Letterboxd Data Analysis

Evaluating my movie-rating behavior by training predictive models on personal data

Nick Wibert

December 8, 2021

Introduction

Letterboxd is a website that doubles as a movie diary tool and a social media app. One can use the app to log films they've watched, rate them out of 5 stars, write reviews, create lists, and follow/interact with other users. I have been using this website since May 2016 and have logged nearly every movie I have watched since then, the vast majority of which I assigned a rating out of five stars.

The website enables users to export their diary in a .csv format, which includes basic information on each film logged, the date it was logged, the rating out of five stars, etc. I am interested in exploring and analyzing my movie ratings using methods learned throughout this course, with the goal of evaluating the predictive performance of various classification algorithms trained on my rating data and hopefully gaining some insight on my movie-rating behavior.

Data collection / Description of data set

Before describing the data set, I will briefly go over some of the data collection process. While you can easily export your diary, the columns in this data set are extremely bare-bones, with only 2-3 features which could be relevant for model-fitting. To expand the covariate space, I wrote a script for web-scraping using the `rvest` package. Since the original data set includes URLs to each film's page, I was able to iterate through the data set, visit each URL, and parse through the page's HTML code to extract information that may have an influence on my movie ratings. For the sake of simplicity, I scraped pretty much everything that was available, with the intent to exclude many of the predictors through variable selection procedures later on. The new columns added to the data set are the following: Average Rating (average of all user ratings for the given movie), Runtime (minutes), Genre, Country, Language, Studio, and various crew members.

The original data set before web-scraping consisted of the following columns: Title, Release Year, Rating, Tags, and Watched.Date. The total number of observations in the final data set after removing duplicates is a bit over 750, with 16 covariates; a single record is printed below. The response variable is Rating, which is on a 5-star scale with intervals of 0.5 stars, resulting in 10 levels for classification. The "Tags" column includes user-defined tags for each diary entry; I have consistently used this feature to denote the format in which I watched the film, with four distinct categories: "theatre", "streaming", "blu-ray", and "dvd". "Release Year" is the year the film came out, and "Watched.Date" is the date I watched the film (which I have reduced to year only). It is immediately clear that classification algorithms are the way to go here, using a combination of continuous and categorical covariates to classify films into one of 10 outcomes.

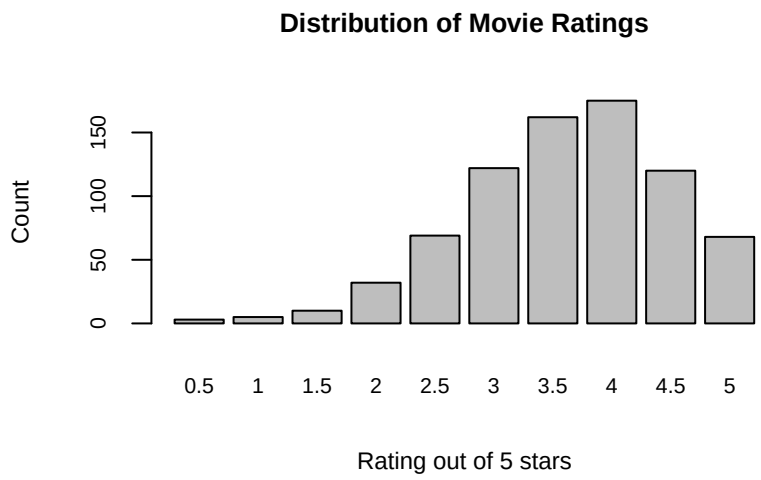
```
##        Name Year Rating    Tags Watched.Date Average.Rating Runtime Genre
##  Uncut Gems 2019    4.5 theatre         2019           3.95     136 drama
```

```
##     Director        Actor      Writer       Editor Cinematographer
##  Josh Safdie Adam Sandler Josh Safdie Benny Safdie  Darius Khondji
```

```
##        Composer        Producer Studio Country Language
##  Daniel Lopatin Martin Scorsese    A24     USA  English
```

Exploratory data analysis

Given the nature of my goals with this analysis, it makes sense to first look at a simple bar chart to get an idea of the distribution of movie ratings.



There is a clear left skew in this data set, with a mean somewhere between 3.5 and 4 stars. Since we will be classifying an imbalanced data set, this will need to be taken into account when performing cross-validation by stratifying samples in an attempt to equally represent each class in the training and test data sets.

Another noteworthy feature of the data set is the high correlation between the response (Rating) and the user-average rating pulled from the website (Average.Rating), which is just over 80%. The correlation for other individual predictors are all much smaller, far less than 50%, so I expect Average.Rating to be the most informative in predicting the response, regardless of the classification method employed.

A large chunk of the web-scraped covariates were sure to be excluded from the get-go after it quickly became clear that they had little significance in predicting movie rating, had many linear dependencies, and were too heavy computationally due to many factor levels. All of the crew member covariates as well as "Country" and "Language" had far too many possible values to be reasonably included as categorical predictors (30+ levels, several hundred for crew members). After performing best subset selection and backward stepwise regression with all of these covariates included, they were the first to go. The three main covariates that were decided upon by both variable selection procedures were Average.Rating, Watched.Date, and "Tags" (the format I watched the film). Best subset selection also tended to include some, but not all levels of "Genre"; some of the more significant levels were "drama", "thriller", and "crime".

In the next section, I will perform simulation studies using some combinations of these selected covariates. Since "Average.Rating" seems to be much more significant than all other covariates, I will also run a study using that as the only predictor to see how performance compares to models with several predictors.

Evaluation of classification algorithms on movie rating data

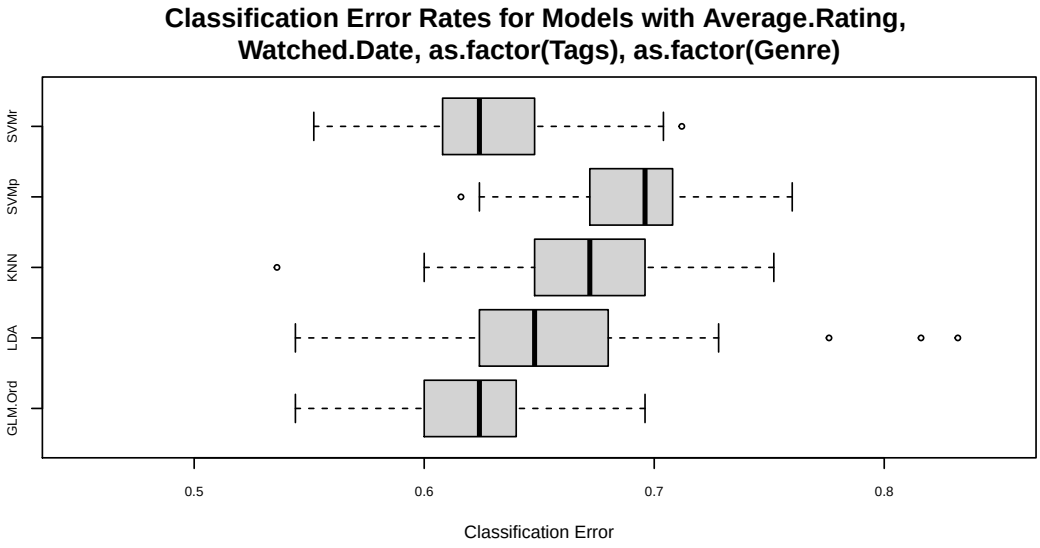
For the simulation studies, I decided to consider four of the major classification algorithms discussed in the course: linear discriminant analysis (LDA), K-nearest neighbors (KNN), and support vector machines with both polynomial and radial kernels. We used these most often for binary classifications in class, but all of them naturally extended to n-level classification in `R`. Logistic regression was one of the major classification algorithms we covered, but ran into some issues with it.

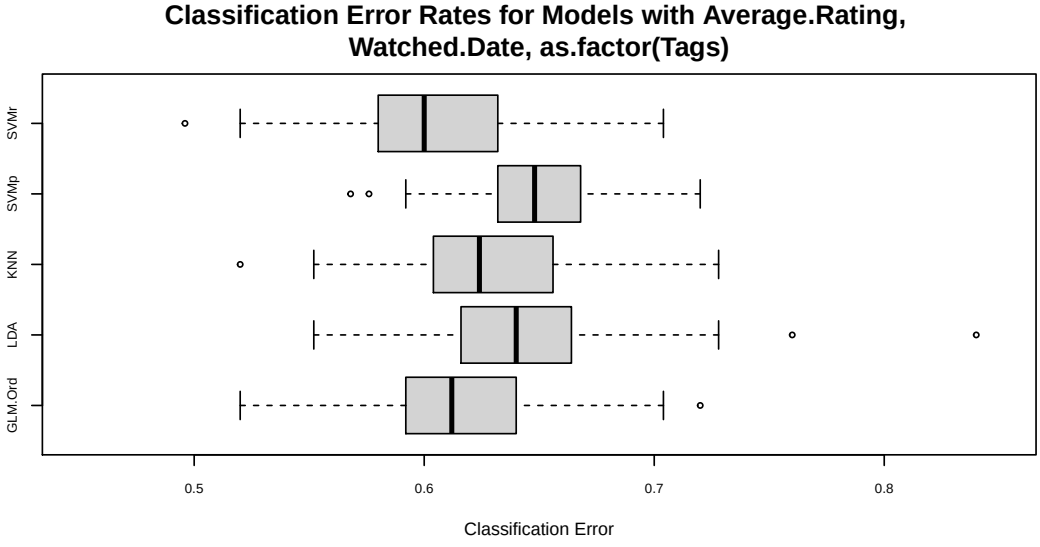
For a classification problem with many levels such as ours, we would need to use a modified logit that can handle a categorical/ordinal response. The extension of the logistic model that we discussed in class was the baseline-category logit model, but these models are not commonly used for classification purposes, and I ran into issues in implementation. Another option would be the *cumulative logit model with proportional odds* discussed in the Categorical Data Analysis course, which is designed to handle ordinal response variables. This model makes an important assumption, however, of *proportional odds*, meaning that the effect of each predictor is the same for each logit (essentially, for each class). Performing a likelihood ratio test using one model with the assumption and one without the assumption resulted in a statistically significant p-value from the chi-squared test statistic, indicating that the proportional odds assumption does not hold. Despite this, I have still implemented the model here just out of the sake of curiosity and to see how its performance compares with other methods, though it is important to keep in mind that the major model assumption is violated by this data set and thus it is not really valid.

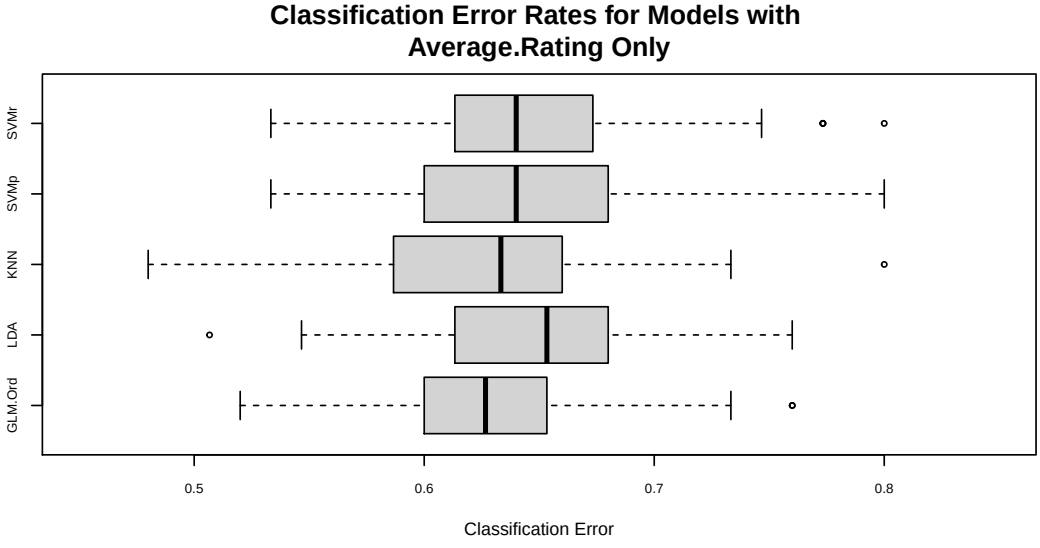
Once the models were decided, I ran three simulation studies: one using `Average.Rating`, `Watched.Date`, `as.factor(Tags)`, and `as.factor(Genre)`, as selected by best subset; one using the same set, minus Genre, as selected by backward stepwise regression; and finally one using only `Average.Rating`. Each simulation was a loop with 100 iterations, with cross-validation occurring in each loop using stratified re-sampling to "randomly" choose 80% of the data to be training data and 20% to be testing data. I put quotes aroudn "randomly" because, by stratifying based on Rating, I ensured that as close to 80% *of each class* would be included in the training data set, as an attempt to prevent classes from being over- or underrepresented. Once the training and test data was separated out, each model was trained on the training data, predictions run on the test data, and the classification error rate was stored in a matrix. After each simulation study, I averaged these classification error rates over all 100 iterations.

Across all three simulation studies and all methods implemented, the classification error rates ranged from 60% to 69%. The error rates were generally higher when Genre was included as a categorical predictor compared to when it was not. Furthermore, the classification error rate for models *only* including `Average.Rating` was generally lower than or very close to the error rates for models including more predictors, which shows how much more influence this covariate has over the prediction than the others. Looking at the coefficients for these models shows that `Average.Rating` is weighed much heavier than anything else, with weights consistently in the range of 4-6 while the absolute values of coefficients for all other predictors were generally less than 1.

The following page has box-plots displaying the classification error rates for every method in each of the three simulation studies.


Classification Error Rates for Models with Average.Rating, Watched.Date, as.factor(Tags), as.factor(Genre)
SVMr
SVMp
KNN
LDA
GLM.Ord
0.5
0.6
0.7
0.8
Classification Error


Classification Error Rates for Models with Average.Rating, Watched.Date, as.factor(Tags)
SVMr
SVMp
KNN
LDA
GLM.Ord
0.5
0.6
0.7
0.8
Classification Error


Classification Error Rates for Models with Average.Rating Only
SVMr
SVMp
KNN
LDA
GLM.Ord
0.5
0.6
0.7
0.8
Classification Error

Discussion / Reflection

Predicting human behavior is difficult, especially when it comes to something as subjective as assigning a rating to a film. There are so many factors that decide what will make an individual like a film, so I knew when I started work on this project that my goal was not to build some excellent machine learning model that predicts what movies I will like, but rather to use the opportunity to work with a data set I have been building up over the years and perhaps get some insight on what influences my movie ratings.

To evaluate the performance of the models, we need to have a benchmark to compare our error rates against. The simplest benchmark to consider is that of random chance; in a binary classification, this amounts to a 50% error rate, while in a 10-level classification problem, this would amount to a 90% error rate (10% success rate). Perhaps a slightly more rigorous metric would be more appropriate, such as the "zero rule," which always predicts the majority class in the dataset. In the case of my data set, the majority class is 4 stars, which makes up a proportion of about 23% of the final data set after excluding duplicates and `NA` ratings. So, a success rate of over 23% would be desirable in this context, and show that the models have some level of predictabililty. The success rates across the simulations ranged from around 30-40% on average, showing that while the performance isn't particularly *great* relative to these benchmarks, it is an improvement, and demonstrates some level of predictability (mostly due to the high correlation of Average Rating with my response).

There are a few possible reasons why Average Rating was such a significant predictor in determining my movie ratings across all the models considered. When you visit a film's page on the website to rate it, one of the last things you see is the site-average rating; so, it is very possible that I could be influenced subconsciously by seeing this average while pondering my rating for a film. A simpler explanation is that perhaps my taste just tends to align with that of the Letterboxd user population. The site is advertised as for "movie-lovers", and most people who are going to use a movie diary social media service probably have different taste/preferences when compared to any old moviegoer. I consider myself a strong lover of cinema, so it does not surprise me that my ratings tend to align with the user-base of the website.

Finally, I will reflect on how the weaknesses of the data set were emphasized by certain aspects of model performance, as well as weaknesses or oversights in my overall simulation design. Like I mentioned before, this data set is imbalanced and classes are not equally represented, and this fact had a clear and direct impact on predictive power. Stratified sampling was my attempt at mitigating this issue, but it had very little effect on error rates compared to standard random sampling. All of the models tended to fail more often when the Average Rating or my true rating were in the lower half of ratings (0-2.5 stars), which is expected due to the lack of data in those classes. Since the models were trained primarily on data in the upper range of ratings, it makes sense that predictions will struggle more in situations where a lower-rated prediction should occur. Just for fun, I tested this idea on the 6 films that I have seen in between starting this project and finishing it. None of the models predicted more than 2 of these film ratings correctly, and the correctly rated ones were always films whose true rating was greater than or equal to 4 stars.

If I were to conduct further work on this data set, I would try a couple of different things. First, I would employ more models that actually incorporate the ordinality of the response. Besides the proportional odds model (which wasn't even valid for this data), none of the models incorporated ordinality, but rather treated Rating as a nominal response. While this isn't an issue per se, there is potential for lost information when disregarding the ordinality. The proportional odds model seemed to perform slightly better than all the other models for the most part, and it may have to do with the fact that it is the only model considering the order of the response. Another idea would be to web-scrape more relevant covariates from other sources. Due to time constraints and the difficulty of figuring out how to web-scrape in R, I limited myself to Letterboxd.com and the information available there. It would be possible to link the films to pages on different sites, like IMDb, to get more numerical covariates that may be significant such as budget and box office numbers.