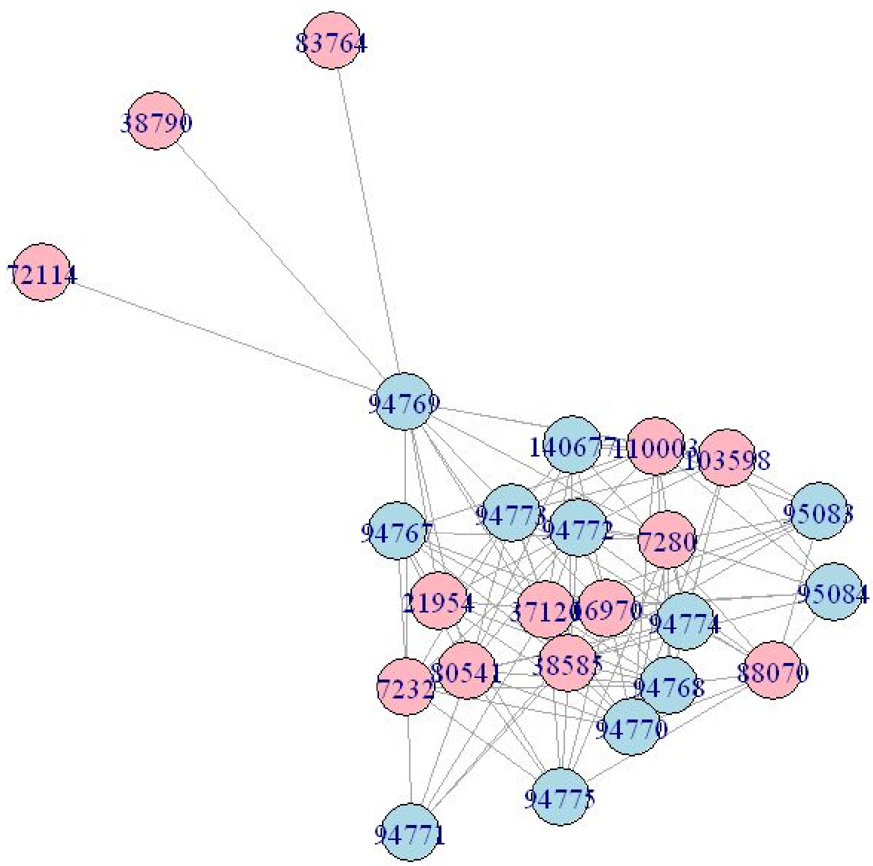
Table 11: Actors in community

Mathieu Bourassa-Simpson	Joshua Leonard	Michael C. Williams	Heather (I) Donahue
Andréanne Valin	Jessica Riel-Dery	Mélanie Guimont	Nick Desjardins
Jessica Charlebois	Simon Legros (I)	Olivier Lafond-Martel	Samuel Fortin (I)
Émile Pascal Boucher-L'Écuyer			

As we can see, the actors are not well-known, A-list Hollywood celebrities.

Now that we have our actors and movies, we have to create a bipartite graph linking actors to the movies they have acted in. There should be 12 movie nodes and 13 actor nodes. For each movie, we took the actors that were credited and added a movie-actor pair to a matrix to represent the edge between them. Then, we plotted the edge list and assigned all movie vertices to blue and all actor vertices to pink. The bipartite graph can be seen below in Figure 2.

Figure 2: Bipartite graph of a community of movies and its corresponding actors with size between 10 and 20



As we can see, some of the pink actor nodes are more centralized, and some actor nodes only have acting credit in one movie. The more centralized in the graph the actors are, the movie movies they would have acted in and the more important they would be to the structure of the community. Here are the top three most important actors in the bipartite graph by how many movies they acted in within the community, seen below in Table 12.

Table 12: Three most important actors in bipartite graph

Actor ID	Actor Name	Number of movies in the community
16970	Nick Desjardins	12
37120	Olivier Lafond-Martel	12
38585	Simon Legros (I)	12

The list confirms our suspicion that the more centralized nodes would be the most important. Each of the top three actors have starred in all 12 of the movies in the community, and without these actors, the community wouldn't be a connected, cohesive unit.

Since the community genre is Short in both problems 8.1 and 8.2, we suspect that these actors are friends who enjoy working together in a group most of the time making short French films. Therefore, this is a high correlation between these actors and the dominant genres we found previously in this problem.

3. Neighborhood analysis of movies

For this part, we downloaded the **movie_rating.txt** and looked at the relationship between the three following movies and similar movies' ratings:

- Batman v Superman: Dawn of Justice (2016); Rating: 6.6
- Mission: Impossible - Rogue Nation (2015); Rating: 7.4
- Minions (2015); Rating: 6.4

QUESTION 9: For each of the movies listed above, extract it's neighbors and plot the distribution of the available ratings of the movies in the neighborhood. Is the average rating of the movies in the neighborhood similar to the rating of the movie whose neighbors have been extracted? In this question, you should have 3 plots.

For each of the movies listed, we retrieved the movie's index ID, and then retrieved the neighbors of the node corresponding to that ID. For each ID of those neighbors, we retrieved the name of that movie and then used the name to look up the corresponding rating if the movie was rated. If it was, we added that to the cumulative ratings to be used in the histogram. We also summed up all the ratings available and divided it by the number of movies with ratings available to get the average rating. The number of neighbors and the average rating for each movie listed above can be seen below in Table 13.

Table 13: Movie number of neighbors and average rating of neighbors

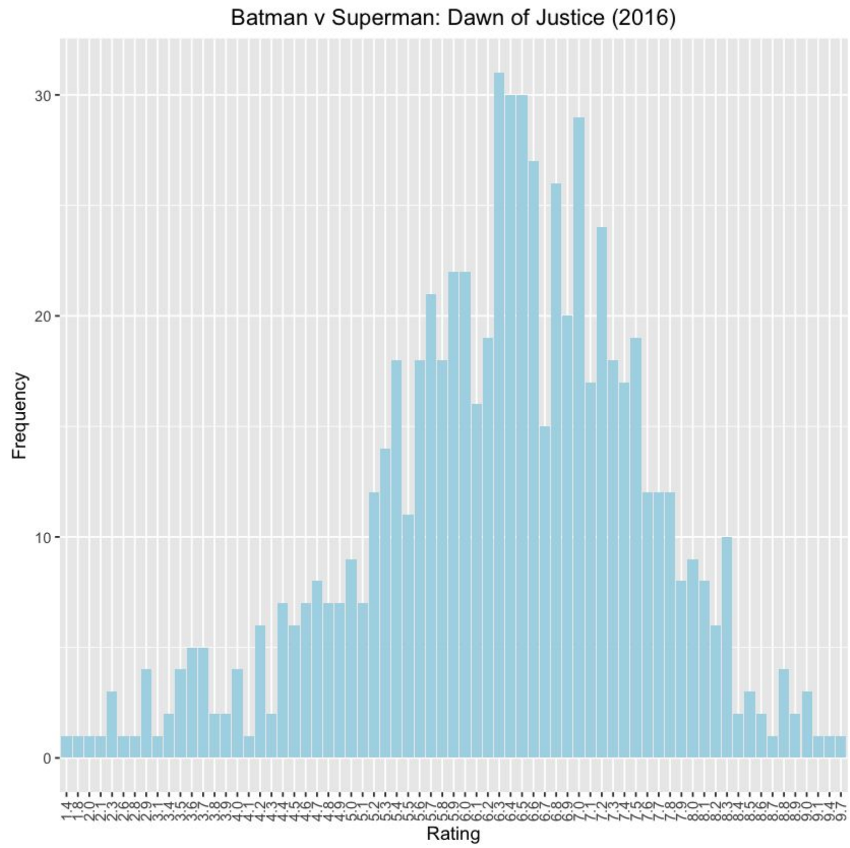
Movie Name	**Total Neighbors**	**Average Rating of Neighbors**	**Rating of Movie**
Batman v Superman: Dawn of Justice (2016)	860	6.3093	6.6
Mission Impossible - Rogue Nation (2015)	647	6.1938	7.4
Minions (2015)	656	6.8513	6.4

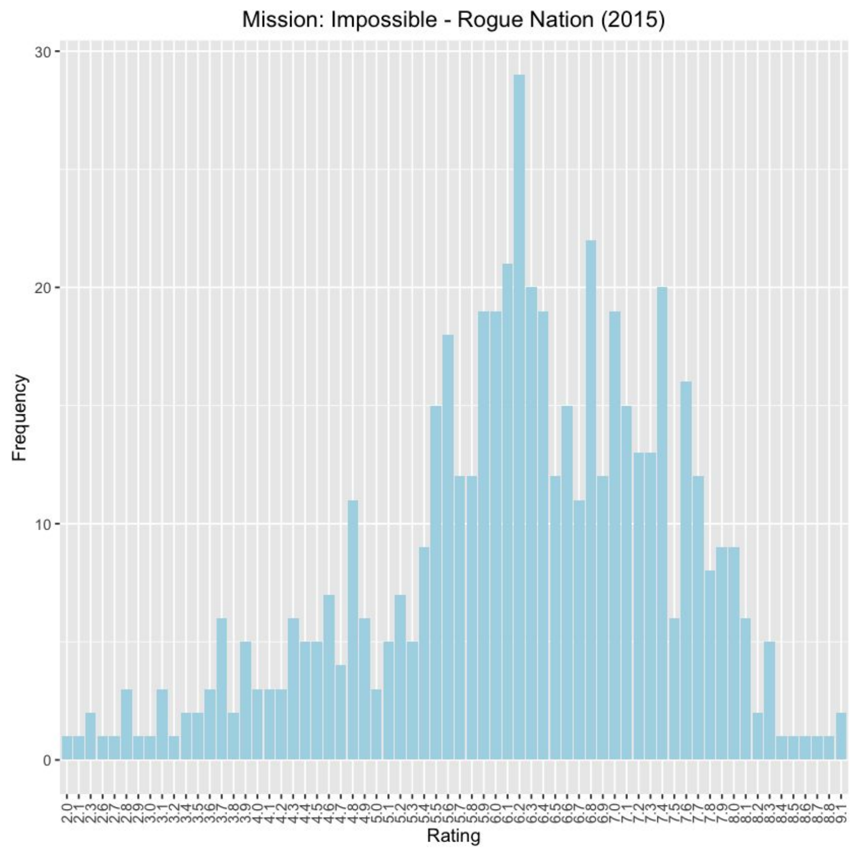
We would say that the average rating of neighbors is in the ballpark of the rating of the movie we are examining, although it is not close enough for an actual prediction. For "Batman v Superman: Dawn of Justice (2016)", the average rating of the neighbors was roughly 0.29 points lower than the movie's rating, which is reasonably close. For "Mission Impossible - Rogue Nation (2015)", the average rating of the neighbors was roughly 1.21 points lower than the movie's rating, which is a quite significant difference in our opinion. And finally, for "Minions (2015), the average rating of the neighbors was roughly 0.45 points higher than the movie's rating, which is a noticeable difference.

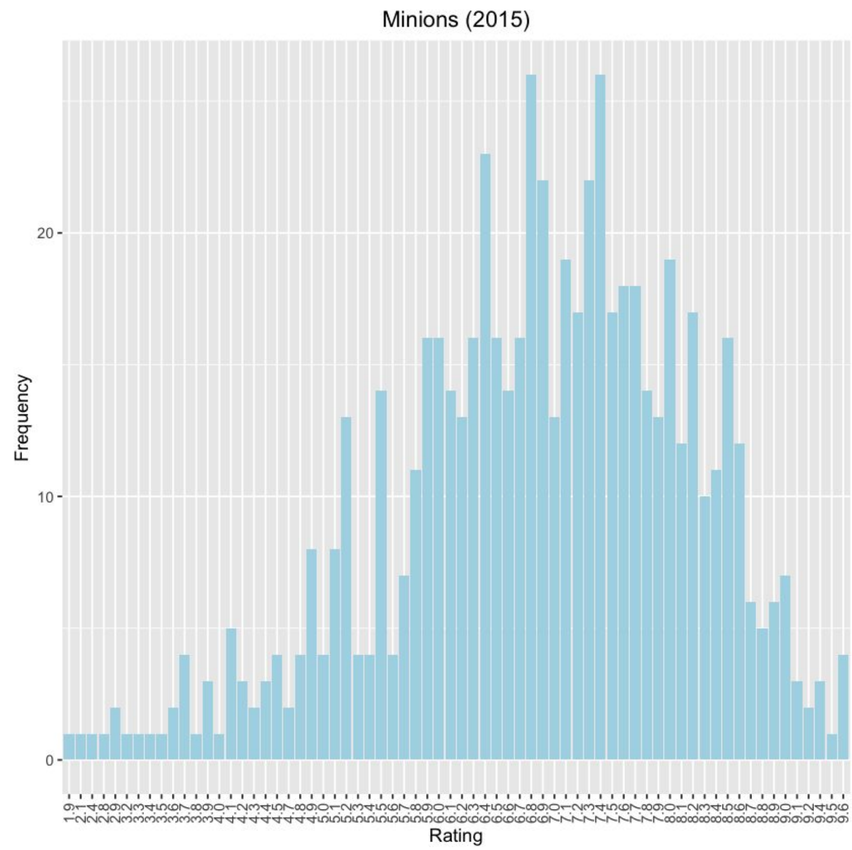
Next, we plotted the distribution of available ratings of the movies in the neighborhood of the three movies listed. The histograms can be seen below in Table 14.

Table 14: Distribution of available ratings of movies in the neighborhood of the movies listed

Movie Name	**Histogram of available ratings of movie's neighbors**

Batman v Superman: Dawn of Justice (2016)	

Mission Impossible - Rogue Nation (2015)	

Minions (2015)	

From the histograms, we can see that the highest frequency of ratings tends to be within a point or two of the original movie we are examining, and that all three distributions are generally centered and unimodal.

QUESTION 10: Repeat question 10, but now restrict the neighborhood to consist of movies from the same community. Is there a better match between the average rating of the movies in the restricted neighborhood and the rating of the movie whose neighbors have been extracted. In this question, you should have 3 plots.

We performed the same operations again, but this time we restricted the neighborhood to consist of movies from the same community by separating the graph into the same communities that we used in problem 7 and 8, and adding an additional check by retrieving the community of the movie node we are examining and seeing whether each neighbor node also belongs to that community.

The number of neighbors and the average rating for each movie listed above can be seen below in Table 15.

Table 15: Movie number of neighbors and average rating of neighbors with restricted neighborhoods

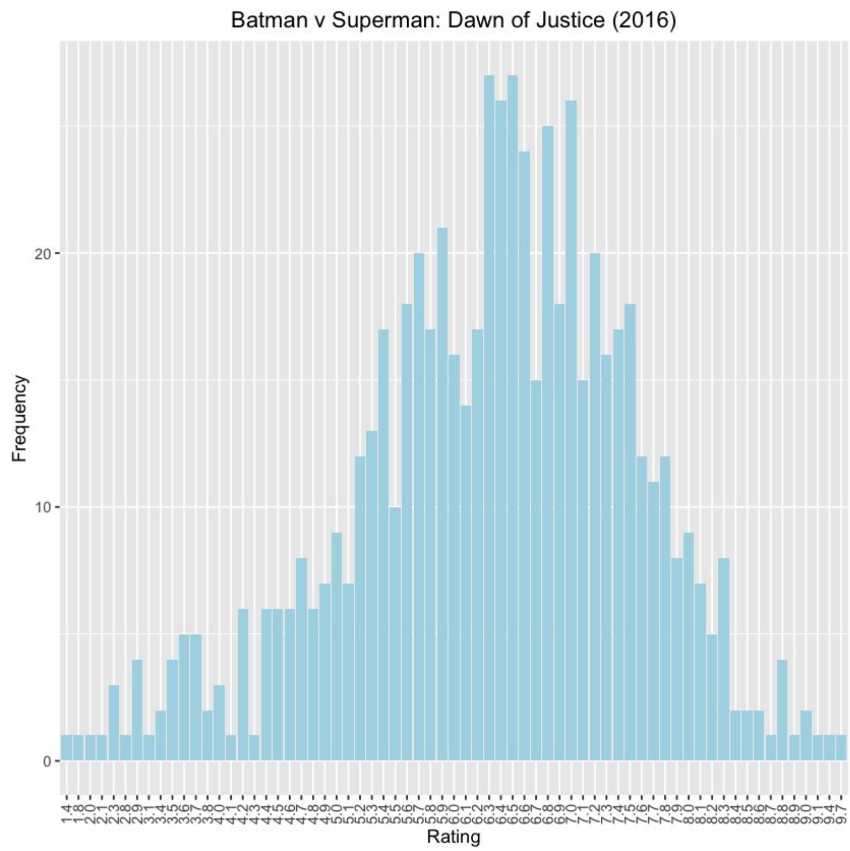
Movie Name	Total Neighbors	Average Rating of Neighbors	Rating of Movie
Batman v Superman: Dawn of Justice (2016)	635	6.3005	6.6
Mission Impossible - Rogue Nation (2015)	453	6.2148	7.4
Minions (2015)	557	6.8675	6.4

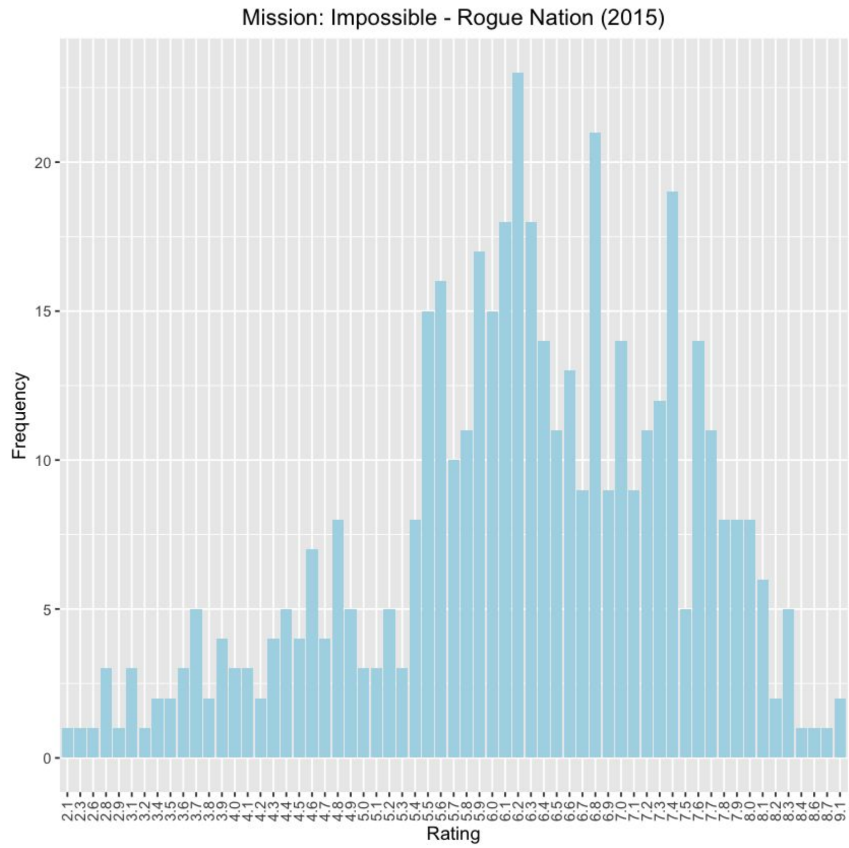
Examining the new results, would say that the average rating of neighbors is, again, in the ballpark of the rating of the movie we are examining, although it is not close enough for an actual prediction. In fact, the average rating of the neighbors, even after removing nodes that weren't in the same community, did not change very much at all. For "Batman v Superman: Dawn of Justice (2016)", the average rating of the neighbors was roughly 0.30 points lower than the movie's rating, which is reasonably close and just a tiny bit farther from the actual rating by an additional 0.01 points from the previous rating. For "Mission Impossible - Rogue Nation (2015)", the average rating of the neighbors was roughly 1.19 points lower than the movie's rating, which is a quite significant difference in our opinion, and only closer to the actual rating by about 0.02 points. And finally, for "Minions (2015), the average rating of the neighbors was roughly 0.45 points higher than the movie's rating, which is a noticeable difference and again, off from the previous average rating by 0.02 points and a tiny bit farther away from the actual rating.

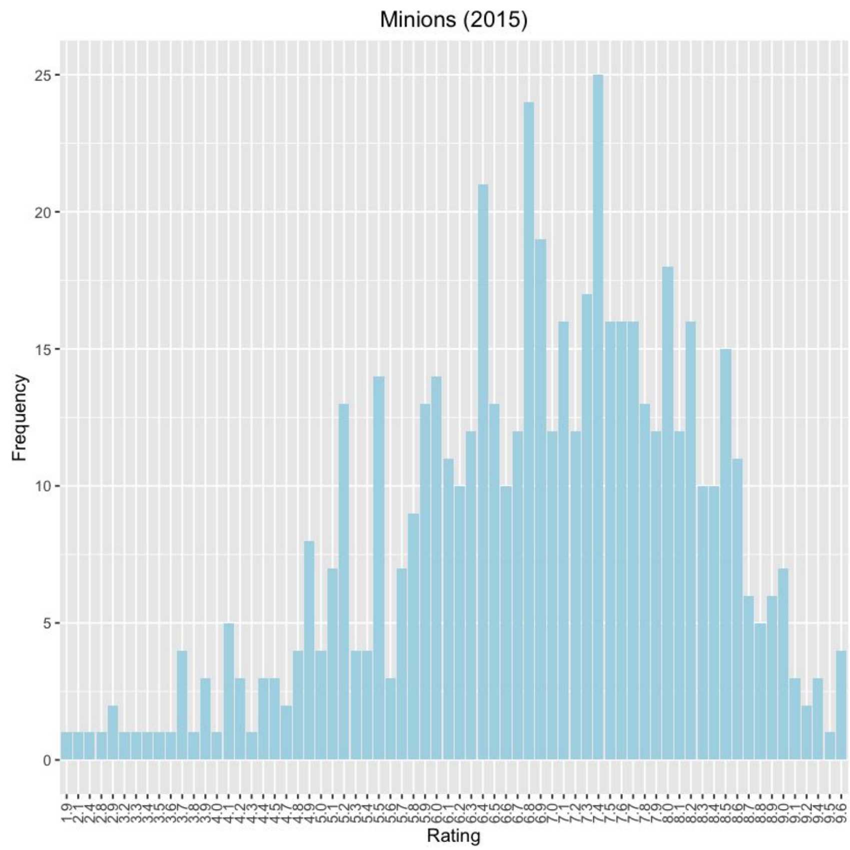
This makes sense because even though quite a few vertices were removed from the total neighbors that the average rating did not really change in a significant way. This suggests that the community that it belongs to does not really influence the average rating or the ability to predict a movie's rating all that much. We would conclude that there isn't a better match between the average rating of the movies in the restricted neighborhood and the rating of the movie whose neighbors have been extracted.

The histograms with the restricted neighborhood data can also be seen below in Table 16.

Table 16: Distribution of available ratings of movies in the neighborhood of the movies listed with restricted neighborhoods

Movie Name	Histogram of available ratings of movie's neighbors
Batman v Superman: Dawn of Justice (2016)	

Mission Impossible - Rogue Nation (2015)	

Minions (2015)	

As we can see, the distributions are extremely similar to that of question 9, showing that removing a few of the neighbor nodes did not significantly impact the distribution or the results.

QUESTION 11: For each of the movies listed above, extract it's top 5 neighbors and also report the community membership of the top 5 neighbors. In this question, the sorting is done based on the edge weights.

To extract the top five neighbors (top being the neighbors who have the heaviest common edge), for each movie we obtained the movie's ID and used it to find the neighbors, then for each of those neighbors, we got the edge between that node and our main movie node, capturing its weight. If that weight was heavier than the the least heavy weight of the five heaviest edge weights we had looked at, we would replace that weight. We repeated this process until we ended up with the five nodes with the heaviest edge weights. Then for each neighbor, we looked up that node's index to see what community it belonged to using a hashmap of the community IDs. The results for the three movies that we looked at can be seen below in Tables 17, 18, and 19.

Table 17: Top five neighbors for Batman v Superman: Dawn of Justice (2016)

Batman v Superman: Dawn of Justice (2016); total neighbors = 860		
Top	**Neighbor Movie Name**	**Community ID**
1	Eloise (2015)	1
2	The Justice League Part One (2017)	1
3	Into the Storm (2014)	1
4	Love and Honor (2013)	1
5	Man of Steel	1

Table 18: Top five neighbors for Mission: Impossible - Rogue Nation (2015)

Mission: Impossible - Rogue Nation (2015); total neighbors = 647		
Top	**Neighbor Movie Name**	**Community ID**
1	Fan (2015)	19
2	Phantom (2015)	19
3	Breaking the Bank (2014)	1
4	Suffragette (2015)	1
5	Now You See Me: The Second Act (2016)	1

Table 19: Top five neighbors for Minions (2015)

Minions (2015); total neighbors = 656		
Top	**Neighbor Movie Name**	**Community ID**
1	The Lorax (2012)	1
2	Inside Out (2015)	1
3	Despicable Me 2 (2013)	1
4	Horton Hears a Who! (2008)	1
5	Gake no eu no Ponyo (2008)	1

The results in the tables above for the most part make sense. The top movies tend to belong to the same genres and themes.

With regards to Batman v Superman: Dawn of Justice (2016), two of the movies are also superhero movies in the same cinematic universe. The other three movies are a psychological thriller, a natural disaster action movie, and a war romance movie. Although it doesn't seem like these movies should go together, they all feature young, Hollywood "it" stars that would be close in a network together and probably be cast in many of the same movies.

Impossible - Rogue Nation (2015) is also a bit difficult to pin down. Mission: Impossible itself is a Tom Cruise action-heavy movie, and its top neighbors include two Hindi movies, a comedy about banks, a British period piece, and a magician action-thriller movie, we can only guess that the plot in at least one of the movies requires going to different locations, hence the need for the overlap in actors.

The last movie, Minions (2015), is the easiest to understand the relationship to its neighbors. All the films are animated children's movies, so it would make sense that there would be a lot of voice actor overlap between the films.

Additionally, almost all of the movies belong to the same community, with the exception of only two, which belong together in a second community as neighbors related to Mission: Impossible.

4. Predicting ratings of movies

We also looked at the relationships of the ratings to help us predict the ratings of the following tree movies:

- Batman v Superman: Dawn of Justice (2016)
- Mission: Impossible - Rogue Nation (2015)
- Minions (2015)

QUESTION 12: Train a regression model to predict the ratings of movies: for the training set you can pick any subset of movies with available ratings as the target variables; you have to specify the exact feature set that you use to train the regression model and report the root mean squared error (RMSE). Now use this trained model to predict the ratings of the 3 movies listed above (which obviously should not be included in your training data).

To train a regression model to predict the movie ratings, we used the ratings of each movie's neighbor. If we are trying to predict the rating of movie "A", then we would take all the actors of movie "A" and generate a set of movies that all of these actors have performed in. These are considered to be the ratings of the neighbors of movie "A". We then took the list of ratings and extracted some statistics from the distribution. We did this in native Python by looking at each movie and finding the neighbors using a set of dictionaries mapping movies to cast to ratings etc.

Specifically, we looked at 5 features:

1) 25th percentile
2) 50th percentile (median)
3) 75th percentile
4) Mean rating
5) Standard deviation of ratings

The reason we chose these features was to capture the success of each movie's neighbor's ratings. Since it's possible that the mean can't sufficiently capture enough information, we also included other statistics like the 25th percentile and the 75th percentile. We chose to omit the 0th and 100th percentiles as they might introduce too much noise into the system. The standard deviation also gives a metric on how consistently the ratings cling to the mean. If a movie has a high mean but the standard deviation is large as well, perhaps its high mean was dragged up by several well performing outliers.

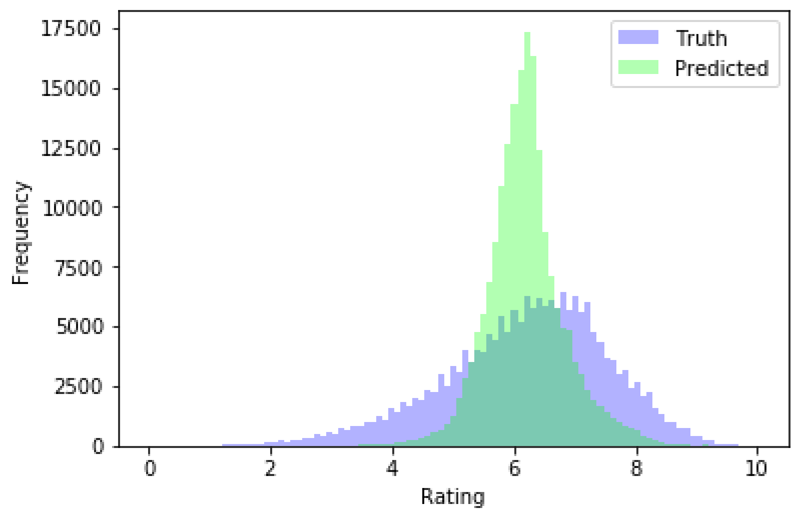
In order to get meaningful results for these features, we removed any movies that had less than 5 neighbor ratings. After extracting the features and running through sklearn's linear regression framework, we derived the following results:

Table 20: True vs. Predicted Ratings for movies

	True Rating	**Predicted Rating**
Batman v Superman: Dawn of Justice (2016)	6.6	6.5
Mission: Impossible - Rogue Nation (2015)	7.4	6.3
Minions (2015)	6.4	7.1

During training, we report an **RMSE value of 1.36211**. To gain more intuition about how the regression model is doing, we show overlaying histograms of the truth and predicted ratings:

Figure 3: Overlaying histograms of the truth and predicted ratings



From here, it appears that the linear model is reducing its loss by estimating most movies to have a rating within 5 and 7. Relatively few of the predictions have ratings at far ends of the true rating distribution.

While there are many degrees of freedom with regards to the success of a regression model, one likely factor is that the features are not convincingly meaningful. A movie can have some minor character, and the minor actor's movies are now taken into consideration just as much as neighbors that are linked to the original movie by a very prominent actor.

In hopes of addressing this issue, we repeat the regression exercise using the relationship between actors and movies, rather than movies and their neighbors. This is done as Question 13.

We also predicted the ratings of movies using a bipartite graph. In the graph *G = (V, E)*, we partition the vertex so that $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \varnothing$ and $e_{ij} = (v_i, v_j)$ where $v_i \in V_1$ and $v_j \in V_2$. In this graph, the vertices belonging to the same set are non-adjacent. We will create a graph that

- V_1 represents the set of actors/actresses
- V_2 represents the set of movies
- Edges e_{ij} between nodes if an actor i has acted in movie j

QUESTION 13: Create a bipartite graph following the procedure described above. Determine and justify a metric for assigning a weight to each actor. Then, predict the ratings of the 3 movies using the weights of the actors in the bipartite graph. Report the RMSE. Is this rating mechanism better than the one in Question 12? Justify your answer.

We represented a bipartite graph by creating a dictionary where each key is a movie title, and each value is a list of actors in that movie. This represents a bipartite graph where the nodes are actors/actresses and movies, and edges connect each movie to the actors/actresses that were in the movie. To calculate the weight of each actor/actress, we looked at all the movies that actor/actress performed in and averaged the ratings of those movies. So each weight in the graph represents a connection between cast and actor, which scales to how much success that actor has enjoyed in their other movies.

To create a feature vector we could use to pass into a regression model, we looked at each movie and identified all of their edges. We then computed statistics based on these edges (as we did with Question 12):

1) 25th percentile
2) 50th percentile (median)
3) 75th percentile
4) Mean rating
5) Standard deviation of ratings

Similarly to what we did in 12, for this analysis we removed any movies that had less than 5 edges. We ran sklearn's linear regression package on the data, and report the following results:

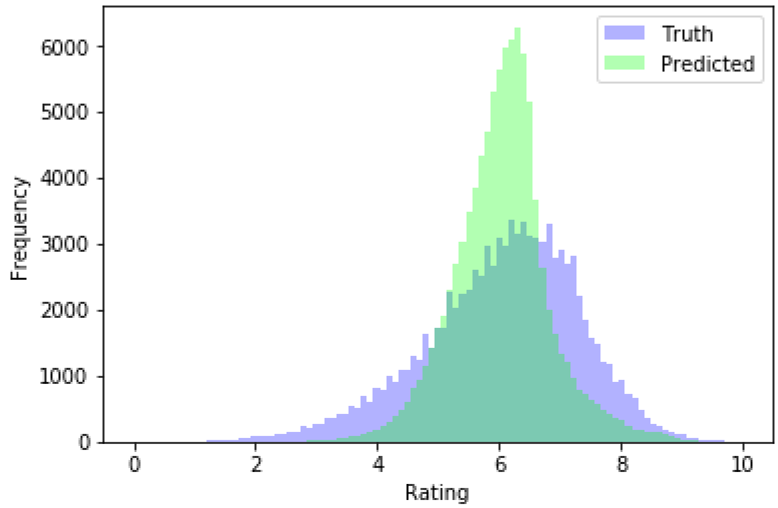
Table 21: True vs. Predicted Ratings for movies

	True Rating	**Predicted Rating**
Batman v Superman: Dawn of Justice (2016)	6.6	6.7
Mission: Impossible - Rogue Nation (2015)	7.4	6.7
Minions (2015)	6.4	7.3

The **training RMSE for this regression was 0.94**, which is an improvement from our results from Question 12.

To compare the distribution of the linear model's predictions to the distribution of the true ratings, we plot an overlapping histogram:

Figure 4: Overlaying histograms of the truth and predicted ratings



Compared to Q12, the results of the regression model trained with the bipartite graph adheres to the shape of the distribution more closely. There is still a tendency to over-predict values near the middle of the distribution, but the effect is less extreme than when using movie neighbors as features.

Based on the training RMSE and by comparing the distributions of the predictions, we find that using information specific to each actor is a better way to predict movie ratings than to look at a movie's neighbor's ratings. There could be cases where Movie "A" has a low rating, but has a minor character who happened by chance to be a part of a very successful movie.

These cases are more filtered out when we use the bipartite graph, because movies each movie's rating is predicted using a metric that measures the actor/actress' success. There could still be cases where a movie has a high-profile cast but is still awful. Some ways we could potentially expand on our analysis in the future is to include other features like the genre, the director's average rating, film budget, etc. But the improvement from using a movie's neighbor ratings to using a movie's actor weights seems quite apparent.