EE232E Project 2
Haixi Ni 404383125
Yu Zhang 904512554

problem 1

We use the cleaned up data to start this project, parsing the data accurately is not trivial, though. The situations we deal with includes: irregular separators, data redundancy, inconsistent movie names and so on. In general, we clean up the data set carefully for generating the network.

problem 2

To construct the desired weight directed graph G(V,E) from the cleaned up data, we use several hash maps to improve efficiency. We also use integers to represent movies and actors/actresses to reduce store and manipulate cost. In the graph,

$V = \{\text{all actors/actresses in list}\}$

$S_i = \{m \mid i \in V, m \text{ is a movie in which i has acted}\}$

$E = \{(i,j) \mid i,j \in V, S_i \cap S_j \neq \varphi\}$

and for each directed Edge $i \rightarrow j$, a weight is assigned as $|S_i \cap S_j| / |S_i|$.

Due to limited encoding/decoding support, some of the Latin movies may not be added into the graph successfully. However, the influence of these missing movies is trivial.

problem 3

After running the page rank algorithm on the network generated above, we get 10 actors/ actresses with highest scores. Their names are listed, ordered by scores, from highest to lowest.

Miller, Harold (I)
Phelps, Lee (I)
O'Connor, Frank (I)
Farnum, Franklyn
Sayre, Jeffrey
Sullivan, Charles (I)
Mower, Jack
Holmes, Stuart (I)
Hack, Herman
Lally, Mike (I)

Honestly, I do not recognize these names. These were all very old actors/actresses and have already passed away. And they all participated in a large amount of movies.

I then listed top 10 famous celebrities in my opinion and get their scores, which is shown below.

Hanks, Tom	0.0001083764
Cruise, Tom	8.565538e-05
DiCaprio, Leonardo	6.80696e-05
Streep, Meryl	8.306493e-05
Aniston, Jennifer	6.26287e-05
Hopkins, Anthony	0.0001084442
Roberts, Julia	9.07201e-06
Bullock, Sandra	6.245144e-05
Adams, Amy	7.103524e-05
Jolie, Angelina	5.592171e-05

To our surprise, the scores of famous celebrates are not very high compared to the max score (0.0003817219), but most of them are above average (2.378065e-05).
We could conclude that the page rank score of this network is not related to actors' fame. Those who spend the most time in casting and cooperating with other actors get the highest score.

Problem 4
Remove all movies with less than 5 actors/actresses on list. Because integers take much smaller memory space than strings, we use hash table to tag the actors/actresses and movies with integers (positive for actors and negative for actresses). And also hash table is utilized to map every movie with its actors/actresses. For each actor/actress, if there's at least one movie, we put the movie in the table and add the corresponding actor/actress to the table. Then we have a hash table with movies with actors/actresses like this:

`Double Tap (1997)`
[1] 6528 7687 11153 13375 14945 17457 22055 22352 24958 32563 42369
`Drushyam (2014)`
[1] 2207 6466 8792 19342 28187 30175 35517 40149 -571 -11885 -12331
`Double Team (1997)`
[1] 2954 3525 4293 9718 13036 14698 15231 26273 29487 29988 32148 32163 33952 34160 34649 39971 40311 41425 42914 -2948 -4634 -5866 -10525

which means the movie `Drushyam (2014)` has 11 actors/actress, which are represented by positive(actors) and negative(actresses) integers.
Then we delete movies with less than 5 actors/actresses. However, the time complexity is so big that it is impossible for us to obtain the result in such a relatively short time. So we remove all movies with less than 10 actors/actresses. Then we get nearly 50 thousand

movies. In order to construct a movie network according to the set of actors/actresses, the jaccard index of the actor sets of 2 movies is calculated as the number of intersect actors / union actors. Deleting those with no intersect actors, finally we get 45020 movies, with their actors/actresses, ratings and genres.

In order to accelerate calculation of every two movie pair's weight, "doParallel" algorithm is used here. We have 8 cores to do the work, which means for each movie, every the other 8 movies' weights are calculated simultaneously.
Finally we create the edgelist file of movie network in a format of a matrix. The first two columns are the movies, i.e. vertices in movie network, and the last column is the weight of edge between every two movies. Examples are like this:
"V1" "V2" "V3"
"1" 1 11936 0.95
"2" 11936 1 0.826
"3" 1 11937 0.1
"4" 11937 1 0.022
"5" 1 16416 0.9
"6" 16416 1 0.947
"7" 1 17840 0.65
"8" 17840 1 0.929
"9" 1 20577 0.05
"10" 20577 1 0.015

The first two columns of integers represent movies, and the last decimal numbers are the weight between the two. Now we have an undirected network of movies.

problem 5
We do a community search with Fast Greedy Newman algorithm for the network generated in the previous problem. And tag each community with genres that appear in 20% or more of the movies in the community. The community index, size and tagged genre are shown below.

We observe top 3 genres of one community, and we find that most tagged genres are "Drama". It is because the majority of the movie genres in the original data is "Drama". Some top 2 genres' frequencies are also above 20%, and these genres could also been used to tag a community. Thus a community can be tagged with one or two genres to make the legs meaningful, These genres are highlighted in the table.

	Size	top 1 genre & frequency	top 2 genre & frequency	top 3 genre & frequency
1	9062	**Drama 0.2178327080**	Thriller 0.1850584860	Romance 0.1438975944
2	7693	**Drama 0.2066813987**	Romance 0.1626153646	Comedy 0.1163395294
3	2950	**Drama 0.2322033898**	Thriller 0.1416949153	Romance 0.1335593220
4	1652	**Drama 0.3044794189**	Comedy 0.1573849879	Romance 0.1113801453
5	8205	**Drama 0.2777574650**	**Comedy 0.2469226082**	Romance 0.1076173065
6	1531	**Drama 0.2893533638**	Comedy 0.1528412802	Romance 0.1149575441
7	2	**Crime 0.5**	**Music 0.5**	
8	174	**Comedy 0.327586207**	**Drama 0.327586207**	Musical 0.063218391
9	2465	**Drama 0.3014198783**	Comedy 0.1756592292	Romance 0.1302231237
10	11	**Drama 0.45454545**	**Comedy 0.36363636**	Music 0.09090909
11	1488	**Drama 0.325940860**	Comedy 0.123655914	Romance 0.116935484
12	1281	**Drama 0.2794691647**	**Comedy 0.2092115535**	Romance 0.1046057767
13	1573	**Drama 0.2797202797**	**Comedy 0.2301335029**	Romance 0.1595677050
14	446	**Drama 0.266816143**	Comedy 0.165919283	War 0.130044843
15	2630	**Drama 0.2418250951**	**Romance 0.2250950570**	Thriller 0.1300380228
16	792	**Drama 0.271464646**	**Comedy 0.232323232**	Romance 0.101010101
17	472	**Drama 0.288135593**	Comedy 0.186440678	Romance 0.152542373
18	594	**Drama 0.311846690**	Romance 0.179442509	Comedy 0.151567944
19	931	**Drama 0.295381310**	Romance 0.169709989	Comedy 0.127819549
20	253	**Short 0.252964427**	Drama 0.185770751	Thriller 0.106719368
21	436	**Drama 0.277522936**	**Romance 0.236238532**	Comedy 0.116972477
22	389	**Comedy 0.383033419**	**Drama 0.223650386**	Romance 0.131105398
23	9	**Short 0.5555556**	**Drama 0.2222222**	Romance 0.1111111

problem 6

We add the 3 new movies into the graph and do the community search again. The result obtained are as follows.

We could see that all the new movies and their neighbors are belong to same community.

New Movie	Batman v Superman: Dawn of Justice (2016)	Community 3
Nearest Neighbors	Asylum (1972/I)	Community 3
	The Sisters (1938)	Community 3
	Tal para cual (1953)	Community 3
	Legend of the Red Reaper (2013)	Community 3
	Never Again (2001)	Community 3

New Movie	Mission: Impossible - Rogue Nation (2015)	Community 2
Nearest Neighbors	Asylum The Last Sunset (1961)	Community 2
	Dial 1119 (1950)	Community 2
	The Commissioner (1998)	Community 2
	Japan (2008)	Community 2
	Now That I Have You (2004)	Community 2

New Movie	Minions (Voice) (2015)	Community 2
Nearest Neighbors	Free and Easy (1941)	Community 2
	Lovers and Other Strangers (1970)	Community 2
	Here Comes Mr. Jordan (1941)	Community 2
	Secrets (1933)	Community 2
	Jalna (1935)	Community 2

problem 7

We use the 3 movies' neighbors to predict the rating. We obtain the rating of neighbors from the rating list and simply get the average. Since the new movies share common cast and genre with their neighbors, we believe that this method could roughly predict the ratings.

Batman v Superman: Dawn of Justice (2016) : 5.8
Mission: Impossible - Rogue Nation (2015) : 6.72
Minions (Voice) (2015): 6.92

Problem 8

First we calculated the intersection of all the movies in the rating list, genre list and the result from Problem 4. The movie name style s=is different in genre list and rating list

mainly differs in extra spaces. By adjusting the names we are able to calculate the intersections and got 45020 movies in total with rate and genre accordingly.
The features we used are:

1. 5 top pageranks of the actors: This is calculated in Problem3 and rescaled by timing 100000.
2. 101 boolean values of 100 top directors: By checking the data on Imbd, we got the namelist of the 100 top directors but when checking the movies they acted there were a lot of missing movies in the actor_movie.txt file. So finally after selection, 40 directors were chosen so 40 boolean values in total.
3. Mean genre score: This is the mean score of the movies with same genre. The difference can be as large as 1 rating point so we propose there is a strong correlation between genre and rating.

We used both linear regression model (lm) and neuron network model (nnt) to do the regression. Caret package is used for training models and tuning the parameters.
80% percent of the data are used for training while the left are used for testing. The data separation is done by createDataPartition() randomly. The result is followed:

Linear (lm) : R square = 0.712
Predicted rate for :

- Batman v Superman: Dawn of Justice (2016) : 8.2
- Mission: Impossible - Rogue Nation (2015) : 7.6
- Minions (2015) : 8.3

Neuron Network (nnt) : R square = 0.96
Predicted rate for :

- Batman v Superman: Dawn of Justice (2016) : 7.9
- Mission: Impossible - Rogue Nation (2015) : 7.8
- Minions (2015) : 8.5

Two results differ around 0.2.
The result from linear regression is acceptable. The most significant factor is the mean rate of the genre (t = 2.9) and the top 2 pagerank actors. (t = 2.1 and 0.9) The Boolean term of not directed by top director doesn't acted as important as expected. The reason might be the affection of missing a lot of informations of the top 100 directors. The main reason for low R square is because the limit numbers of parameters since the complexity of linear model is very limited. If we are able to have the full movie list of every top 100 director, the model will be improved.
The result from NNT is trustable since the dataset is relatively large. Because the data lost from the first three problems introduced from Latin encoding problem, the dataset scale is reduced and thus caused less training of the NNT. Also, deleting movies introduced by limiting the number of movies acted per actor (>15)and number of actors

per movie (>10) reduced the reliability of NNT. Given more computational power and time, the result will be better.