Using a Linear Model to Predict Product Ratings

Sean Pompea

2023

Contents

Introduction

Background

Recommendation systems are machine learning–based software systems that attempt to predict the rating that a user might give a particular item; if the predicted rating is high, the item is then recommended to the user. The data used to make such predictions often consists of, at the very least, information about the previous ratings history of the user, and the ratings histories of other users and items; in practice such systems are sophisticated and make use of many different kinds of data. One context where such systems are

used is health recommender systems, which roughly can be broken down into four categories: general health care, lifestyle, nutrition, and particular health problems (1).

Irizarry (2) demonstrates the use of a linear model to generate movie recommendations using the publicly-available MovieLens 10M Dataset from GroupLens (3). Irizarry's example model uses terms that calculate the bias or effect of *user id* and *movie id* dimensions of the data; their model had three terms and achieved an RMSE (root square mean error) of 0.881. (RMSE is commonly used to assess the quality of a machine learning model.) A rating can be from 0 to 5 points, so 0.881 represents an error of approximately nine-tenths of a point.

Approach

For this study, Irizarry's example is extended via the addition of two new bias/effect terms—terms accounting for *genres* effect and effect of the film's *release year*—with the hypothesis that it would improve the performance of the model. I built and tested a linear model based on this hypothesis. This model also employed regularization to dampen the effect that outliers in the data would have. The λ parameter for regularization was chosen via tuning. Data was split into working and holdout data sets; the model was evaluated at the very end on the holdout set.

Performance

The enhanced model achieved an improved RMSE of 0.864797 on a test set partition created from the working data set, and 0.8651948 on the holdout data set. This work highlights the advantage of identifying additional features and introducing additional terms when constructing linear models for prediction systems.

Data set

For this study, there are four pertinent columns MovieLnes 10M data set:

- movieId: unique identifier for each movie
- userId: unique identifier for each user
- title: the movie title (which also contains the movie's year of release)
- genres: a pipe-separated string of all genres applicable to the movie
- rating: a rating from 0 to 5

Methods and analysis

Key steps

- Initial split of the data into a working set and holdout set.
- Exploring the data (and cleaning if necessary–though it wasn't)
- Data visualizations along the way
- Further splitting the working data into train and test sets
- Building out the model step by step, one term at a time:
  - movie effect term
  - user effect term
  - genres effect term
  - release year effect term
- Implementing regularization and the discovery of a tuned λ term
- Lastly, assessing the model with the holdout data set.

Data exploration

The data was split into holdout and working sets.

Because this data set was prepared by GroupLens, it is already cleaned and prepped. Analysis confirmed that there were no empty values for the pertinent columns.

The working data set contained 9000055 rows where each row was a particular movie rating by a particular user.

Reviewing the first few rows of the data:

	userId	movieId	rating	title	genres
1	1	122	5	Boomerang (1992)	Comedy|Romance
2	1	185	5	Net, The (1995)	Action|Crime|Thriller
4	1	292	5	Outbreak (1995)	Action|Drama|Sci-Fi|Thriller
5	1	316	5	Stargate (1994)	Action|Adventure|Sci-Fi
6	1	329	5	Star Trek: Generations (1994)	Action|Adventure|Drama|Sci-Fi
7	1	355	5	Flintstones, The (1994)	Children|Comedy|Fantasy

Since not all users have rated all movies, there will be user-movie combinations that are as yet unrated—these "empty" slots are precisely what the prediction model will aim to fill in.

The data, then, can be thought of as a sparse matrix, with some slots filled and other slots emptied.

Visualizing a subset of the data as a sparse matrix:

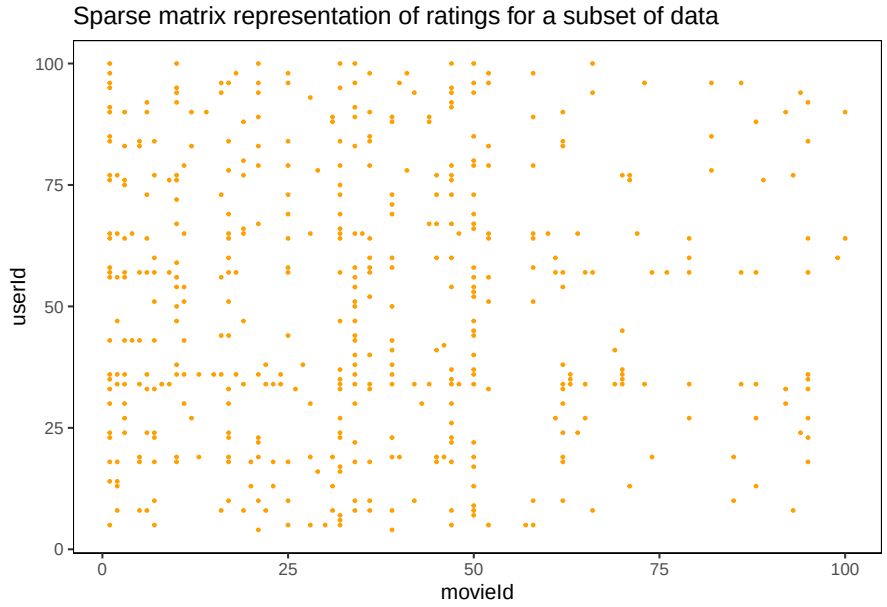

Figure 1: A sparse matrix representation of ratings for a small subset of the working dataset. It can be observed that a majority of the cells in the matrix are empty, which helps to illustrate the objective of the recommendation model: to predict the values for empty cells.

To better under the distribution of the values for `genres` in the data set, a bar plot can be helpful.

Visualizing the genres values:

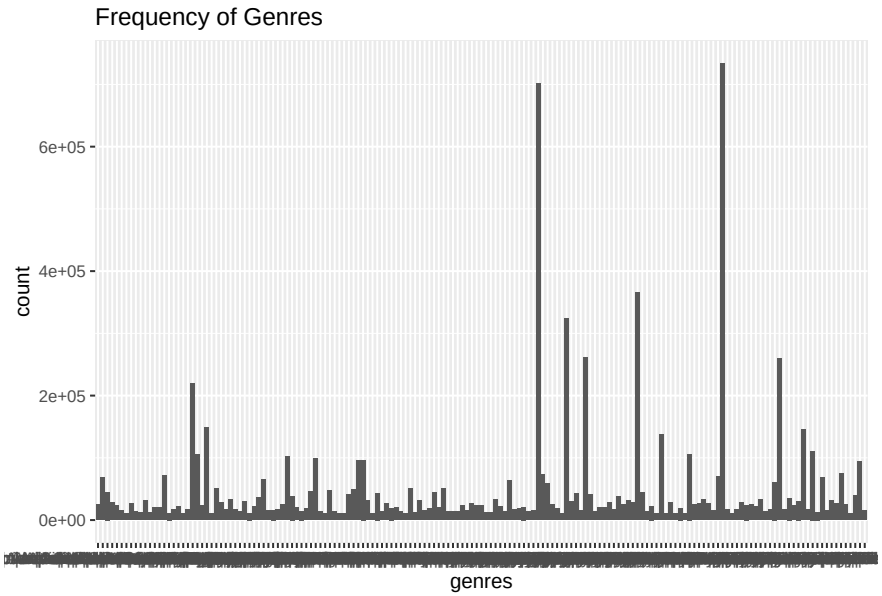

Figure 2: A visualization of the frequency of genres values.

The release year of the movie is included at the end of a movie title in parentheses. Part of the work for this project involved extracting those values and creating a new column for them. Visualizing the distribution of values for year can be helpful.

Visualizing values for year:

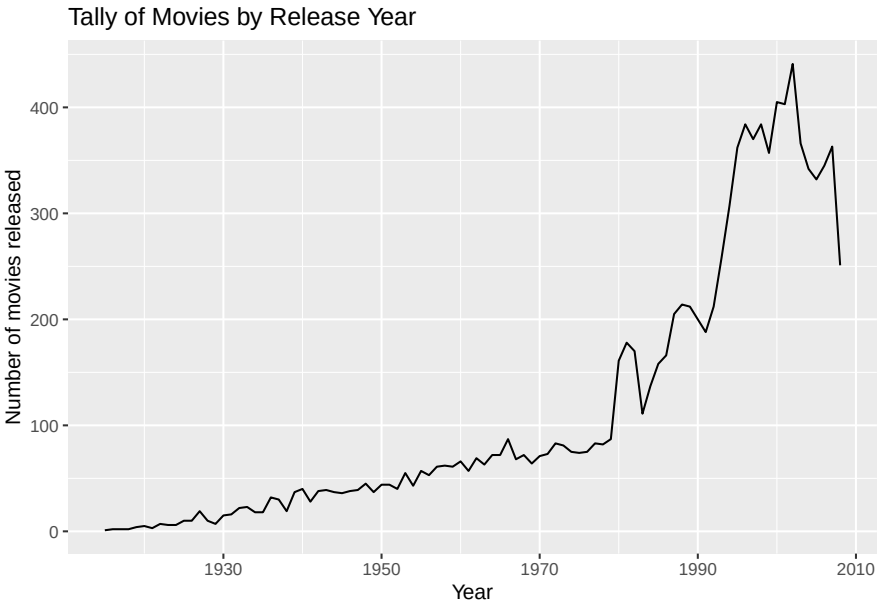

Figure 3: The number of movies per year in the data increases over time. The dip at the end is likely due to limitations in the data collection.

Adding a year column

I added the `yr` column to the `working` data set prior to modeling (which was extracted from the `title` column).

Building the model

Splitting into working and holdout sets

The MovieLens 10M data set was split into two separate sets, the working data set and the holdout data set. The model buildout was done with the working data set.

Train and test sets

The working data set was then split into train and test sets.

Naive model

The naive model was created by merely using a simple average of all ratings. This formed the constant term of the model as it continued to be built out.

The resulting formula:

$$Y = \mu \tag{1}$$

Resulting RMSE:

```
## [1] 1.059904
```

Movie effect

A term for the effect of a movie, b_i, can be modeled as the average of all ratings for that particular movie, and then adjusted for the value of μ.

The resulting formula:

$$Y = \mu + b_i \tag{2}$$

Resulting RMSE:

```
## [1] 0.9437429
```

Visualizing the distribution of b_i estimates:

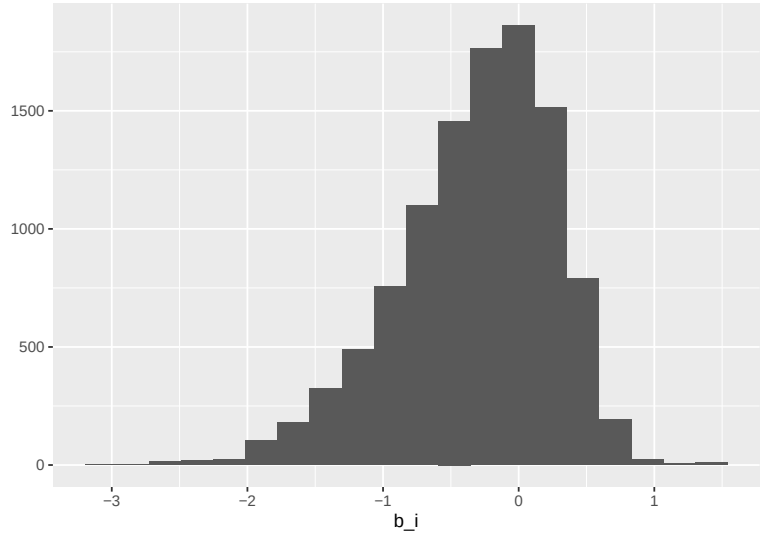

Figure 4: Visualizing movie effect estimates.

User effect

The next introduced term modeled the user effect, b_u. It is modeled similarly to the movie effect. It is adjusted using μ and b_i.

The resulting formula:

$$Y = \mu + b_i + b_u \tag{3}$$

Resulting RMSE:

```
## [1] 0.865932
```

Visualizing the distribution of b_u estimates:

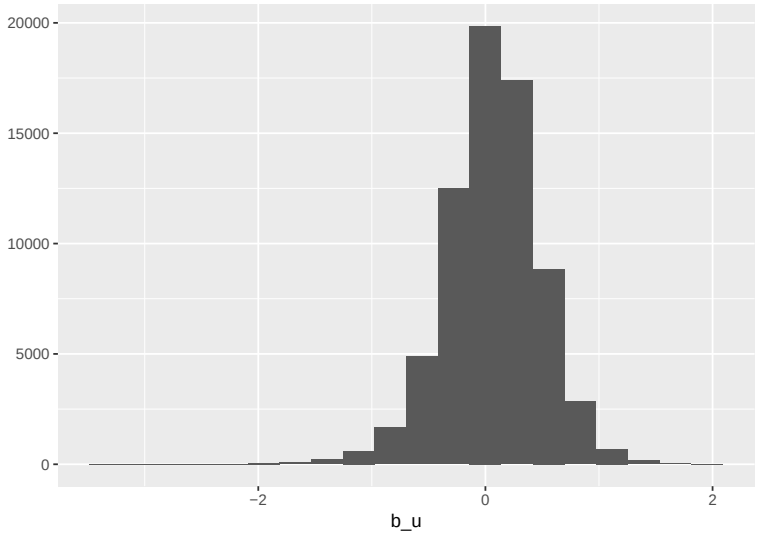

Figure 5: Visualizing user effect estimates.

Genres effect

In order to test my hypothesis, I then modeled the genres effect. This addition of the term b_g results in the following formula:

$$Y = \mu + b_i + b_u + b_g \tag{4}$$

Resulting RMSE:

```
## [1] 0.8655941
```

Visualizing the distribution of b_g estimates:

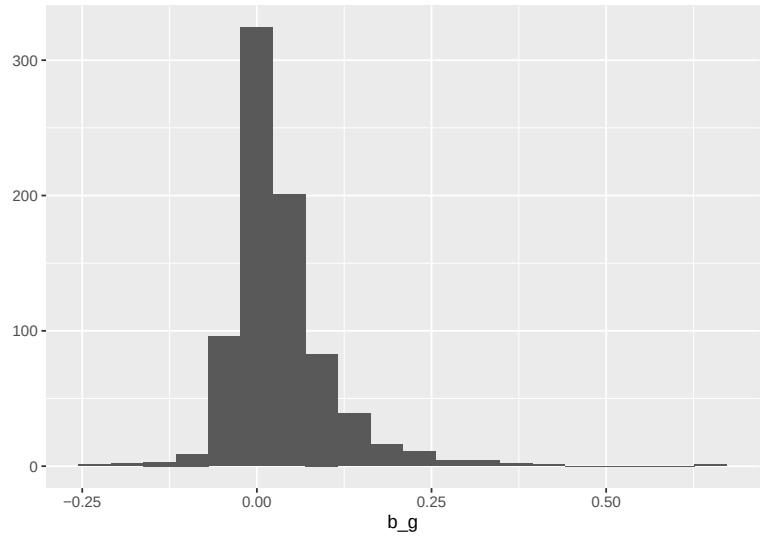

Figure 6: Visualizing genres effect estimates.

Release year effect

I then modeled release year effect term b_y similarly as before. Resulting formula:

$$Y_{u,i} = \mu + b_i + b_u + b_y \tag{5}$$

Resulting RMSE:

```
## [1] 0.8654189
```

Visualizing the distribution of b_y estimates:

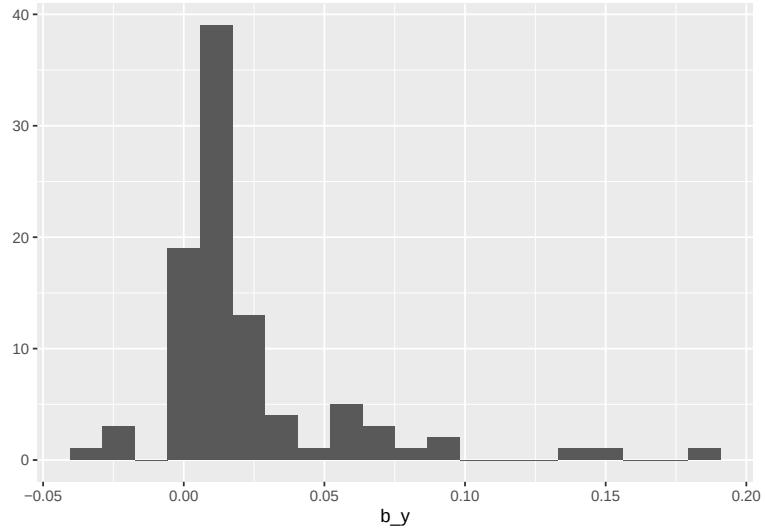

Figure 7: Visualizing release year effect esimates.

Regularization

Regularization was employed to soften the effect that outliers (e.g., movies with only a single rating) might have. A dampener calculation was introduced which was used to calculate new values for each team. λ was tuned to identify the optimal value.

Each effect term was then calculated using the dampener calculation; for example, the calculation for b_i:

$$b_i = 1/(\lambda + n_i) * \sum_{u=1}^{n_i} (Y - \mu) \tag{6}$$

In R, the calculation:

```
b_i = sum(rating - mu) / (n() + lambda))
```

Similarly, the calculation in R code for b_u becomes:

```
b_u = sum(rating - mu - b_i) / (n() + lambda))
```

The other two terms, b_g and b_y, are treated similarly.

Visualizing the generated lambda and RMSE values:

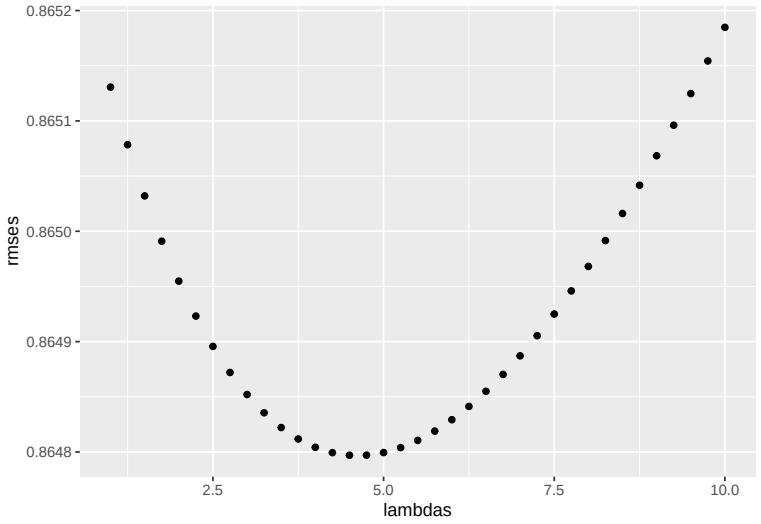

Figure 8: Visualizing RMSE vs. Lambda values.

Optimal lambda:

```
## [1] 4.5
```

With regularization employed and a value of 4.5 for λ, the final RMSE using the working data set was:

```
## [1] 0.864797
```

Evaluating the model with the holdout data set

A default of μ is used if the model encounters a movie-user-genres-year tuple that it hasn't encountered before (and therefore can't issue a prediction for).

When applied to the holdout data set, the final model achieved an RMSE of:

```
## [1] 0.8651948
```

Results

Outcomes with the working test set:

Model	RMSE
Naive Model	1.0599043
Movie Effect	0.9437429
Movie + User Effects	0.8659320
Movie + User + Genres Effects	0.8655941
Movie + User + Genres + Year Effects	0.8654189
Movie + User + Genres + Year Effects with Regularization	0.8647970

Outcome with the holdout data set:

Model	RMSE..Holdout.
User + Movie + Genres + Year Effects with Regularization	0.8651948

Conclusions

Summary

For this project, a linear model was built to predict movie ratings using the MovieLens 10M data set. The following effects were modeled: movie, user, genres, and year; regularization was employed with the introduction of a dampener calculation to reduce the effect of outliers. The model was able to achieve an RMSE of 0.864797, which supports my hypothesis that introducing additional terms to a linear model for movie recommendations can be beneficial.

Limitations of the current approach

The recommendation system built here does not take into account correlation patterns likely present in the data. A linear model is not the best choice for capturing such patterns. A non-linear model could take advantage of correlation patterns. Additionally, such correlations could be used to impute rating predictions for movies or users with few or even no data points, based on identified correlations.

Future work & discussion

A movie recommendation system is not something that benefits society in any meaningful way. Meaningful future work in this area could involve developing and improving health recommender systems or identifying other socially beneficial uses of recommender technologies.

"We live in a world where there is more and more information, and less and less meaning." – Baudrillard (4).

References

1. De Croon R, Van Houdt L, Htun NN, Štiglic G, Vanden Abeele V, Verbert K. Health Recommender Systems: Systematic Review. J Med Internet Res. 2021 Jun 29;23(6):e18035. doi: 10.2196/18035. PMID: 34185014; PMCID: PMC8278303.
2. Irizarry R. Introduction to Data Science. 2019. LeanPub. https://leanpub.com/datasciencebook.
3. GroupLens. MovieLens 1M Dataset. https://grouplens.org/datasets/movielens/10m/.
4. Baudrillard J. Simulacra and Simulation. 1981; paperback December 1994. University of Michigan Press.