Movies_Analysis

February 20, 2018

1 Movie Analysis

```
In [1]: import pandas as pd
        import numpy as np
        import networkx as nx
```

1.1 Abstract

Cinema has become one of the highest profiting industries over the past century. The total box office revenue in North America alone amounted to $11.38 billion in 2016. With the possibility of great success, there is also a large risk of financial failure. This data exploration is motivated by answering the question what makes a movie successful. There is plenty of quantative data available for movies, such as the movies' budget, the release date, ratings etc., but in this analysis an attempt will be made to quantify movie information that is less measurable and then predict movie success.

1.2 Introduction

Research has been done to determine what aspects of a movie make it more successful; however, much of this research is contradictory. The research paper "Early Predictions of Movie Success: the Who, What, and When of Profitability" states movies with a motion picture content rating 'R' will likely have lower profits, whereas the research paper "What Makes A Great Movie?" states a motion picture content rating 'R' will have higher a box-office. Both papers analyzed thousands of movies, but came to opposite conclusions. Some variables used to predict movie success in these studies, included budget, motion picture content rating, and actor popularity.

Based on these previous models, the dataset used will include movie title length, run-time, motion picture content rating, director, genre, release date, actors, an actor popularity score, average salary of the actors in the movie, budget, and opening weekend box-office revenue for predictor variables. The actor popularity score will be calculated from a network of actors connected through the movies they appeared in together and from the average actor income. Movie success will be determined by the awards it is nominated for, the awards it won, the MetaCritic score, the IMDb rating, and the profit of the movie.

1.3 Data Scraped, Downloaded, Cleaned & Engineered

1.3.1 Beginning Dataset

A beginning dataset is downloaded from IMDb with 10,000 movies, each entry containing the movie title, URL on IMDb rating, run-time, Year, Genres, Num Votes, Release Date, Directors. From this dataset, additional information on the movie budget, gross income, opening weekend box revenue, actors, Oscar nominations, Oscars won, other award nominations, other awards won, MetaCritic score, and content rating is scraped and cleaned. The data points will be collected from IMDb, which is a reputable source for information, according to their website,

> "we [IMDb] actively gather information from and verify items with studios and filmmakers".

1.3.2 Cleaning Data

After gathering each data point, the data set is complete, although the information is not clean or uniform. The first step to clean the data will be to remove all commas across each column in the DataFrame. Removing commas will make it easier to convert monetary amounts to ints. Next each date in the Release column will be changed to a pandas date object, which will simplify any calculations that rely on the release date of the movie. Each monetary amount will be converted into an int and converted into USD. Each unique genre will be made into a column, with a true or false boolean for each movie entry. ### Feature Engineering To resolve the disagreement in monetary amounts, due to inflation, a dataset containing the CPI for each year from 1914 will used to adjust the monetary amounts. The CPI, Consumer Price Index, describes the amount of purchasing power the average consumer has. The length of the movie title will be added, and a NetworkX graph of all actors will be made. This network will connect nodes of actors to each other, if they appear in a movie together. The edges of the network will be weighted by the amount of movies the actors appear in together. An actor popularity score will be calculated from the actors appearing in the movie, based on how many other movies they appear in with other actors and the actor's income.

The total variables in the new dataset are movie title, title length, motion picture content rating, run-time, IMDb rating, genres, MetaCritic score, Oscar nominations, Oscar wins, other award nominations, other award wins, director, release date, budget, opening weekend, gross, profit, budget adjusted for inflation, opening weekend adjusted for inflation, gross adjusted for inflation, profit adjusted for inflation, the top ten actors in the movie, and actor popularity score. A separate network will hold the actor nodes and their connections.

```
In [3]: # Ideally in this cell we will have the cleaned/engineered dataframe to display... but u
        df = pd.read_csv("Result_Data/first_2000_engineered.csv",encoding = "ISO-8859-1")
        del df["Unnamed: 0"]
        print(df.columns)
        df[["Actor_0","Actor_1","Actor_2","Actor_3","Actor_4","Actor_5","Actor_6","Actor_7","Act
               'Gross', 'IMDb Rating', 'Meta Score', 'Num Votes', 'Opening Weekend',
               'Oscar Nominations', 'Oscar Wins', 'Other Nominations', 'Other Wins',
               'Release Date', 'Runtime (mins)', 'Title', 'Year', 'Decade', 'Profit', 'Budget_Ad
               'Gross_Adjusted', 'Open_Adjusted', 'Profit_Adjusted']].sample(2)
```

```
Index(['Actor_0', 'Actor_1', 'Actor_2', 'Actor_3', 'Actor_4', 'Actor_5',
       'Actor_6', 'Actor_7', 'Actor_8', 'Actor_9', 'Budget', 'Directors',
       'Gross', 'IMDb Rating', 'Meta Score', 'Num Votes', 'Opening Weekend',
       'Oscar Nominations', 'Oscar Wins', 'Other Nominations', 'Other Wins',
       'Release Date', 'Runtime (mins)', 'Title', 'Year', 'Genre: Comedy',
       'Genre:  Music', 'Genre: Fantasy', 'Genre:  Western', 'Genre: Action',
       'Genre: Drama', 'Genre:  Fantasy', 'Genre: Musical', 'Genre:  Comedy',
       'Genre: Mystery', 'Genre:  Horror', 'Genre:  Film-Noir',
       'Genre:  Adventure', 'Genre: Horror', 'Genre:  Crime',
       'Genre:  Musical', 'Genre: Romance', 'Genre:  Sport', 'Genre: Western',
       'Genre:  Mystery', 'Genre: Biography', 'Genre:  Family',
       'Genre:  Drama', 'Genre:  Thriller', 'Genre: Animation',
       'Genre:  Action', 'Genre:  History', 'Genre:  Romance',
       'Genre:  Sci-Fi', 'Genre: Film-Noir', 'Genre: Crime',
       'Genre:  Biography', 'Genre:  War', 'Genre: Sci-Fi', 'Genre: Adventure',
       'Genre: Family', 'Content Rating: PG-13', 'Content Rating: APPROVED',
       'Content Rating: PASSED', 'Content Rating: R', 'Content Rating: NR',
       'Content Rating: G', 'Content Rating: UNRATED', 'Content Rating: GP',
       'Content Rating: NOT RATED', 'Content Rating: NC-17',
       'Content Rating: PG', 'Decade', 'Profit', 'Budget_Adjusted',
       'Gross_Adjusted', 'Open_Adjusted', 'Profit_Adjusted'],
      dtype='object')
```

```
Out[3]:              Actor_0       Actor_1        Actor_2            Actor_3  \
        1335      Ron Howard  Peter Morgan   Peter Morgan     Frank Langella
        860   Robert Luketic  Amanda Brown  Karen McCullah  Reese Witherspoon

                    Actor_4      Actor_5            Actor_6        Actor_7  \
        1335  Michael Sheen  Kevin Bacon     Frank Langella  Michael Sheen
        860     Luke Wilson  Selma Blair  Reese Witherspoon    Luke Wilson

                   Actor_8        Actor_9       ...         Release Date  \
        1335  Sam Rockwell    Kevin Bacon       ...           2008-10-15
        860    Selma Blair  Matthew Davis       ...           2001-06-26

             Runtime (mins)          Title        Year  Decade       Profit  \
        1335          122.0     Frost/Nixon  2008-01-01    2000   -6406844.0
        860            96.0  Legally Blonde  2001-01-01    2000  123774679.0

              Budget_Adjusted  Gross_Adjusted  Open_Adjusted  Profit_Adjusted
        1335     2.864293e+07    2.130250e+07            NaN    -7.340431e+06
        860      2.507554e+07    1.975042e+08            NaN     1.724287e+08

        [2 rows x 31 columns]
```

```
In [4]: # Code for actor tree would go here, maybe a diagram too
        # Check out plotly for networkx
        rows,columns = df.shape
        nx_graph = nx.Graph()

        actors = df.as_matrix(columns=["Actor_0","Actor_1","Actor_2","Actor_3","Actor_4","Actor_

        for i in range(len(actors)):
            actors_in_movie = actors[i,:]
            for j in range(len(actors_in_movie)):
                if j == 9:
                    if (actors_in_movie[j],actors_in_movie[0]) not in nx_graph.edges():
                        nx_graph.add_edge(actors_in_movie[j],actors_in_movie[0],weight=1)
                    else:
                        nx_graph.add_edge(actors_in_movie[j],actors_in_movie[0],weight=nx_graph.
                else:
                    if (actors_in_movie[j],actors_in_movie[j+1]) not in nx_graph.edges():
                        nx_graph.add_edge(actors_in_movie[j],actors_in_movie[j+1],weight=1)
                    else:
                        nx_graph.add_edge(actors_in_movie[j],actors_in_movie[j+1],weight=nx_grap
```

```
In [5]: # networkx.Graph.degree   weighted number of edges  https://networkx.github.io/documenta

        weights = []
        for edge in nx_graph.edges():
            weights.append(nx_graph.get_edge_data(edge[0],edge[1])['weight'])

        set(sorted(weights)[::-1])
```

```
Out[5]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 14}
```

```
In [37]: nx_graph.remove_node(np.nan)
         print(nx_graph.number_of_nodes())
```

```
6772
```

```
In [11]: actors_in_graph = list(nx_graph.nodes())
         actors_in_graph = [str(i) for i in actors_in_graph]
         p = sorted(actors_in_graph)
         for i,ugh in enumerate(p):
             if i%500 == 0:
                 print(ugh)
```

```
50 Cent
Audrey Hepburn
Charles Martin Smith
David Kelly
F. Murray Abraham
Horton Foote
Jeremy Northam
Jukka Hiltunen
```

```
Linda Larkin
Michael Dougherty
Paul Feig
Robert Schwentke
Stephen Boyd
Victor Gojcaj
```

```
In [12]: nx_graph.size(weight='weight')

Out[12]: 18281.0

In [ ]: # Do some feature engineering with new actor dataset

In [ ]: # Talk about machine learning methods, why there are good and suitable for our data

In [1]: # And now do some machine learning....

In [2]: # Analyze ML models

In [ ]: # Case study to see how well we predict on a specific movie

In [ ]: # Insert some amazing conclusion here
```