# EE 232E

# Project - II Report

*Chandan Dhal - 904588105*

*Sidharth Gulati - 104588717*

*Tushar Sudhakar Jee - 004589213*

# Introduction

In this project, we used a graph of IMDb movie networks. In the first part of the project, we created movie network based on actors and actresses to find out the rankings of the actors and actresses using PageRank algorithm. We also used the network to analyze the genre of the various movies in the network by mapping it with the communities structures. In the second part of the project, we created a network based on movie rating and defined a prediction model to predict the rating of a movie.

## ***Exercise 1***

*Preprocessing data*

In this exercise we used the following text files to create network.

- Actor_movies.txt
- Actresses_movies.txt

These file have entries for all the movies each actors and actresses have acted, voice or credited. The text files have tags stating whether the actor/actress has voiced or uncredited in the movie. Hence we removed such tags from the text file so that the file has only entries of actors/actresses and their movie. The objective of the exercise was to create a single file with movies entries of each actors and actresses who have acted in more than 4 movies. Hence we also removed the actors and actresses with less than 5 movies and merged both the text files.

## ***Exercise 2***

*Designing the actors/actresses network*

In this exercise we constructed a weighted graph $G(V,E)$ using the data set created in the previous exercise. The parameters of the graph are defined as follows:

- Vertex (V) has all the actors and actresses name
- $Si = \{m|i \in V$, m is a movie in which i has acted$\}$, i.e. set of movies an actor/actresses has acted.
- Edge (E) = $\{(i, j)|i, j \in V, Si \cap Sj \mathrel{!=} \varphi\}$, i.e. an edge is created if two actors (vertices) have at least one common movie.
- Weight (W) of each edge (i to j) was assigned as $W = |Si \cap Sj|/ |Si|$, i.e. ratio of the movies acted together by actors/actresses i and j to the movies acted by actor/actress i. Hence the graph is directed as each edge has different weight.

*Creating the actors/actresses network*

As the network is directed, hence we created a data frame for creating the network. Similar to the homework 3 problem set, the data frame was structured as three columns, the first column had the vertex from where the edge is originating and the second column where the edge is terminating and the third column had the weight of the edge. We used the following algorithm to efficiently process the large data frame and create the network:

```
1. Create a [MOVIE-TO-ACTOR] dictionary with 'movies' as the keys and 'set of actors who have acted in that movie' as the values.

2. For each movie in [MOVIE-TO-ACTOR]:
    2.1. Find combinations of all actors. This will give us nC2 combinations where n is the
         number of actors who have acted in that particular movie.

3. Now I create a [ACTOR-ACTOR] dictionary with '(actor-actor) combination' as the keys and the 'number of movies those two actors have acted together in' as the values.

4. For each movie in [MOVIE-TO-ACTOR]:
       For each (actor-actor) combination in the movie list:
           Initialize the key of [ACTOR-ACTOR] dictionary to 1 if first encounter.
           Add 1 to the key of [ACTOR-ACTOR] dictionary if movie already encountered.
```

We also calculated the graph density of the created network, using the the given formula below:

$$D = \frac{|E|}{|V|\,(|V| - 1)}$$

The various parameters of the network are tabulated as below:

| Parameters | Values |
|---|---|
| Connected | True |
| Nodes(actors/actress) | 243,911 |
| Edges | 53,655,030 |
| Graph density | 9.01e-4 |

We can infer the network created is very sparse as it has very low graph density.

***Exercise 3***

*PageRank Algorithm*

PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a network and its mostly used to determine the importance of the node in the network. The algorithm returns the probability distribution representing the likelihood of a random walker visiting a particular node in the network.
We ran the pagerank algorithm on the network created in above exercise. The pagerank score of node against node degree is plotted below:

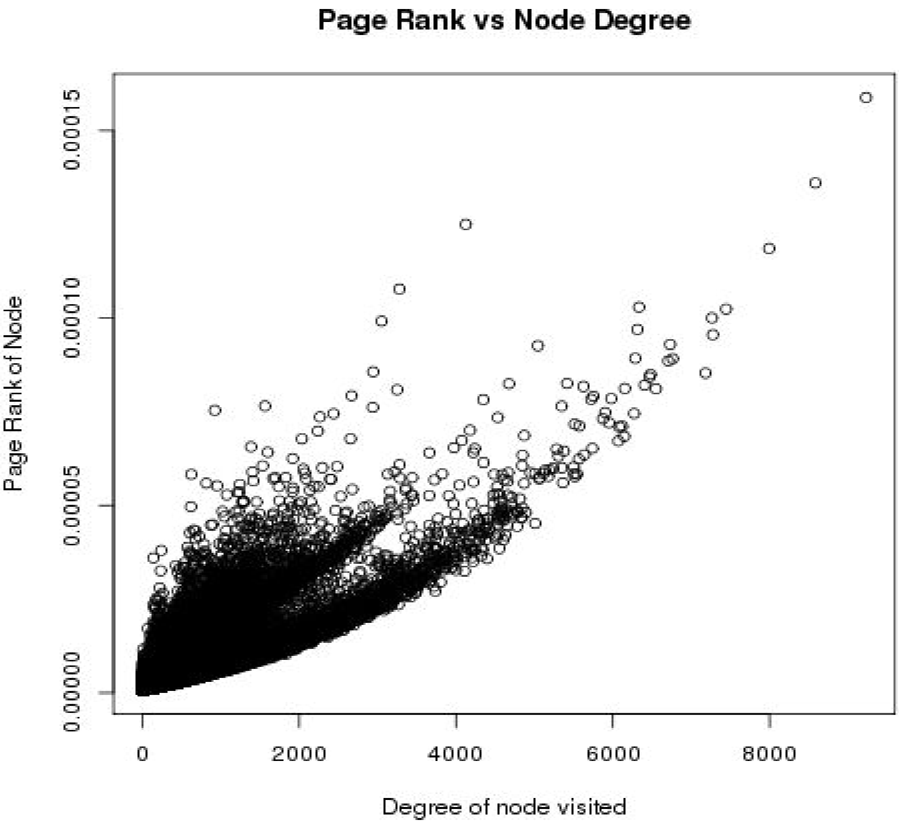


Fig 1:- PageRank Algorithm

From fig 1, we can infer that the node which is visited more has high PageRank score, i.e. node with high degree has higher PageRank score. Hence the plot appears linear as the PageRank is directly proportional to degree of the node. Intuitively in the created network, the actor/actress who have acted with different actor/actress has high PageRank score as their node degree will be high. The table below has the top 10 actors/actresses in the network defined by the PageRank score. The table also has description of the actor/actress and their IMDb page.

**Rank #1**
**Bess Flowers**

**Pagerank score - 1.58*10$^{-4}$**
Number of credits - 878
Number of movies - 828

**Rank #2**
**Sam Harris(II)**

**Pagerank score - 1.36*10$^{-4}$**
Number of credits - 681
Number of movies - 600

One of four people to have appeared in a record five Best Picture Oscar winners: All the King's Men, Around the World in Eighty Days, My Fair Lady, The Sound of Music, and You Can't Take It with You.

**Rank #3**

**Ron Jeremy**

**Pagerank score - 1.24*10$^{-4}$**

Number of credits - 1423

Number of movies - 637

Number one adult film star in the U.S during 1980s and 1990s. Also, directed many adult movies.

**Rank #4**

**Harold Miller**

**Pagerank score - 1.18*10$^{-4}$**

Number of credits - 620

Number of movies - 561

He was an actor, known for Souvenirs (1928), Tipped Off (1923) and Mountain Madness (1920).

**Rank #5**

**Fred Tatasciore**

**Pagerank score - 1.07*10$^{-4}$**

Number of credits - 542

Number of movies - 355

Fred Tatasciore was born in New York City, New York, USA. He is known for his work on Kung Fu Panda: Legends of Awesomeness (2011), 9 (2009) and Hulk and the Agents of S.M.A.S.H. (2013).

**Rank #6**

**Lee Phelps (I)**

**Pagerank score - 1.028*10$^{-4}$**

Number of credits - 661

Number of movies - 647

Lee Phelps was born on May 15, 1893 in Philadelphia, Pennsylvania, USA as Napoleon Bonaparte Kukuck. He was an actor, known for Anna Christie (1930), Limousine Life (1918) and War Dogs (1942). He was married to Mary Warren. He died on March 19, 1953 in Culver City, Los Angeles, California, USA.

**Rank #7**

### Jeffrey Sayre

**Pagerank score - 1.024*10$^{-4}$**

Number of credits - 510

Number of movies - 430

Jeffrey Sayre was born on December 3, 1900 in Illinois, USA. He was an actor, known for Major Difficulties (1938), Men of San Quentin (1942) and Mutiny in the Big House (1939). He was married to Lucille. He died on September 26, 1974 in Los Angeles, California, USA.

**Rank #8**

### Franklyn Farnum

**Pagerank score - 1*10$^{-4}$**

Number of credits - 608

Number of movies - 565

Boston-born Franklyn Farnum was on the vaudeville stage at the age of 12 and was featured in a number of theatre and musical productions by the time he entered silent films near the age of 40. He appeared to be at his most comfortable in the saddle, his career dominated mostly by westerns.

**Rank #9**

### Yuri Lowenthal

**Pagerank score - 0.992*10$^{-4}$**

Number of credits - 500

Number of movies - 318

Yuri is becoming well-known for his work in voice over in video games and animation, some of his roles include Superman on Legion of Super Heroes (2006), Sasuke on Naruto(2002) and The Prince in Prince of Persia: The Sands of Time (2003) video game series.

**Rank #10**

### Frank O’Connor (I)

**Pagerank score - 0.969*10$^{-4}$**

Number of credits - 633

Number of movies - 320

He was an actor in a supporting role and most of his work was uncredited. He is known for his role in The King of Kings.

***Observations***

From the bio of the actors/actresses in the above table we can infer that most of them have appeared in many movies. These actors/actresses have also appeared in movies as supporting role or voice actors and many movies have uncredited their roles too. However PageRank algorithm score depends on the degree of the nodes, hence these actors/actresses have very high score due to large number of movies they worked, as well as shared a lot of movies with other actors. We can also observe that most of the actors from early 1900, where actors/actresses used to act in lot of movies unlike today.

*PageRank of famous actors/actresses*

The table below depicts the PageRank score of the famous actors/actresses in our opinion.

| Actor/Actress | PageRank Score |
|---|---|
| Leonardo Dicaprio | 2.42e-5 |
| Robert De Niro | 6.02e-5 |
| Tom Hanks | 3.52e-5 |
| Robert Downey Jr. | 3.38e-5 |
| Meryl Streep | 3.36e-5 |
| Christian Bale | 3.02e-5 |
| Natalie Portman | 2.35e-5 |
| Mila Kunis | 1.64e-5 |
| Kate Winslet | 2.312e-5 |
| Tom Cruise | 3e-5 |

***Observation***

From the above  table we can infer that the famous actors/actresses in our opinion has very low PageRank score compared to the top 10 actors/actresses based on PageRank score. These actors/actresses have appeared in very less movies as lead role as well as in supporting roles. The top 10 actors/actresses have many appearance [support roles], as a result they have high node degree in the concerned network. These days actors/actresses tend to act in particular movie genre, which makes their appearances and number of co-stars very less compared to actors/actresses in early 1900s. Hence this anomaly is justifiable in a sense and we can observe that PageRank is not an effective measure for calculating actors/actresses popularity or judge for acting.

***Significant Pairing***

We analyzed the network for any significant pairing of actors/actresses who have acted together most of the time. Their pairing nodes (actors/actresses) have directed edges between them in both directions and has weight of 1. We found 440 such pair nodes (actors/actresses) and of such pair is tabulated below:

| Actor/Actress Pair | Description |
|---|---|
| Matthew Salinas - William Brush | They acted together in all the seasons of this comedy series Prank Calls. |
| Randi Brough - Candi Brough | Twin sisters and have acted only in movies together. |
| Alic Damir - Miro Mikanovic - Uros Obolnar | The Trio have acted together in the Bosnian and Herzegovinian movie series Suplje Price |
| Marlon Wessel - Leon Wessel Masannek | Short movie actors and have worked with each other on all their films. |
| Ben Bailey (IX) - Eric Muller (III) | They are german actors and siblings and have acted only in movies together. |

***Observations***

From the above table we can infer that pairings mostly take place in the following scenarios:

- Actors have only worked in a TV series or movie series for all seasons

- Smaller film industry (like Bosnia) where there are very few actors
- Actors only working in short films
- Actors who are siblings
- Bias of a director towards his/her children

***Exercise 4***

*Designing the movies network*

In this exercise we used the data text files referred in exercise 1 to create the movie network. We constructed a weighted graph G(V,E) using the data set created in the exercise 1. The parameters of the graph are defined as follows:

- Vertex (V) has all the movies name
- We only considered movies which has at least 5 actors/actresses
- Edge (E) = {(Mi, Mj)}, i.e. an edge is created if two movies (vertices) Mi and Mj which have A set of actors/actresses and B set of actors/actresses respectively.
- Weight (W) of each edge (i to j) was assigned as W = |A ∩ B|/ |A U B|, i.e. ratio of the set of actors/actress acted in movies Mi and Mj to the set of total actors/actresses of movies Mi and Mj. This ratio is known as jaccard index, which makes the graph simple and undirected.

*Creating the movie network*

As the network is undirected, hence we created a data frame for creating the network. Similar to the homework 3 problem set, the data frame was structured as three columns, the first column had the vertex from where the edge is originating and the second column where the edge is terminating and the third column had the weight of the edge.

We also calculated the graph density of the created network, using the the given formula below :

$$D = \frac{2|E|}{|V|\,(|V|-1)}$$

The various parameters of the network are tabulated as below:

| **Parameters** | **Values** |
|---|---|
| Connected | True |
| Nodes(Movies) | 370,344 |
| Edges | 71,376,537 |
| Graph density | 1.04e-4 |

We can infer the network created is very sparse as it has very low graph density.

***<u>Exercise 5</u>***

*Community Detection*

We analyzed the community structure of the movie network using fast greedy community detection algorithm. This community detection algorithm can find communities in simple undirected networks. It defines the communities in the network by optimizing a modularity score. Starting from a set of isolated nodes, the links of the original graph are iteratively added such to produce the largest possible increase of the modularity. Hence this community detection algorithm has the highest modularity index.
The algorithm found 22 communities in the movies network, the plot below depicts the sizes of each community. In this exercise we did not consider communities with less than 10 movies, as these movies are  isolated in the network

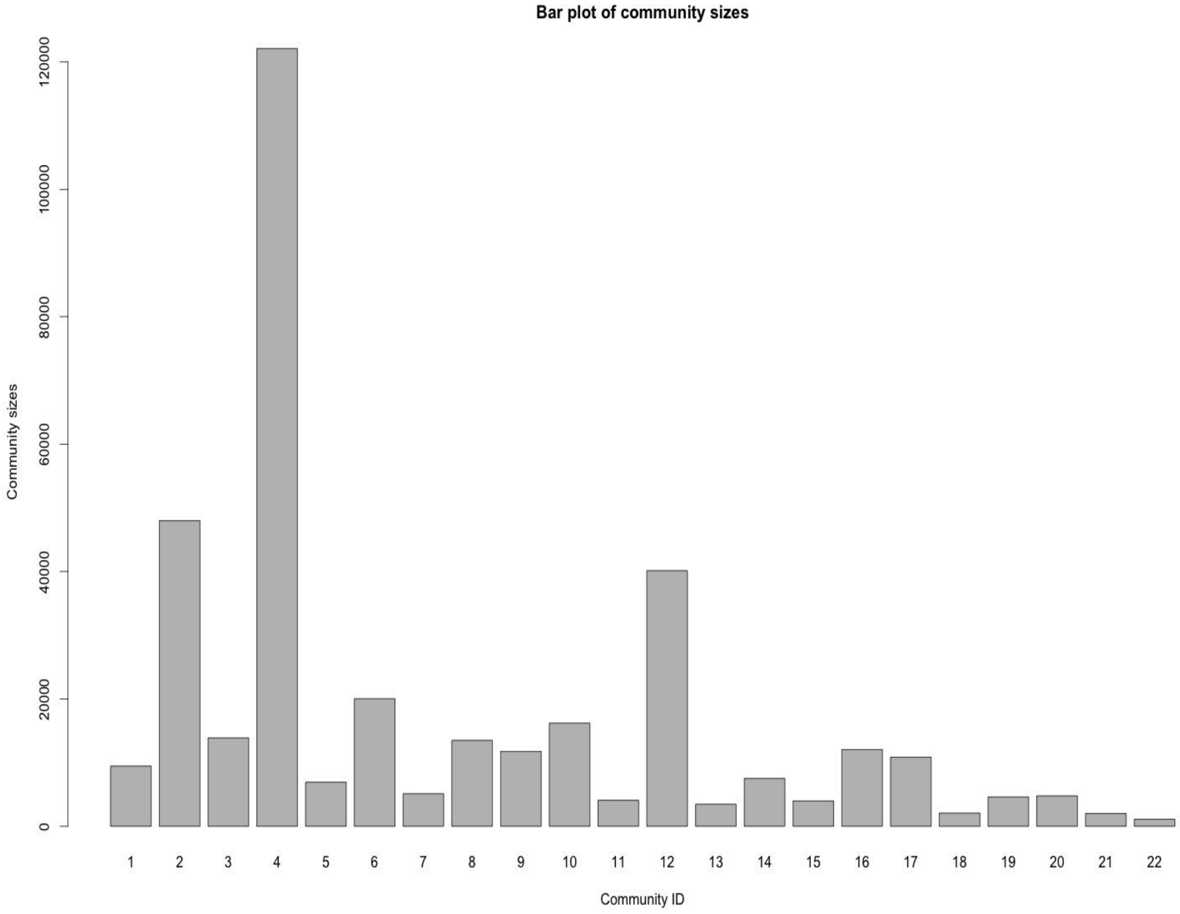


Fig 2. Bar plot for different community sizes

The sizes of each community is tabulated as below.

| Community ID | Community Size |
|---|---|
| 1 | 9454 |
| 2 | 47980 |
| 3 | 13883 |
| 4 | 122102 |
| 5 | 6925 |
| 6 | 20031 |

| | |
|---|---|
| 7 | 5113 |
| 8 | 13497 |
| 9 | 11761 |
| 10 | 16196 |
| 11 | 4088 |
| 12 | 40143 |
| 13 | 3469 |
| 14 | 7514 |
| 15 | 3982 |
| 16 | 12061 |
| 17 | 10859 |
| 18 | 2081 |
| 19 | 4604 |
| 20 | 4776 |
| 21 | 2013 |
| 22 | 1106 |

We can observe from the table and figure 2 that community 4 contains most numbers of movies. The modularity index for the partition is 0.8, which means although the network is sparsely connected but the movies inside the community is strongly connected.

*Genre tagging on communities*

We mapped each community with a genre, based on the movies present in the community. We found the genre of each movies in the community, if any genre is found in the 20% of the community size (total movies in the community) then we tag the particular genre to the community. The genre mapping of each 22 community is tabulated as below:

| Community ID | Genre Tag |
|---|---|
| 1 | Drama |
| 2 | Drama |
| 3 | Drama |
| 4 | Short |
| 5 | Drama |
| 6 | Romance |
| 7 | Drama |
| 8 | Thriller |
| 9 | Adult |
| 10 | Musical |
| 11 | Drama |
| 12 | Short |
| 13 | Short |
| 14 | Short |
| 15 | Drama |
| 16 | Drama |
| 17 | Drama |
| 18 | Short |
| 19 | Drama |
| 20 | Drama |
| 21 | Drama |
| 22 | Adventure |

*Observation*

From the above table, we can infer that the most of the communities have been tagged as Drama or Short. For better comprehension, we generate a similar bar plot as in figure 2 using the data file named movie_genre.txt. The figure below depicts the barplot for the genre datalist.

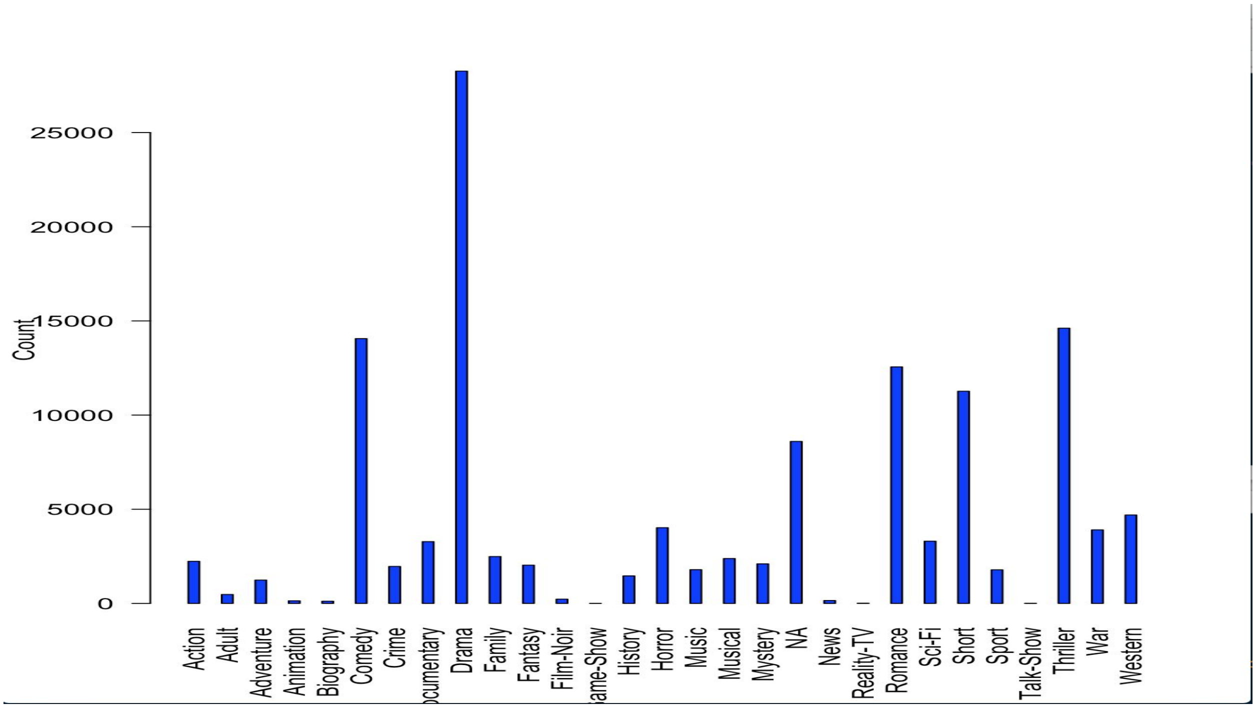


Fig 3: Barplot of the movie_genre dataset.

From fig 3, we can observe that most movie's genre in the dataset have been classified as Drama, followed by thriller. Hence most of our community was tagged as Drama during the community-genre tagging. Therefore, this method of tagging a community with genre based on 20% threshold has a bias towards the large genres and as a result the tagging relationships will not be useful for large communities. However, the tagging relationships will be more informative for smaller communities

***<u>Exercise 6</u>***

*Neighborhood analysis*

In this exercise we analyzed the network to find the neighborhood of a node(movie) in the movie network created in exercise 5. We found the neighborhood of the following movies and tabulated the top 5 neighbors of the movie. The rank of the neighborhood was defined in terms of the weight of the edge connecting the said node to its neighbor.

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

*Neighbors of Batman v Superman: Dawn of Justice (2016)*

| Neighborhood movies | Community Membership | Genre |
|---|---|---|
| Eloise(2015) | 4 | Thriller |
| Into the storm(2014) | 4 | Thriller |
| The end of the tour(2015) | 9 | Drama |
| The justice league part one (2017) | 4 | Sci-Fi |
| Grain (2015) | 4 | Drama |

*Neighbors of* Mission: Impossible - Rogue Nation (2015)

| Neighborhood movies | Community Membership | Genre |
|---|---|---|
| Fan(2015) | 10 | Short |
| Phantom(2015) | 10 | Thriller |
| The program(2015\|II) | 4 | Sport |
| Breaking the bank (2014) | 4 | Comedy |
| Legend (2015\|I) | 4 | Thriller |

*Neighbors of* Minions (2015)

| Neighborhood movies | Community Membership | Genre |
|---|---|---|
| The Lorax(2012) | 4 | Fantasy |
| Inside out(2015) | 4 | Reality-TV |

| | | |
|---|---|---|
| Surf's UP(2015) | 4 | Fantasy |
| Despicable me (2017) | 4 | Family |
| Up(2015) | 4 | Short |

***Observations***

The neighborhood of the movie Batman V Superman: Dawn of justice (2016) belongs to community 4 and we also observed that top movies of the neighborhood of the target movie is almostly from community 4 too. How ever some movie do not share the same genre , for example Batman V Superman: Dawn of justice (2016) is Sci-Fi , but Eloise is thriller. The reason such anomaly in the same community is because the movie network neighbor depends on common actors, hence same extras or supporting actors cast in different genre movies can result in neighbor nodes.

The higher the number of common actors the larger the weight of the neighbor. Since the extras contribute the most to the weights of the edges and they act in movies of all genre, as a result it makes sense to have neighbors with different genres.

Similarly for the movie, Minions (2015) we observed that most neighbors belong to community 4 same as Minions(2015). We can also infer that most movies in the neighborhood of minions are animated, hence the neighborhood analysis of the movie Minions(2015) makes more sense.

## ***Exercise 7***

*Prediction model using neighborhood*

We downloaded the rating list file of the movie to determine the rating of the following movies:

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

We defined a prediction model for predicting the rate of a movie using the neighborhood analysis. The algorithm for predicting the rating of a movie is summarized as below:

1. Extract the list of the neighbors of the target movie.
2. Sort the neighbor list according to the edge weights. Neighbors with highest edge weights. with the target movie comes first in the sorted list. Hence the sorted list is in descending order according to weights.

3. Select the first 20 movies from the sorted list and their corresponding ratings and weights.
4. Compute the weighted average of the ratings of these 20 movies
5. Return the weighted average as the predicted rating of the target movie.

The above algorithm was used to determine the ratings for the above three target movies and the result is tabulated as below:

| Movie | Predicted Rating | Actual Rating |
|---|---|---|
| Batman v Superman: Dawn of Justice (2016) | 6.1 | 7.1 |
| Mission: Impossible - Rogue Nation (2015) | 6.9 | 7.5 |
| Minions (2015) | 6.3 | 6.4 |

***Observation***

From the above table we observe that the rating for Minions(2015) is close to the actual rating. This observation is due to the fact that the neighbor analysis of Minions(2015) made more sense compared to others [Exercise 6]. For the other two movies, the neighborhood consists of movies which do not convey much information as they are completely different type of movies. As a result, the prediction function returns ratings which are very different from the actual ratings.

We can infer that this prediction mechanism is highly sensitive to the genre of the movies in the neighborhood and performs well only if the neighborhood consists of the movies of similar genre.

## ***Exercise 8***

*Predicting rating of a movie using regression model*

In this exercise we determine the rating of the following movies using linear regression model:

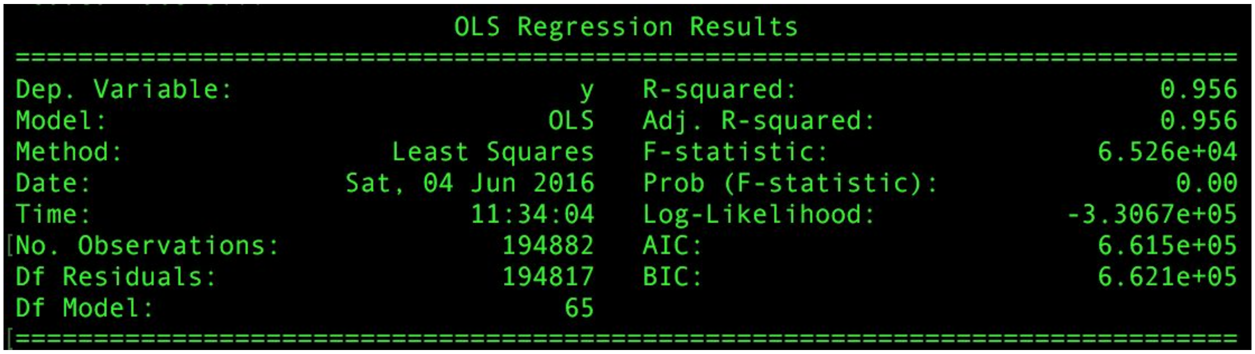
- Batman v Superman: Dawn of Justice (2016)
- Mission: Impossible - Rogue Nation (2015)
- Minions (2015)

We used the following algorithm to extract the features from movie corpus.

1. From the movie list, consider only the movies which have entries of more than five actors. (filter out movies with less than five actors).
2. Obtain the PageRank of all the actors who have acted in more than five movies (obtained from Question 2).
3. For each movie in the movie list:
    a. Obtain the top 5 PageRanks of actors who have acted in that movie. These will be the first 5 dimensions of the feature vector.
    b. Obtain the name of the director who directed this movie. Then create a 101 dimensional boolean vector of zeros. Each dimension of this vector corresponds to the rank of a top 100 movie. If the director (of this movie) has directed a movie in the top 100, then place a 1 in the index which corresponds to the rank of that movie. If the director (of this movie) has not directed any movies in the top 100, then place a 1 in the last index (101st) of the boolean vector. For example, if the current movie is Titanic and the director is Steven Spielberg, then we know that he has directed movies which are ranked 3, 28 and 31 in the top 100 IMDb ranking list. Hence the feature vector for Titanic will be the top 5 PageRank of actors in Titanic concatenated with a 101 dimensional boolean vector with 1 at positions 3,28 and 31.

4. Train a linear regression model (using the statsmodel.api or sklearn toolkit) in Python.

Now, in the testing phase, extract the features for the three movies (given above) for which we want to find the rating for, and obtain the predicted rating for these movies using the above trained regression model.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.956
Model:                            OLS   Adj. R-squared:                  0.956
Method:                 Least Squares   F-statistic:                 6.526e+04
Date:                Sat, 04 Jun 2016   Prob (F-statistic):               0.00
Time:                        11:34:04   Log-Likelihood:            -3.3067e+05
[No. Observations:             194882   AIC:                         6.615e+05
Df Residuals:                  194817   BIC:                         6.621e+05
Df Model:                          65
[==============================================================================
```

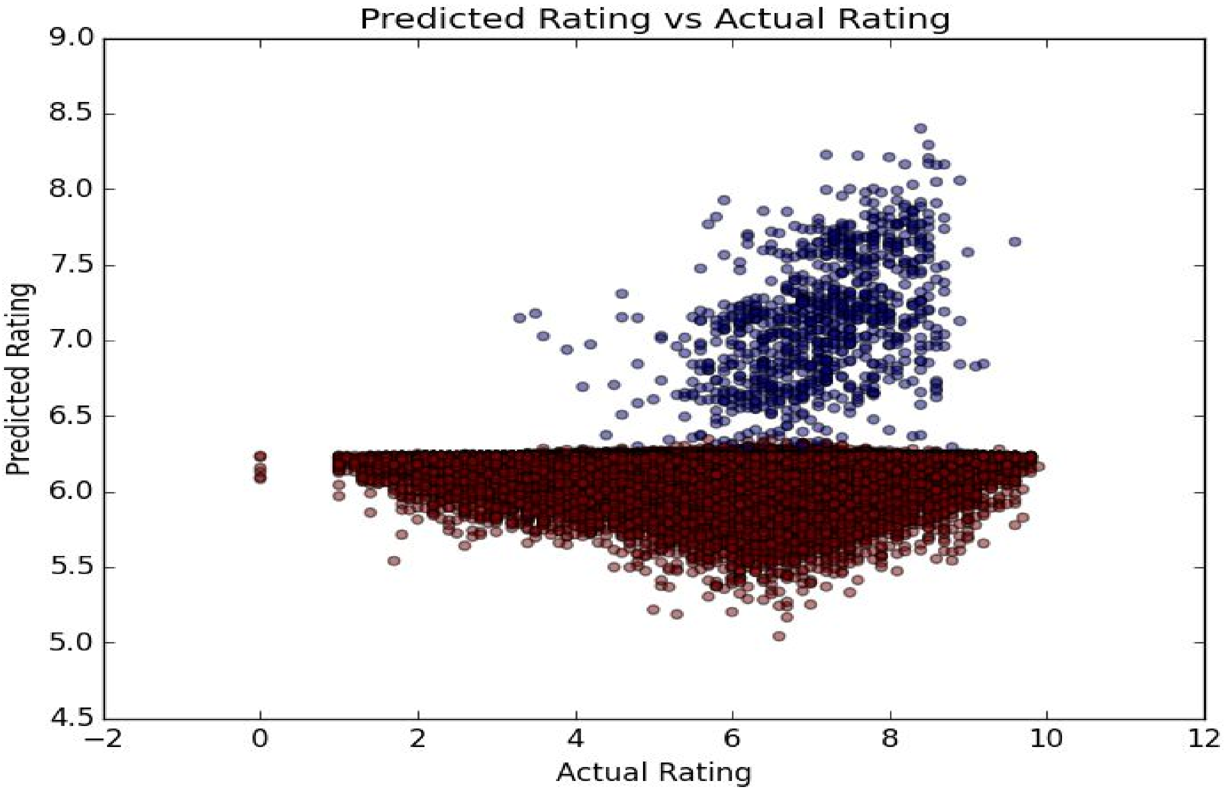
*Observation*

We got an r^2 value of around 0.956 for the regression model which indicates that almost all the variance in the data is explained by our model. Here are the predicted ratings for the three movies that we obtained:

| Movie | Predicted Rating | Actual Rating |
|---|---|---|
| Batman v Superman: Dawn of Justice (2016) | 6.20 | 7.1 |
| Mission: Impossible - Rogue Nation (2015) | 6.17 | 7.5 |
| Minions (2015) | 6.18 | 6.4 |

As we can see from the above table, the ratings for the three movies are almost the same and are not close to the actual IMDb ratings of these movies. What might be going wrong here? The reason the ratings are not accurate is because the feature space that we considered is not rich enough (or, in other words, is not directly correlated to the ratings of these movies). We feel that the top 5 PageRank scores of actors in the movies is clearly not a good feature. This is because, as seen from the previous sections, only the supporting actors who have acted in many movies have a high PageRank compared to lead or star actors. And the rating of a movie is hardly determined by these support actors, but rather by the lead actors in that movie. Since the first 5 dimensions of the feature vector is not rich enough to predict the ratings, we believe that the ratings get predicted entirely by the director (next 101 dimensions of the feature vector).

In order to analyse the effect of the features, and to graphically give an indication that the rating is entirely predicted by the directors alone, we plotted a graph of the actual rating vs the predicted rating. Also, we gave a color label of BLUE to the movies whose director has directed movies present in the top 100, and a color label of RED to the movies whose director does not have any movies in the top 100.



As we can clearly see from the above graph, a linear trend is observed ONLY for movies directed by top directors but the ratings for all the other movies directed by lesser directors are around 6. This is because, we are giving the first 100 dimensions of the boolean vector to the top directors but only a single dimension (101st) contributes to the variance if a director does not have a top 100 movie (as all dimensions 1-100 are zero) . Hence, the variance for these movies cannot be explained accurately enough with this feature space thus leading to constant incorrect predictions for these movies. And since all the three movies in our test set: Batman vs Superman: Dawn of Justice (2016), Mission Impossible: Rogue Nation (2015), and Minions (2015) are from directors who **DO NOT** have 100 top movies, the predicted ratings we get for these movies is somewhat same close to around 6.2.

Other possible features that might be good to explore:

Having analysed the problem thoroughly, we believe that a better feature space could have led to better predictions. For example, instead of looking at the PageRank of the actors in the movie, we could have a given a weighted measure to the role played by the actors and then taken the top 5 values from this measure. For example, we know that the PageRank of a lead actor, say, Leonardo Di Caprio is 2.69 * 10^-5, which is very less. But even though Leonardo Di Caprio has acted in only 37 movies, he has acted in the lead role in almost of those movies (around 30 movies) and he clearly has an impact on the rating of the movie. So if we give a **measure weighted by the number and type of role each actor has acted in**, this would give an “impact score” for each actor which would result in a better rating prediction. For example, now Leonardo Di Caprio, having actor in lead roles in more than 75% of his movies, would have a higher score, compared to an actor who has acted in higher number of movies but only as a supporting actor. We could also look at exploring the **genre** of movies as possible features as we observed that movies of the same genre had nearly similar ratings.

***Exercise 9***

*Bipartite Graph*

In this section we try to predict the movie ratings from a bipartite graph between movies and actors. A bipartite graph is a graph whose vertices can be divided into two disjoint sets U and V where U and V are each independent sets such that every edge connects a vertex in U to one in V. In this project, U is the set of actors and V is the set of movies.

We describe the algorithm we devised to predict the movie ratings from the bipartite graph below

Rating prediction Algorithm:

- For each actor we compute Score as below:
    - ❏ Suppose actor $A_1$ acted in movies $m_1$,$m_2$….$m_n$ with ratings $r_1$,$r_2$,$r_3$….$r_n$.
    - ❏ Thus the Score for actor $A_1$ is $\boldsymbol{Score(A_1) = (\sum_{i=1}^{n} r_i\,) / n}$
- We then create an actor set for each movie including all actors who have acted in it.
- The ratings are predicted in the following manner:
    - ❏ Suppose movie m1 has actor set consisting of $A_1$,$A_2$….An with scores of $s_1$,$s_2$….$s_n$. We then evaluate the rating as:

$$\boldsymbol{Rating(m_1) = (\sum_{i=1}^{n} s_i\,) / n}$$

We ran the above algorithm on the bipartite graph of movies and actors. The predicted ratings for the 3 movies are tabulated below

| Movie name | Predicted rating | Actual rating |
|---|---|---|
| Batman v Superman: Dawn of Justice (2016) | 6.8 | 7.1 |
| Mission: Impossible - Rogue Nation (2015) | 6.7 | 7.5 |
| Minions (2015) | 6.2 | 6.4 |

From the above table it can be seen that this algorithm yields much better results than the previous two rating prediction algorithms. This algorithm performs better because it uses actor scoring to predict movie ratings. The actor scoring mechanism filters out the irrelevant information from the prediction mechanism and as a result the predictions are better.

## ***Conclusion***

In this project, we analyzed the IMDb database to create various network corresponding to actors/actresses and movies network. In the first part we generated required data frame using the text files to generate the network. We used PageRank algorithm on actors/actresses network to determine the PageRank score [popularity] of each actor/actress who has acted in more than 5 movies. During the PageRank exercise we concluded that PageRank is not an effective measure as it takes into considerations the number of appearances, i.e. the more the movies the actor/actress acted, the higher the PageRank score[popularity].

Similarly, we created a network for movies and analyzed community detection algorithms to tag various community with genre tag. We designed a prediction model for predicting the ratings of a target movie using its neighbourhood analysis. Lastly we also designed a regression model to predict the rating for the same target movies.