Can professional networks predict success in Hollywood?

Final Project MACS 30123, Keertana Chidambaram

Every year, thousands of budding artists move to Hollywood with hopes of being rich and famous. Movies like La La Land confirm that the allure of this career path is as strong as ever. But the industry is known to be very tough and competitive. Most aspiring artists fail and only very few 'make it'. Moreover there is a common perception that connections and 'knowing the right crowd' is more important than actual talent to pervail in Hollywood. In this study, I try to understand the effect of professional network on career outcomes. More specifically I seperately look at the predictive power of the actual quality of an artist's movies versus the characteristics of co-artists she is associated with professionally on her career outcomes like volume of work, and continuity.

To answer this question, I turn to a dataset of 3,892 movies produced in Hollywood with major releases in the US between 2000 and 2018. This data has been collected and consolidated from sources like `imdb.com`, `omdb.com`, `rotten tomatoes`, and `thenumbers.com`. The dataset is only a small subset of all the available data on movies because this study is only a pilot. Once the right processes, parallelization architechture, and pipelines have been setup (which is scope for this final project), I plan to scrape and use a larger dataset to get more robust results.

`data_collection.py` contains python codes for data cleaning and the automated part of collecting the data (there was also some manual collection/data scraping that was carried out that I will skip from this document). Most of the data is collected via the `omdb.com` API. This is also the single most time consuming step in data collection. But it is also an embarassingly parallel workload because each API call is independent of one another. Gathering 1000 data points took about 201.01s in the serial version of the code run locally (`omdb_api_call()` in `data_collection.py`). The same process took only 5.2s when I ran the MPI code on midway using 5 cores (`data_collection_MPI.py`) (speed up = 38.7x).

Each row in the dataset is a movie along with the following information: movie title, main actors, writer(s), director(s), genre, year of release, run time, awards, ratings (imdb, metascore, rotten tomatoes, metacritic), bechdel score, production budget, and box office collection. For the purpose of analysis, I turn the dataset into a graph where each node is either an artist (actor/director/writer) or a movie. Artists are connected to the movies they have worked in as an actor/director/writer and indirectly with other industry artists through these movies. There is a total of 11,673 edges and 18,566 nodes in this graph. This information is stored as a pyspark graphframe for further analysis.

The outcome variables of interest are measures of professional opportunity: number of movies worked in (work volume) and years active (career continuity). One set of features represent the talent of the artist: ratings & awards (quality), bechdel score (diversity), and box office collection (popularity) for associated movies. Another set characterizes her professional connections: number of co-artist connections (connection volume), centrality of co-artist connections, prominency of production company and production budget raised (connection quality).

`analysis.ipynb` contains all the codes for feature engineering as well as model building. For this, I have extensively use graph algorithms and spark dataframe methods. There are two reasons for using graph algorithms: (1) in some cases, graph algorithms are more efficient and readable compared to regular algorithms, and (2) indices calculated using graph algorithms like page rank give additional insight into the data's network characteristics. Many graph algorithms are computationally intensive and

hard to scale. Therefore, there is a strong case for using pyspark in this project. The top graph algorithms that I have used are: (1) `aggregateMessages`, (2) `pageRank`, and (3) `connectedComponents`. `aggregateMessages` is an efficient way of aggregating neighbor characteristics; `pageRank` helps measure the importance of each vertex in the graph; and `connectedComponents` helps divide the graph into its constituent connected components.

To give an example for why graph algorithms are very efficient, consider the calculation of mean ratings for all movies an artist has worked in. First, we would need to filter out all the movies she has worked in, and then fetch its rating to finally calculate the mean. Here, there are many repetitive steps because the ratings for most movies are being fetched many number of times. Alternately, while using `aggregateMessages`, all the 'movie' nodes pass its ratings into all the 'artist' nodes directly connected to it. Each 'artist' node then aggregates the many messages it receives to get the mean rating. In the first case, the complexity of computation is of the order mn where m and n represent the number of movies and actors respectively. Where as for `aggregateMessages`, the complexity is only approximately $m + n$, which is much quicker than the former.

Finally, I am using machine learning to understand whether network measures or merit measures are better at predicting career outcomes. Clearly all the input and output variables are numerical values - float or int. Therefore, after reviewing all the models discussed in DataCamp, I decided to try out both linear regression and tree-based random forest regression. In both cases, the network variables are stronger at predicting career success as compared to merit-based variables. One limitation for this study is the dataset size. Now that the entire pipeline has been set up, I plan to put together more data to get more robust results. Another limitation is a chicken-or-egg paradox. On one hand, better network position rewards an artist with more opportunity and career success. In the same way, it could be that career outcomes also helps an artist to build a stronger network. When one variable predicts the other, it is difficult to understand this feedback dynamic. This calls for other types of models. But as far as preliminary studies like this one are concerned, the results establish that network characteristics do play a significant role in making Hollywood careers.