Assignment 2 - Dataset Analysis

DENNIS LANG

University of California, San Diego
dlang@ucsd.edu

3 December 2024

Abstract

Clothing preference is one of the most notoriously subjective human characteristics, but user modeling can capture overall trends based on user reviews to personalize recommended items to wear. Large datasets like these are vital for applying machine learning algorithms to train recommendation systems even in situations when the data is less concrete. On the website RentTheRunway, a clothing rental subscription service,[1] a robust recommender system can be built using the in-depth review fields that users leave for each verified item rental.

1. The Dataset

The dataset contains a JSON-formatted subset of approximately 200,000 clothing fit reviews for items on RentTheRunway (*RTR*), which has an exhaustive number of valuable fields for every user-left review, as each corresponds to a verified purchase and wearing of the listed item. Each review contains information like the user's weight, body type, and the occasion the item was used for, along with a text review field of up to 1000 characters.

SIZE WORN:	SR
OVERALL FIT:	TRUE TO SIZE
RENTED FOR:	PARTY
USUALLY WEARS:	6
HEIGHT:	5' 6"
AGE:	43
BUST SIZE:	34C
BODY TYPE:	HOURGLASS
WEIGHT:	146LBS

Fig 1. Example Review Fields from *RentTheRunway.com.*[2]

The text of the reviews can be relatively easily mined for sentiment analysis by counting words like "love" and "hate" as well as exclamation points and punctuation that can be used to determine neutral and negative sentiments. Users can report whether the item was too large, too small, or perfect using the "fit" category.

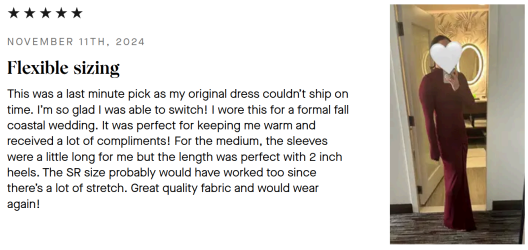

Fig 2. Typical Review Text on *RentTheRunway.com*[2]

Thus, physical features like bust size and height could be used to strongly predict fitting clothes based on closely matching body size, especially if the user tends to highly rate clothing based on fit. There is also a review summary field, which allows the user to summarize their review using the same number of characters as the full review text, but most users limit their reviews to a few words, making this a useful field for data manipulation. Each field can contribute to building a model of a prospective renter and determine the likelihood of a user making a purchase due to a specific item recommendation. Creating a model of the most popular types of

items based upon certain characteristics would be a useful way to use this data, as the most popular items with the fewest negative reviews would be the safest listings to recommend to new users. In order to find the most agreeable items on the site, it would be trivial to find the items with the least negative reviews using sentiment analysis, targeting terms like "didn't like" and "hated." With the star review as a reference, the model can use the other review categories combined with review text features in order to predict the likelihood of a review's text being negative or positive. The number of exclamation points can be trained to predict star rating directly, as in Professor McAuley's Homework 1[3]. This could be combined with a mean squared error in order to judge the accuracy level for the prediction. This dataset is very valuable for the verified nature of the reviews and how in-depth each user goes into their submission, so the use of multiple fields would make the best use of the data.

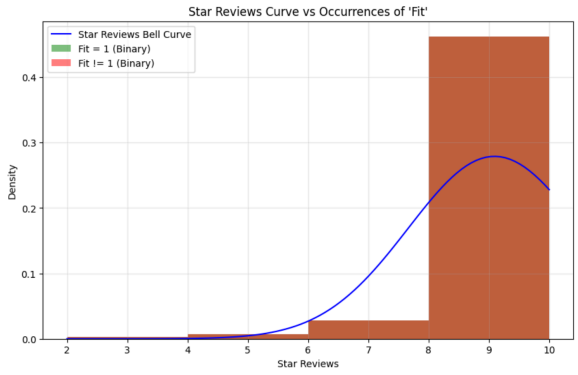

Fig 3. MatPlotLib Chart of Review Scores and "Fit" Labels

Crafting a robust recommendation service using this data should be a simple task and will be the end goal of this study. As the data has review scores that go from 1 to 5, the data is translated in JSON on a 2-10 scale, so a chart can be made comparing the curve of the star reviews' frequency from 2-10, and the occurrences of the "fit" label at each review score. We can see that the highest occurrences of "fitting" clothes appear at the 4 and 5 star range, while the lowest two scores of 1 and 2 stars had very few occurrences of the review label "fit" equalling "fit," as expected but wise to quantify as in Figure 3.

2. A Predictive Task

Using the other fields separated from their associated star review, there should be identifiably "more relevant" categories that should have more weight on their impact on the user's overall sentiment for the item. The baseline could be a simple average of all the star reviews in the set to use for comparison.


```
{"fit": "fit", "user_id": "420272", "bust size": "34d",
"item_id": "1083818", "weight": "137lbs", "rating": "10",
"rented for": "vacation", "review_text": "Absolutely
gorgeous dress. Great fit, no complaints! Lots of
compliments.", "body type": "hourglass", "review_summary":
"Beautiful!", "category": "dress", "height": "5' 8\"",
"size": 14, "age": "28", "review_date": "April 20, 2016"}
```


Fig 4. JSON-formatted all review fields on *RentTheRunway.com*

Upon initial inspection, "fit" is the most obvious indicator towards a positive review if they responded positively with the samely named "fit" as opposed to "big" or "small." I will also take into account both length of the review and the review summary, as the length of these appear to trend towards a positive review, a fact that I will verify. Counting the number of exclamation points used in each review will be trivial, after taking a smaller sample size of the dataset to account for my hardware, along with splitting further into training and testing data down the line. The initial coefficients shall be predicted star rating sans exclamation points and how much the star rating changes based on count, likely going up with more punctuation symbols. Then, encoding the three fit labels as '0', '1', and '2,' we can use a model that takes into account number of exclamations, length of review text and/or summary, and the categorical "fit" features in order to get a solid accuracy on the predictor when it comes to the star rating.

We can take the mean-squared error of each stage of the predictor in order to find out the correct trends. Then, for the likely best and most complex version of the predictor, we shall have 5 features to compare the associated MSE to determine the best predictor. For the user recommendation prediction to have the best results, narrowing down these categories that impact star rating the most will be key. “Fit” and the length of the reviews are my most likely candidates for improving my prediction from baseline. By analyzing trends in my error level as new features are added, I can identify which features add the most to my prediction accuracy. Additional features such as sentiment analysis by identifying the presence of positive and negative words in the review, may also help refine the predictions and reduce the error further. The validity of my model's predictions will be assessed by checking the trends in MSE as features are added incrementally and by ensuring consistent performance on unseen testing data.

3. The Model

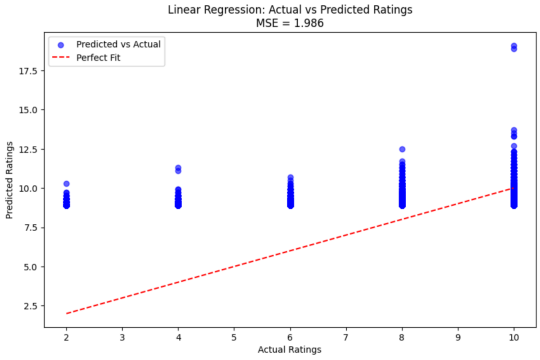

Fig 5. Actual vs. Predicted Ratings + MSE (Iteration 1)

The linear regression model here was used to incrementally improve the code by adding features over time, as the first feature was marginally better than baseline, as expected due to the nature of the exclamation points used on the site. The users tended to use the punctuation marks for positive and negative features, which I did not expect. Using 4 and 5 star reviews as “good” or ‘0’, 3 star reviews as “ok” or ‘1’, and 1 and 2 stars reviews as “bad” or ‘2,’ we can also determine the buckets that each verified review and predictive star rating will fall in. The data is formatted so the numbers 2, 4, 6, 8, and 10 correspond to 1-5 star reviews, respectively.

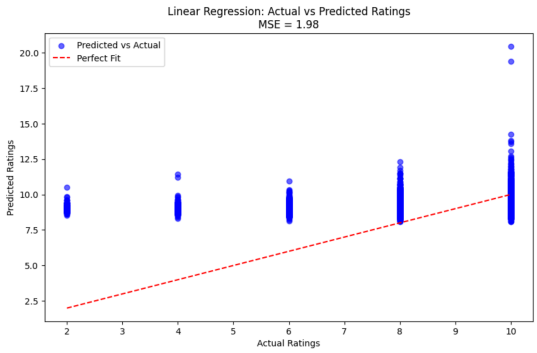

Fig 6. Actual vs. Predicted Ratings + MSE (Iteration 2)

Then, the first part of the predictor was simply to determine the effect of exclamation points on star count. I received a MSE of 1.985, just lower than the baseline of simply guessing the average of the star review which had an MSE of 2.05. The matplotlib library was able to help depict the initial actual versus predicted rankings as a graph. Adding a second feature to the model marginally improved it, bringing the MSE to within 1.98 of the actual star rating.

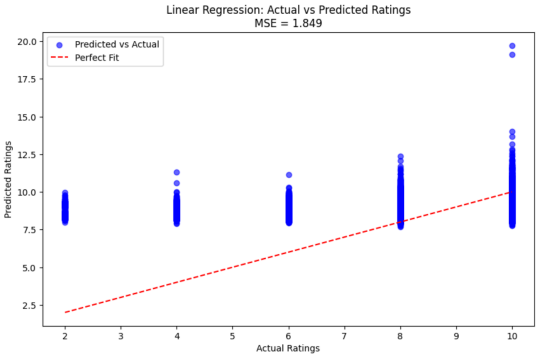

Fig 7. Actual vs. Predicted Ratings + MSE (Iteration 3)

Then, the next step was to add the contribution from the length of the review onto the linear regression prediction model as a second feature.

The real improvement came from a third feature; adding a positive weight for a fitting item, a negative weight for a non-fitting item, and both all resulted in the same MSE of 1.86, a 6% improvement from the previous version of the model. Another feature added for length of review summary simply as a proof of concept proved that adding review summary text length was marginally impacting the prediction of the review, in fact, the end result was slightly worse.

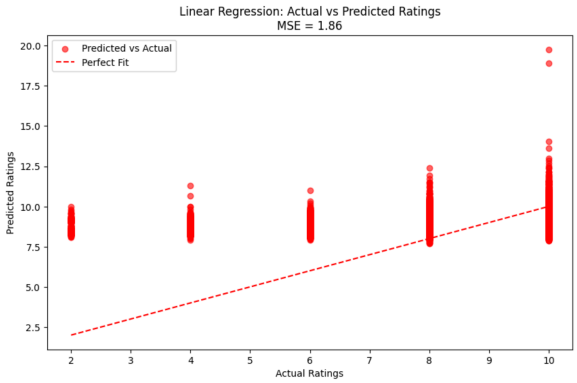

Fig 8. Actual vs. Predicted Ratings + MSE (Iteration 4 - Worse)

It appears from this data that the length of the review has less bearing on the positive or negative end result of the accuracy that I predicted. Finally adding a feature for negative weighting words like “bad” and “hate” and a positive weight for words like “like” and “love”, I was able to drop the MSE to my final value of 1.79 with a cap at 10, a marginal improvement but a useful increment towards perfection for this project. I attempted to create the model with a series of polynomials of degrees matching the star ratings based on exclamation points, review length, and reported item fit. After establishing the features from the predictor, I checked how well the model fit the data by reporting MSE values, which proved poorly as they were over 10, so I chose a different approach.

$$\text{Star Rating} \approx \theta_0 + \theta_1 \times [\text{length}] + \theta_2 \times [\text{number of '!' characters}] + \theta_3 \times [\text{fit adjustment}] + \theta_4 \times [\text{positive count}] + \theta_5 \times [\text{negative count}]$$

Fig 9. Linear Regression Model Equation

To show the non-linear relationships between the features, I tried to use the polynomial regression model and experimented with different degrees of the function to determine the appropriate level of underfitting or overfitting.

```
def feature(datum):
    feat = [
        1,  # Intercept
        float(len(datum['review_text'])),  # Review length
        float(datum['review_text'].count('!')),  # Exclamation count
        3 if datum['fit'] == "fit" else -3,  # Fit adjustment
        sum(word in datum['review_text'].lower() for word in
            ['love', 'great', 'amazing', 'perfect', 'wonderful']),  # Positive count
        sum(word in datum['review_text'].lower() for word in
            ['bad', 'poor', 'terrible', 'dislike', 'hate'])  # Negative count
    ]
    return feat
```

Fig 10. Final Feature List with Sentiment Analysis

To generalize the model, I compared the MSE across different iterations to minimize overfitting while maintaining prediction strength. That was the attempt in which the MSE was not improving at all and seemed to get worse than in my original linear model, so that’s where my experimentation with the polynomial model ended.

4. Literature

The dataset was adapted from *Decomposing Fit Semantics in Product Size Recommendation in Metric Spaces*, by Misra, Wan, and McAuley[4], a proposed predictive framework for determining product fit, using feedback from customers and the review text semantics. Metric learning was used to handle the label imbalance problem. With customer preference in play and a strong bias towards “fit,” (the label for an accurately-sized item according to a reviewer) ordinal regression can preserve the two other labels of “too small” and “too big” with the heuristic model to capture highly representative samples for these two traits. The proposed latent factor formulation was used to identify the key aspects of the customer reviews tuned with the metric learning approach that handled the unbalanced weighted fit labels.

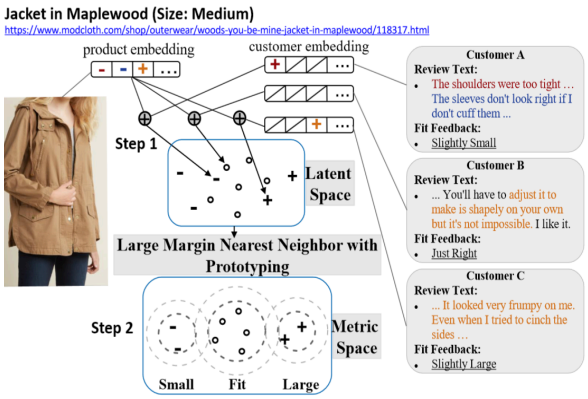

Fig 11. The workflow from "*Decomposing Fit Semantics.*.[5]"

The model was run on JSON-formatted data from both RentTheRunway and Modcloth[6], an online clothing retailer. The latter produced less fruitful results as the products got a lot fewer reviews due likely due to the mass-produced nature of the products on the site, which prompted me to run my own model on RTR, which had a much more significant amount of data for each item. Another study, *Examining collaborative filtering algorithms for clothing recommendation in e-commerce*, by Hu, Li, Wei, and Zhou,[7] showed the ability of similar models to improve customer satisfaction with recommendations to boost sales, using similar "massive historical records in the big data era."[8] They achieved high recommendation results using the co-occurrence matrix reduction method with the MovieLens[9] film recommendation website's dataset. We can see the propensity of the large data era for fine-tuning the recommendation problem based on vast numbers of genres, traits, features, and categories in order to achieve personalized, high-success rate for product endorsement to prospective customers. My results seem to enforce these ideas as the lower MSE with each iteration shows that the chosen data fields all correlate with either a high or low review score. In 2024, Vinitha *et al*. showed the latest attempt at perfecting the recommendation model, using a novel approach combining deep learning with machine learning, using collaborative filtering and matrix factorization for identifying the relationships between user and item[10]. Upon a repeat of this study, I would consider this approach of recurrent neural networks used in their design to better use the history created by each user in their reviews, as the study achieved high user satisfaction results. The established research I've uncovered seems to uphold my results; given the fact that machine learning models with access to this much data could provide even more accurate results with tuning and multiple iterations of modeling. One potential challenge for my analysis was narrowing down variability in user behavior, as some users tend to leave overly long reviews while others left little to no verbal feedback at all, skewing the results in my case.

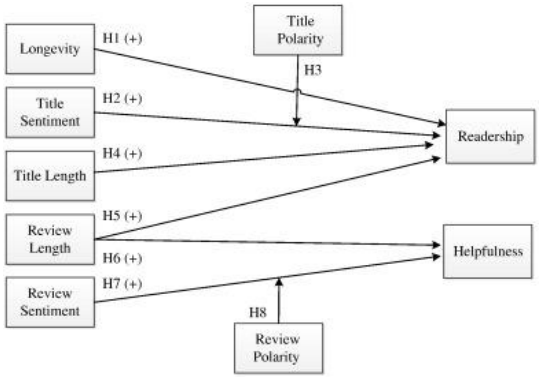

Fig 12. Model diagram from"*Predicting the performance…*[12]"

Other researchers solved this issue by normalizing review length, which could be a solution to my problem as seen in Salehan *et al.* who were able to also come to the conclusion of a positive influence of review length to a good review score in 2006 with *Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics.*[11] They were able to forego the differences in review length to exclusively perform sentiment analysis on the text.

5. Results

In this assignment we determined the length of the review as a second feature was actually a less important part of the linear regression model, as the baseline MSE was minimally improved with the exclamation point count feature, but the review_text length as a contributing factor towards prediction only marginally improved the MSE. We were able to predict between approximately 1 star accurately the score a review would receive just the features I did, so other future features may include size and shape of body when compared to fit especially on certain item categories like nightgowns that had relatively high rates of not fitting.[12] The addition of review text length was somewhat less impactful compared to the baseline MSE improvement achieved by including exclamation point count. While review text length did contribute to improving prediction accuracy, its effect was marginal at best. This shows that verbosity alone is not always prescriptive of sentiment, and other factors likely play a major role in determining

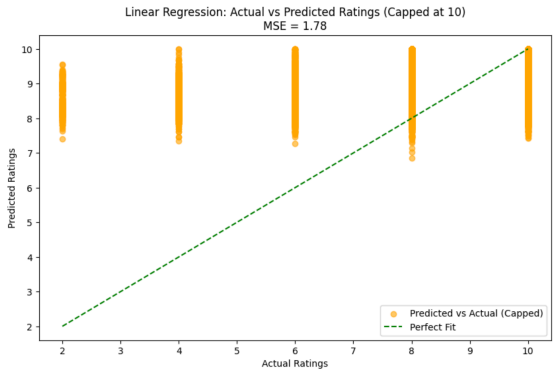

Fig 13. Actual vs. Predicted Ratings + MSE (Final Iteration)

review ratings. But its inclusion alongside other features provided a slight edge, indicating that lengthier reviews may signal higher engagement or stronger opinions, which are often associated with outlying higher and lower ratings. Even after capping my ratings at 10 to remove extreme values out of the 5 star range, my MSE remained at 1.78. The model achieved a prediction accuracy within approximately 1 star of the actual review score using only the features I tested. This tells us that while the features chosen provide a launchpad for prediction, there can be greater results achieved with better modeling. Future feature additions could include other punctuation counts, such as periods, commas, or question marks, which might reveal new sentiments. More potential avenues for improvement could involve leveraging more advanced text processing techniques, such as sentiment analysis or natural language processing, to decipher tone. By identifying nuanced phrases that reflect sentiment, the

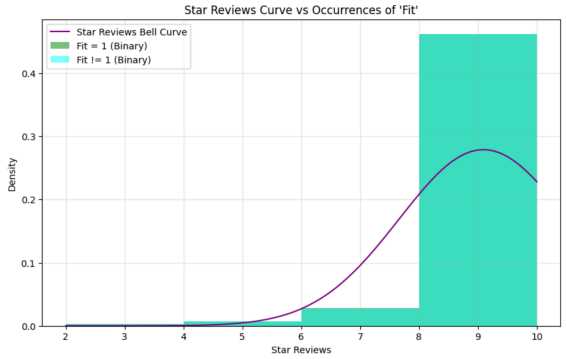

Fig 14. MatPlotLib Chart (Special Edition)

model could better extrapolate between reviews with similar lengths/punctuation counts but differing underlying contexts. Incorporating demographics such as the reviewer's age or height, could provide insights into how certain features correlate with preference. A promising direction would be to sample item-specific attributes, such as premium fabrics or designer branding, which may influence the rating positively. Exploring non-linear relationships through polynomial regression or even trees may capture new interactions between features. While the features used in this study allowed for reasonably accurate predictions, there remains significant potential for refinement to account for relationships not in the scope of this project to improve predictions of star ratings.

References

1. <https://www.renttherunway.com/>
2. <https://www.renttherunway.com/shop/designers/farm_rio/arabesque_floral_gown>
3. <https://cseweb.ucsd.edu/classes/fa24/cse258-b/files/homework1.pdf>
4. Rishabh Misra, Mengting Wan, Julian McAuley. *Decomposing fit semantics for product size recommendation in metric spaces.* RecSys, 2018. <https://cseweb.ucsd.edu/~jmcauley/pdfs/recsys18e.pdf>
5. *ibid.*
6. <https://www.modcloth.com>
7. Hu, Zhi-Hua, et al. *Examining Collaborative Filtering Algorithms for Clothing Recommendation in E-Commerce.* SagePub, 2006. <https://journals.sagepub.com/doi/abs/10.1177/0040517518801200>
8. *ibid.*
9. <https://movielens.org/>
10. Vinitha, M, et al. *A Fashion Recommendation System.* Irjaeh, 2024. <https://irjaeh.com/index.php/journal/article/view/204>
11. Salehan, M., & Kim, D. J. *Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics.* Science Direct, 2016. <https://www.sciencedirect.com/science/article/abs/pii/S0167923615002006>
12. RentTheRunway JSON Data <https://cseweb.ucsd.edu/~jmcauley/datasets.html>