CSE519 Project Progress Report

Which movies endure and Why?

Objective:

To build a model that predicts the current popularity of a movie as a function of variables known at its time of release.

Introduction:

The IMDb Dataset probably is the most comprehensive Internet Movie DataBase of information related to films including crew,cast,vote count, average rating, genre.The TMDb dataset obtained from Kaggle provides us with the popularity score of the movies which helps us to understand why some movies are more popular over others. Also, it provides us with the keywords and taglines for each particular movie allowing us to broaden our understanding of how certain plots have stood the test of time.OMDb dataset was obtained using the OMDb API which gives us details such as awards, Box office revenue, ratings from websites such as Internet Movie Database, Rotten Tomatoes , IMDb and Metacritic.

Analyzing deeper into the popularity of the movie, we strongly believe that the awards data from Wikidata and Kaggle would give us interesting insights about whether the most popular movies win Oscars and if Oscars affect the popularity of the movies henceforth. The data here contains the Academy award nominees and winners from 1927 until today for various categories offered. Therefore, analysis for the Oscars shouldn't be a problem.

Google Trends API offers a layer for Google Trends Data, which was used to retrieve keyword(movie) popularity over the years starting from 2004.

Dataset:

The dataset used was generated from :

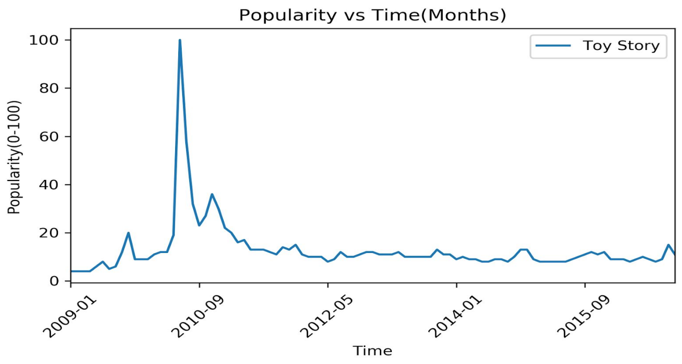
- IMDb dataset
- IMDb metadata
- OMDb API
- TMDb data
- Awards from Wikidata querying service
- Kaggle awards Dataset
- Google Trends API
- BoxOfficeMojo

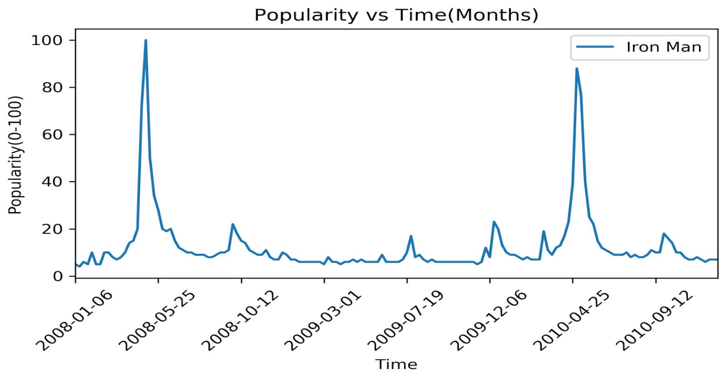
The data retrieved from IMDb , OMDb , TMDb, Wikidata and Kaggle awards Dataset were merged using the imbd_id column. The final dataset had columns such as movie title, actor, budget, genre , revenue, awards, keywords, vote count ,vote average, movie type, runtime. For the baseline model, a Linear Regression model and Random Forest Regressor was built using genre, budget, vote_count, vote_average, revenue and awards as the features.For training the models, we used the TMDb data that has the popularity column. The merged data that was obtained from other sources were used for testing the models.

Data Preprocessing:

As mentioned earlier, we used various sources to gather the data.While merging the data, we came across several issues and resolved them.Some of them are:

- In the IMDb dataset , whitespace was present at the end of the string in the movie title column. Because of this, we were unable to merge this dataset with OMDb ,TMDb and awards.The str.rstrip( ) method was used to remove the whitespace.
- There was a name mismatch in the movie title across all datasets. For eg, Birdman was named as 'Birdman or An unexpected Virtue of Ignorance' in one of the datasets.Due to this mismatch, some of the data was lost when merging.
- OMDb dataset had unicode characters present in the languages column that had to be encoded .It also had new line characters that had to be stripped.
- In the Kaggle awards dataset instead of movie name , publisher name was present .Also, if the film won an award for cinematography ,the name of the person was given under the name of film and the film name was given under the name of actor.For few entries, the name of the actor and the name of the film were interchanged.This dataset also did not provide us with director name. Since, this resulted in a loss of important data we queried the Wikidata to obtain the award information.
- In the TMDb credits dataset , the columns cast and crew were in json format.Same was the case with genres, keywords, production_companies and production_countries columns that were present in the TMDb movies dataset.To extract this data and store it as a string with commas being used as the separator we had to write down a function. json.loads( ) method was used to load this data.
- The OMDb API has a free limit of 1000 queries per day per API key .So, this data was obtained over a period of days for all the movie titles that were present in the IMDb dataset.
- When we tried to scrape data from Wikidata ,it retrieved a lot of valid movie titles but some of them were missing although we could find them on Wikidata.Then we figured out that this was because certain entities are instances of sub-categories of film.So we had to use ?item wdt:P31/wdt:P279* wd:Q11424 . in the query to navigate the 'instance of ' the chain until you hit upon 'film'.With the previous query that we were using, say an animated film like Toy Story ,which is a subclass of a film, wouldn't be returned.
- Data obtained from OMDb API were in json format. And when this was converted to a data frame all the values were stored as one column.We had to extract this data and store it in separate columns in the dataset.
- GoogleTrends API had a timeout issue because the limit for the day had been reached.
- The numeric columns had values from another column leading to a data type mismatch.The datatype of many such columns had to be changed manually.
- The old movies were rated 1-4 while the current movies are rated between 1-10 therefore, we had to normalize this data.



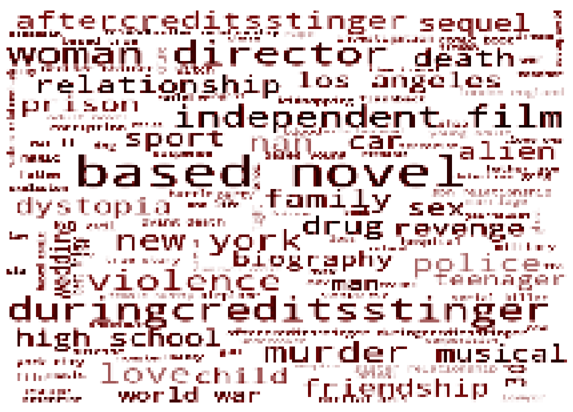


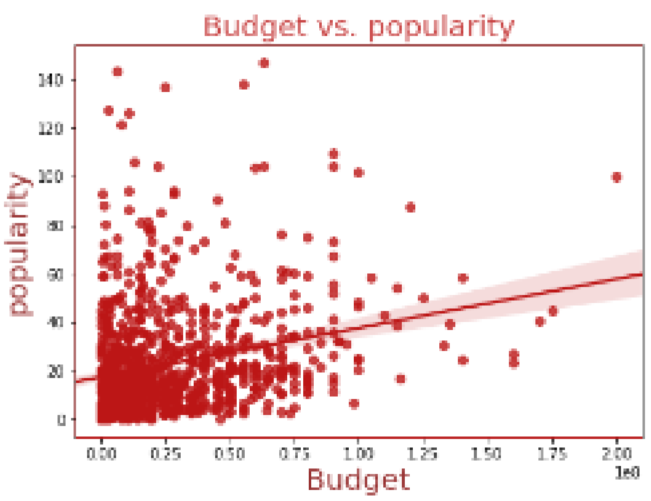
Here, We plot the popularity of two movies a function of time since their release to observe any trend in the popularity. For this, we used the google trends data to generate the number of times the keyword was searched for. We notice that for the movie hits peak popularity at the time of release and falls of gradually in the following months until some reason comes up that makes it relevant again, such as the release of a sequel, or the lead actor in the movie winning an oscar/having an exceptional performance. As we see in the case where we track the popularity of iron man over time, the popularity rises at the time of release and falls off in the following months and again rises at the time of release of its sequel, which correlates with the fact that people catch up on the movies in the franchise(prequel).

Features:

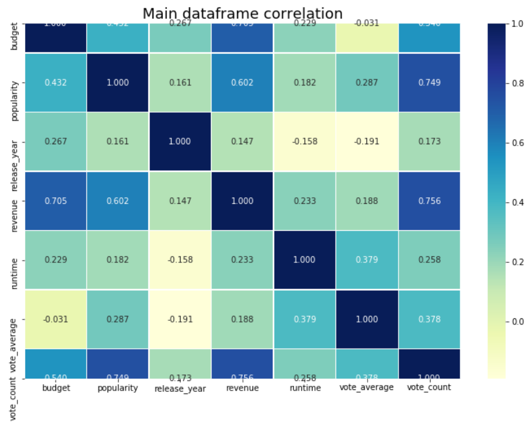
MovieId : This was a very important feature in the data processing stage as it was paramount in integrating various datasets.
Title : The title of the movie.
Budget : The total Budget which is important for predicting current popularity based on the time of release.
Revenue : This gives the popularity at the time of release using which we built a baseline model.
Year: The year in which the song was released.
Genre : The most important genre of the movie was taken into account separating the given genres.
Keywords : The keywords were considered for exploratory data analysis.
Popularity : The popularity metric in the TMDb(of 5000 movies) makes it easier for us to build a model and predict the popularity of the movies not in the TMDb.
Release Year : The year of release of the movie.
Vote_count & Average_Rating: The total number of votes and the average rating for a movie as given by IMDb.

Exploratory Data Analysis:(Popularity)





Also, from the keywords obtained from the Imdb_metadata, we plotted a wordcloud to understand what kind of plots have stood the test of time. As there has been a tremendous growth in the number of movies produced after 1990, the word cloud is slightly biased. Here, we can see that *novel* is one such important keyword indicating the growing popularity of the adaptations from books like the 'Lord of the Rings.'



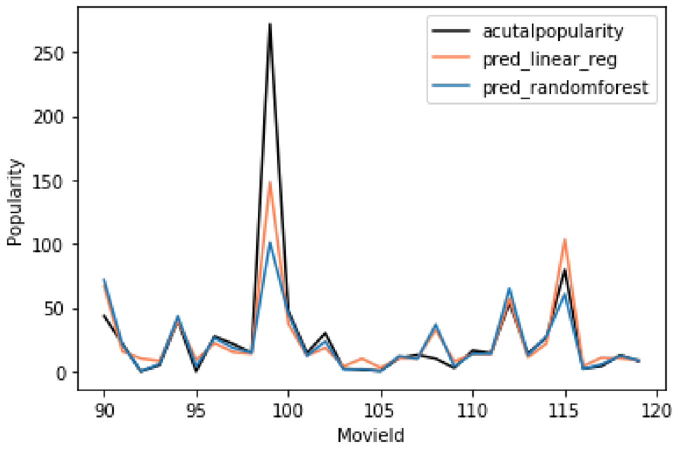
The plot 'Budget' vs 'popularity' has been plotted for movies which were released before 1990 to understand the popularity trend avoiding the huge budget movies since the beginning of 2000. The plot shows that the popularity of the movie isn't dependant on the budget of the movies.

As the popularity remains an important feature, we have plotted the heatmap where the correlation between some of the numerical variables have been taken into account. We perceive that the popularity of a movie is highly correlated to the vote_count, the revenue generated. Also, unlike the popular opinion, the popularity doesn't hold much correlation with the average rating and bears lesser correlation with the budget.

Approach:

We model our data using linear regression. We take a set of ten features which are based on the production of the movie to predict the popularity, which we decide using the correlation matrix that correlate well with popularity. We convert the categorical values to numerical values for genre, cast and director. We use L2 regularization(Lasso Regression) to penalize the features with large coefficients. It shrinks the coefficients of the less important features to 0, and in our model we notice that the production features such as director and cast do not affect the prediction. Through k cross validation we notice that the lambda with value 1 gives least root mean square error.

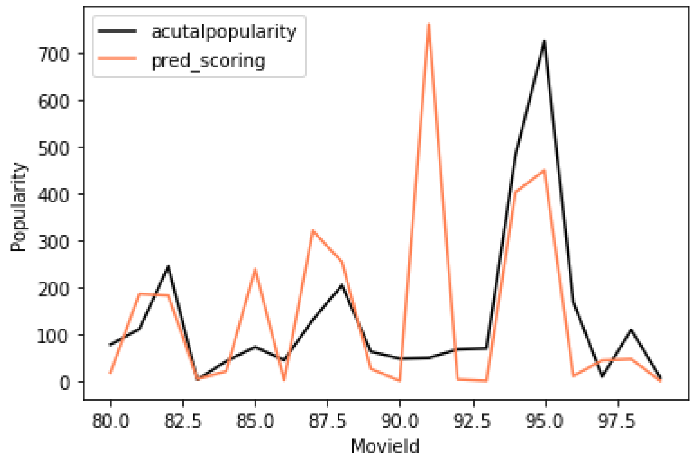
We also use random forest regression to model the data. In this approach we make use of bagging, which trains each decision tree on a different data sample, where sampling is done without replacement. We combine multiple decision trees, to predict a single popularity. We choose appropriate values for the hyperparameters max depth of tree and number of trees in the forest. We use K cross validation again to tune the hyperparameters and we obtain max depth of tree as 5 and number of trees as 100, and use mean absolute error as the error metric to measure how well the model performs.



Also, we compared the predicted popularity of the movies, by random forest and the linear regression, with the actual popularity as obtained from the TMDb dataset. By using the Train_test_split, we trained 80% of the data and tested it on the 20% and obtained popularity based on regression and random forest. We can see that, except for the outliers, the popularity predicted have been strikingly similar.

Step By Step:

- We define a scoring function using features that are most relevant with the popularity of a movie(correlation) and use it to calculate the popularity. This is a heuristic function. This will be our baseline model. The function used to calculate the popularity is as follows,



$P = (\log(C_1F_1) + C_2F_2 + C_3F_3 + C_4F_4) * e$ ^ (- years)

Where C_1, C_2, C_3, C_4 are the coefficients. At times, log is used for the features like *Budget or Revenue or N_votes* if they are supposedly having much impact on the overall prediction because of high values. Also, as the popularity fades over time, we have used the decay function.

Also, based on a scoring function which we had designed, we predicted the popularity metric and plotted the difference between the actual popularity and that obtained by the scoring function. Though the overall flow of the graph is the same, the values are difference at many points indicating the need to improve scoring function.Using the above function, we calculated the popularity and calculated the error with the original training data provided in the IMDb. We got a RMSE of 188 showing a need to improve on it.

- We then use machine learning models to make a prediction on the popularity based on the production values(normalized data), to see how well it stands the test of time in terms of popularity. We expect movies with high budget, a strong cast and director, reviews, awards to affect its popularity over time.
- We first train a linear regression model on the selected features as mentioned above and we get a root mean square error of 22 on the popularity. We notice that the linear regression model performed very close to the baseline model for most movies, but there were certain movies that underperformed and overperformed.
- We then train a random forest regression to predict the popularity. We get a mean absolute error of . We notice that random forest regression performs slightly better than the linear regression model.

Validation:

We will make use of IMDB pro database to extract live relative popularity among movies and use it as the evaluation metric for our predictions. We take our predictions for a set of movies at the current instant and compare with the movie meter data to see if the predicted movies fall in the same order.

Next Steps:

- Obtain more data in the form of twitter data and scraping critic reviews from newspapers and using them as features.

- Genre has been studied but only one genre per movie has been considered until now. We plan to consider multiple genres for the movie.
- We plan to use the CPI index to adjust the budget so that an appropriate value is considered in the scoring function.
- Consider more number of features to make a more accurate prediction. Perform feature engineering to build additional features from existing features.
- In depth analysis of shift in trend of more favored genre by movie goers.
- Improve on the existing scoring function to for a better heuristic prediction.
- IMDb Pro gives the *StarMeter* which gives us the popularity of a celebrity every week. We plan to extract that data and look for a correlation between the changing popularity of the movie and the *StarMeter.*
- Also, we want to focus on developing a method to define some robust features which helps us define the initial popularity index and then test our best model to predict its popularity index over the years and analyse if a popular song really does endure over the years or not.

  .