**Letterboxd Ratings**

**Problem statement**

When releasing a new film, it’s a good consideration for stakeholders to investigate how difference audiences will respond, especially avid film fans who are more likely to review and share films. Letterboxd is a social network for film reviewers, increasingly popular with critics and movie fans alike with an audience skewed towards young adults, a critical demographic. Letterboxd allows users to rate films from 0 – 5 with stars

**Dataset and EDA**

I used Andres Hernandez’s Letterboxd dataset scraped from the site. The data is in CSV format with five files for the genres Animation, Horror, SciFi, Thriller and War. Each file included the features:

Title – title of the film

Year – year the film was released

Director

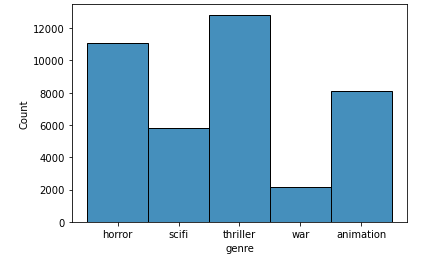
Running\_time in minutes

Views – number of people who have viewed the film on Letterboxd

Likes – number of people who have liked the film on Letterboxd

Avg\_rating – average rating of the film given by users on Letterboxd

Half\_star, One\_star, One\_half\_star, Two\_star, Two\_half\_star, Three\_star, Three\_half\_star, Four\_star, Four\_half\_star, Five\_star – number of people who gave a certain rating to a film



useful to see if a film is “controversial,” i.e. has a large number of high and low ratings.

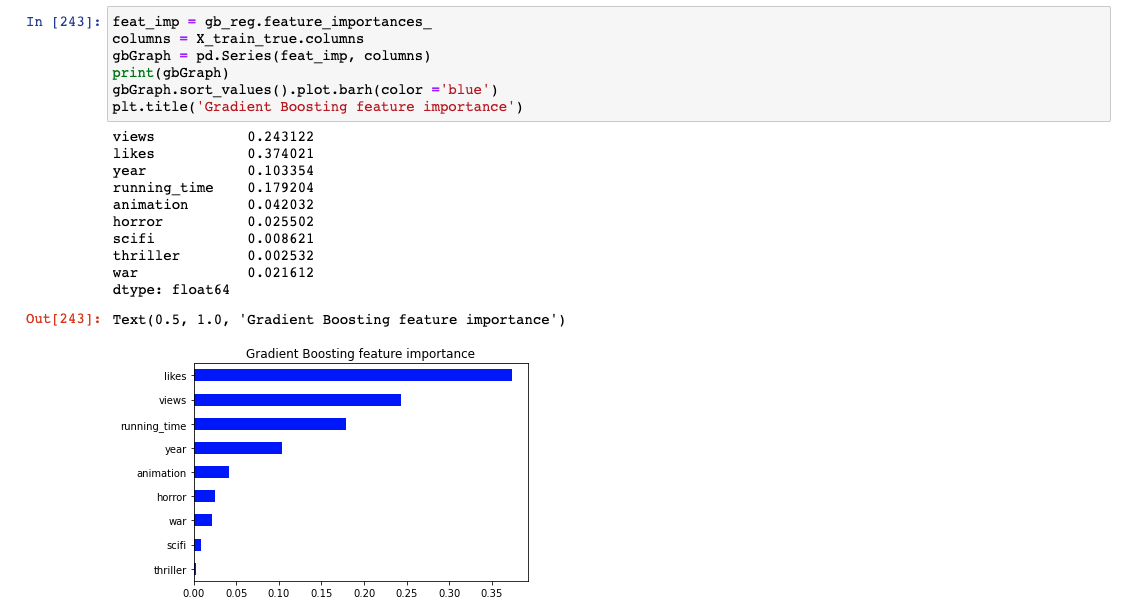
**Data Wrangling and Cleaning**

Filled director with directorNA

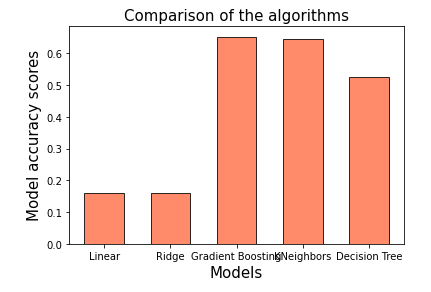
Made new column for missing years, replaced NaNs with 0

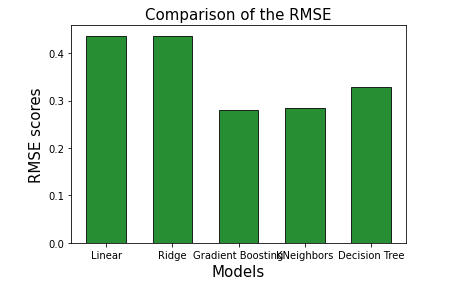
**Leakage and Giveaway features**

As part of my experiments, I also trained several models including views and likes with the other features.



This resulted in the best accuracy of any of the models, with a model score on unseen dating achieving 0.65 accuracy and an RMSE of 0.28 with a Gradient Boosting Regressor and an accuracy of 0.646 with an RMSE of 0.28 for KNeighbors Regressor. Decision Tree beat Linear and Ridge Regression, however it overfit to the data with a training set accuracy of 0.99 and a test set accuracy of only 0.52, lower than Gradient Boosting and KNeighbors Regressor. Below is a comparison of models trained on all the numeric data including views and likes.

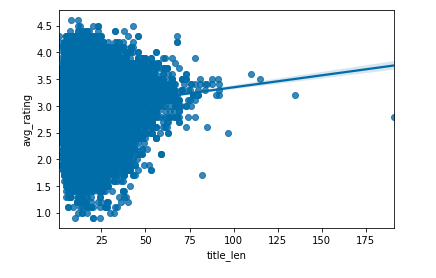




However, these features are giveaway features, as stakeholders would not know the likes and views of their film before it is released. These features are the most important to the Gradient Boosting Regressor because likes and views are highly correlated with higher rated films. While these models have the impressive accuracy for the dataset, these are not useful for the problem at hand. For stakeholders who want to know how to market and release their film, these features are useless as these numbers are difficult to estimate until the film is actually released.

**NLP on Title Length**

One interesting feature I explored is the length of films’ titles. I used Natural Language Processing techniques to extract the length of characters in each title and created a new feature, title\_len. Length is lightly correlated to the average rating and longer titles tend to have slightly higher ratings.



Adding this feature to the existing models improved accuracy.

**Best Models**

Gradient Boosting Regression with title\_len and all numeric features. While not accurate enough to give a precise rating, can help stakeholders understand the features that determine whether a film is highly rated

However, Linear Regression is easier to interpret and simpler model with faster training time

**Future Explorations**

It might be useful to perform NLP on different features in the movie titles, such as I, II and V to identify sequels. Directors might also be a useful feature, though it would be very difficult to train a model on them using indicator variables as there would be thousands of features and most directors are listed very few times.

Past experiments:

>directors and ratings

- linear regression model

- directors as features and ratings as target variable, 6000 features not helpful for model

- dimensionality problem, good on directors its seen but not ones it hasn't, too many features

**Takeaways**