***Summary:***

None of our models met our final goal, which was to exceed 70% accuracy or R-squared for predicting a rating using features we had selected and extracted from the data. We attempted both regression and classification using Random Forest. Regression did not perform better than our baseline, 50% (around 50% of the movies in the data are rated as a 10). Classification did perform better than the baseline, but only when classes were reduced from the original 1-10 rating scale to a scale of three (movies rated less than 8, movies rated 8-9, and movies rated a 10). Nevertheless, our best model still only had a 68% accuracy score. Hence, we would not consider the project successful.

In our model, the most important features were sentiment of the review and release year. Genres were at the bottom of the list, showing that text features as ranked by TF-IDF dominated the model and suggesting six next steps for analysis (in order of potential value add):

1. Attempting a binary classification (is the movie a 10 or not a 10);
2. Conducting sentiment analysis with Spacy and seeing if this creates a more predictive feature;
3. Adding in other features from IMDB and other datasets that might add predictive power irrespective of model (i.e. marketing budget scaled for inflation, critic reviews and sentiment);
4. Feature Engineering: are there combinations of existing features in our dataset that are more predictive together than on their own?
5. Attempting a Gradient Boosting Classifier, as the ensemble methods used bagging ones (Random Forest) and one boosting one (AdaBoost);
6. Customer Segmentation: if we could match review to type(s) of customer Netflix wants, would we know for that segment what makes it likely they rank a movie highly, and what would attract them to Netflix?

***Problem Statement:***

Our goal was simple, to figure out what factors can determine what people rate a movie. More to the point, for a general set of people, what factors may indicate that they rate a movie highly, such that it is worthwhile for Netflix to offer movies that fit those factors?

From the IMDB data, we know that movies with a rating of 10 dominate the top 100 movie list - around 50% of the list. Therefore, success would be if our model scores better than the baseline by a sufficient margin. For the sake of simplicity, let us say that we want to be right between 70 and 80 percent of the time. The model that meets that is what we will deploy to help pick movies to bid on for Netflix.

*Out of Scope but fun if possible:* would the model provide insights useful to the creation of original content for Netflix so that it has more leverage in bidding on content Hollywood and international studios create?

***Data*:**

To answer this question, we used summary and individual review data from the IMDB database, focusing on the top 100 movies of all time as rated by individuals. We included the following features from the database:

* Genres
* Movie length

We also extracted the following features:

* The top 200 one and two word n-grams as ranked by TF-IDF
* User sentiment from each review using TextBlob and its sentiment polarity ranking (a -1 to 1 scale with -1 being highly negative and 1 being highly positive).

Excluded features worth considering for future analysis:

* Actors
* Marketing budget (not in IMDB, would have to be found elsewhere, and scaled for inflation)
* Critic reviews (they may have an influence on people seeing movies and deciding if they like them) - mainly sentiment and a critic’s actual rating.

***Analysis Done:***

The main analysis that we did consisted of exploratory analysis to see potential relationships, which we saw among two features, and ensemble methods.

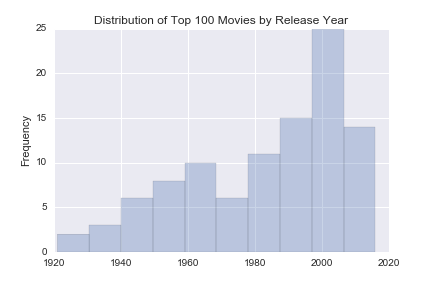
The two features of interest, because of their positive correlation to ratings, are:

* The release year
* Sentiment

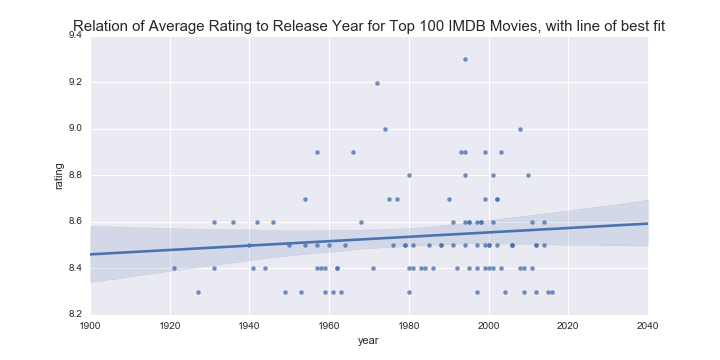
Given that the output we wanted to predict, movie ratings, is on a scale of one to ten, we judged that attempting both regression and classification made sense. For the former, keeping the 1-10 scale made sense. For the latter, we attempted to classify on the 1-10 scale, and a reduced scale (less than 8, 8-9, and 10). For classification, we judged that trying to discern if a movie was in the top 8-9, or the top 10, was worthwhile because those movies were likely to attract and retain the customers willing to pay for Netflix’s monthly subscription.

The two ensemble methods we employed were Random Forest Regression and Classification, both based on bagging.

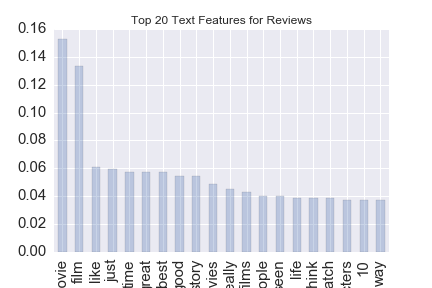
***Exploratory Data Analysis results***



Our preliminary analysis suggested that newer movies dominated the top ranked listed. Potentially this is because of actual popularity, perhaps because of the fact that the database relies on user submissions and recency of new movies in users’ memories (plus that more of them have seen them than the older movies) tilts the list.

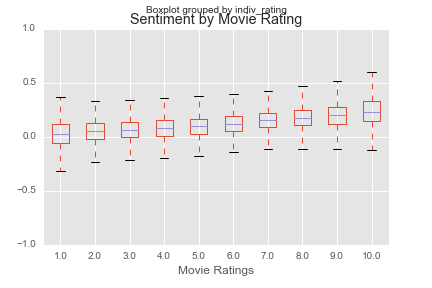


Buttressing the above observations, newer movies are skewed slightly towards higher ratings.



The top twenty features in the reviews as ranked by TF-IDF seem odd - movie and film seem unusual terms to predict if a movie is top ranked. Indeed, given the context, we would expect inverse document frequency to reduce the weighting of these terms (which it probably did). While we lost a lot of context in this analysis, it is possible that those terms were near other terms that are predictive of a high rating. Hence, we kept all 200 features in the analysis and let the algorithm filter them out.

Finally, as mentioned above, sentiment of users’ as judged by TextBlob looks to be correlated with rating, as we would hypothesize. The dispersal between low and high sentiment, and low and high ratings, is not as high as we would hypothesize. This could be because the database is highly skewed towards top rated movies, and so the proportion of negative reviews for those movies is small.



***Analysis:***

*Regression*

Our Random Forest Regressor returned an R-squared of .28, which is below the baseline. Rather than using GridSearchCV to find optimal parameters, we deemed it a better use of time to consider the problem a classification one (i.e. is this movie going to be a highly rated or not based on the features we can observe).

*Classification*

We utilized Random Forest Classification on unscaled and scaled data - for the scaling, we only scaled the release year and movie length, as both had different scaled than the best of our features. However, we did not notice scaling improving our model.

We first attempted to classify reviews on the original 1 to 10 scale. The model had a low accuracy score, 0.49, and only did well with classifying 10s as 10. The improvement on the baseline (essentially not 10 vs 10, since 10 is the dominant part of the data, was 0.01. Below is the classification report, which shows how the True Positive Rate (recall) for the model is only good for 10s. Both precision and recall are not good for the model for other ratings, telling us the model does not do a good job finding all positive ratings (recall) or discerning true positives from false positive reviews (precision).

**precision recall f1-score support**

1.0 0.27 0.26 0.26 847

2.0 0.08 0.02 0.03 275

3.0 0.07 0.02 0.03 332

4.0 0.04 0.01 0.01 359

5.0 0.05 0.01 0.02 458

6.0 0.08 0.02 0.03 633

7.0 0.16 0.05 0.07 1003

8.0 0.14 0.06 0.08 1732

9.0 0.20 0.12 0.15 3165

10.0 0.58 0.85 0.69 9716

avg / total 0.38 0.49 0.41 18520

However, when reducing the scale for ratings to three classes, our accuracy significantly improved to 0.621 for our best model. The improvement on the baseline was 0.15. While not enough to go over our threshold, our model still relied primarily on user sentiment, the release year, and a number of text features.

Our best model had the below parameters, discovered via GridSearch. Given the depth of the trees and number of estimators, it is a complex model. It does better than the above model for precision and recall for 10, and improves the average precision/recall significantly.

*Optimal Parameters for Random Forest Classifier:*

{'criterion': 'entropy',

'max\_depth': 120,

'max\_features': 'log2',

'min\_samples\_leaf': 2,

'n\_estimators': 90}

*Classification Report for the Best Random Forest Classifier*

**precision recall f1-score support**

10 0.65 0.87 0.74 9716

8-9 0.45 0.15 0.23 4897

<8 0.60 0.59 0.59 3907

avg / total 0.59 0.62 0.57 18520

*Most Important Features in the Model*

importance

sentiment 0.063685

release\_year 0.020506

best 0.019240

runtime 0.018036

good 0.017642

movie 0.017357

film 0.015071

like 0.013438

just 0.013263

great 0.011832

time 0.011381

seen 0.011019

story 0.010479

*AdaBoost*

We also attempted an AdaBoost classifier utilizing a Decision Tree Classifier as our base estimator since we had used Random Forest above and thought using trees for a base estimator was still a good idea.. We ran four iterations, with the number of estimators as 50, 100, 150, and 200. Unfortunately, our accuracy score did not exceed 58.1%.

The classification report below shows the same problem as for the above models - recall is just too low for the classes other than 10.

**precision recall f1-score support**

10 0.62 0.84 0.71 9798

8-9 0.36 0.21 0.27 4841

<8 0.64 0.41 0.50 4060

avg / total 0.56 0.58 0.55 18699

***Conclusion:***

While we discovered a model that improved on the baseline for features that predict if a movie is highly rated or not, and disclose those features of interest, we were not able to create a model that passed our imposed goal of a 70%+ accuracy score. The best model was an ensemble method utilizing bagging, the Random Forest Classifier. The model’s most important features were sentiment of the user review, and most of the top 200 text features in the reviews as determined by TF-IDF.

***Next Steps***

* Classification of Rating: Would a binary grouping of 10 and not 10, or 9/10 and not 9/10, have sufficiently greater accuracy to be a more useful model?
* Sentiment: we used TextBlob. Would testing with SpaCy produce a different sentiment analysis that had more predictive power?
* More Features: Are there other features in IMDB or other databases that would add predictive power?
* Feature Engineering: are there combinations of existing features in our dataset that are more predictive together than on their own?
* Boosting: would the Gradient Descent Boosting perform better?
* Customer Segmentation: if we could match review to type(s) of customer Netflix wants, would we know for that segment what makes it likely they rank a movie highly, and what would attract them to Netflix?