Decomposing Reviews for Categorizing Clothing Products

Joshua Singer
Computer Science
UC San Diego
jjsinger@ucsd.edu

Aditya Moghe
Computer Engineering
UC San Diego
amoghe@ucsd.edu

Salwan Shathar
Computer Engineering
UC San Diego
sshathar@ucsd.edu

Jonny Tran
Computer Science
UC San Diego
jktran@ucsd.edu

ABSTRACT

Predicting the occasions that clothing items are worn for isn't necessarily the most straightforward task. With some basic review information, you can make a reasonably decent estimation of what occasion an item is for. In our exploratory analysis, we utilized text based reviews to predict what kinds of categories clothing items would be worn for. This method is by no means comprehensive and has much room for improvement, but it serves as a framework for what basic estimation methods should look like in the world of Recommender Systems.

KEYWORDS

Recommender Systems, TF-IDF, Bag-of-Words, Sklearn, Count Vectorize, Naive Bayes

1 Dataset Analysis

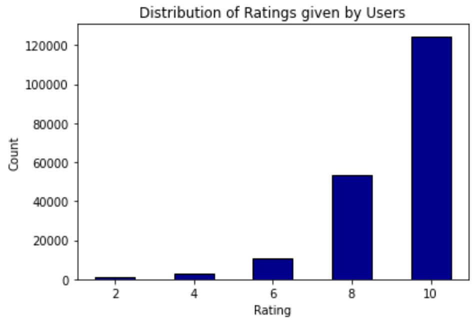
We introduced the dataset from Rent The Run Way[1], which includes clothing fit data. *RentTheRunWay* is a unique shop where women are able to rent clothes for various occasions. The dataset consists of around 190,000 entries, with each entry containing self-reported information, such as reviews, ratings, product categories, catalog sizes, customer's measures, etc. There were around 90,000 customers and around 6000 unique products.

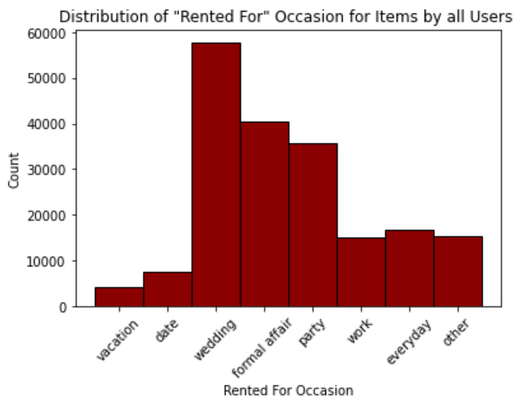
[1] Dataset is available at https://www.renttherunway.com/

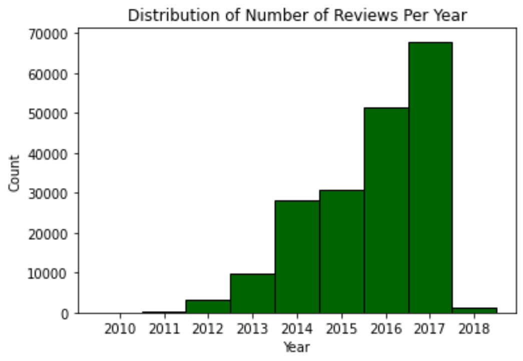
To understand the data better, we went ahead and observed the ratings, number of occasion categories, and the year distributions. There were eight categories for the rented occasions, with the following distributions:

Occasion Category	Count
Work	15,042
Date	7,387
Wedding	57,768
Party	35,613
Vacation	4,073
Formal Affair	40,365
Everyday	16,822
Other	15,381

To visually see the data, below we have histograms to display the distribution of entries for the ratings, the eight rented for occasions, and the years.







From all the entries, no odd rating was given; it only included even numbered ratings starting from 2 and up to 10. The reviews were also heavily skewed towards higher ratings, 8 and 10. There was one review in 2010, and the number of reviews gradually increased every year up until 2018, where the data collecting might have stopped early that year. For the rented occasion, even though there was some skew towards the wedding category, the data was fairly spread out among all categories.

With this information and analysis in mind, we proceeded to build our predictive model.

2 Predictive Task

The predictive task we decided to study on this dataset was to predict what event a piece of clothing was rented for based on the "review_text" and "review_summary". In other words, by using the user's text from the review, our model would predict the "rented for" category for each item. The model's validity will be assessed by computing the accuracy of our model using the validation data. This will be done by comparing the predictions from our model to the actual labels from the validation dataset and calculating the percentage of predictions that were correct. For each of our models, We compared the accuracy to that of the baseline model. The baseline model predicted the "rented for" category by seeing if the category appeared in the review text or summary. For example, if the word "vacation" appeared in the review, it would predict that the item was rented for a vacation. If none of the words appeared in the review, the baseline would randomly default to predicting either "wedding" or "other", since wedding was the most common event according to our exploration of the dataset. This baseline model had an accuracy of 0.3491 or 34.91%. As mentioned before, the features we will be using in our models are the "review_text" and "review_summary". Both of these features will be extracted directly from the reviews in the dataset and used by our models to predict the "rented for" category.

3 Model

In order to improve the accuracy of the baseline expectation (0.3491), we experimented with several models. Our approach was to leverage NLP techniques such as Bag-of-Words and TF-IDF, to find a model that would yield the highest accuracy. For all models that utilized Bag-of-Words features, we decided to start with a dictionary size of 1000, along with a regularization strength of C=0.7. We first created a model that leveraged the Sklearn library, SVM.LinearSVC, and trained on a Bag-of-Words feature vector. This implementation resulted in an accuracy of 0.572928. Next, we experimented with a logistic regression model that was also trained on a Bag-of-Words feature vector. This implementation yielded an accuracy of 0.572979. Despite using different models, we found that a Bag-of-Words approach would produce similar accuracies with minimal improvements. Given this finding, we experimented with CountVectorizer, which is Sklearn's library implementation of Bag-of-Words. After running a logistic regression model with regularization strength C=0.7, we found the accuracy to be 0.582229. We accounted for this improvement in accuracy because of the robustness of the Sklearn library in comparison to the Bag-of-Words feature function we encoded. We tried to further modify this CountVectorizer approach by using a Naive Bayes over a logistic regression model for our prediction. The Naive Bayes model lowered our accuracy to 0.56721. Once we had exhausted the models built from a Bag-of-Words approach, we decided to switch approaches and use Sklearn TfidfVectorizer. By using this library on a logistic regression model with regularization strength C=0.7, we found that our accuracy improved to 0.595123. From our basic experiments we found that the TF-IDF approach produced better accuracies than Bag-of-Words. In order to fine tune the TfidfVectorizer based model, we first tried to find a more optimal regularization strength. After running multiple iterations, we found that C = 0.755 produced the higher accuracy of 0.59516. Furthermore, we also tried to remove stop words from our text, but found that this only reduced accuracy. Therefore our final model used logistic regression with regularization strength C = 0.755 on a TfidfVectorizer.

When observing the various model testing, we were able to see that TfidfVectorizer was far superior than a Bag-of-Words approach. Regardless of how much we changed the dictionary size for BOW models, they were never able to exceed the accuracy of our Tfidfvectorizer. Because Tfidfvectorizer gives the most weight to important words, it always held the highest accuracy in all our tests.

Model	Accuracy
Baseline	0.3491
BOW SVM.LinearSVC	0.572928
BOW Logistic Regression	0.572979
CountVectorizer	0.582229
Naive Bayes	0.56721
TfidfVectorizer	0.59516

4 Related Literature

The clothing reviews dataset has previously been used before by Julian McAuley et al. [1] to predict fit for the individual customer. Their aim was to improve customer satisfaction in online shopping by utilizing previous customer feedback. They found that using a K-dimensional latent factors metric learning model was the most successful in predictions on their data set. The authors believe this success is due to the metric learning approach.

Another approach to analyzing the Rent the Runway dataset was done by Sheikh et al. [2]. This research group attempted to find correct fits for customers so that customer satisfaction could increase and returns would decrease. They split the data set in the same way that we did with 80% for training and 10% for both validation and testing. In their method, they handled cold-start cases by having a default feature which was randomly assigned. Their work was vastly different from our own because they used neural networks as their baseline. Furthermore, they used actual people to test the results generated by the algorithm.

Previous research has explored the effectiveness of TF-IDF algorithms and sought to make improvements to the existing TF-IDF algorithm [3, 4]. Joachims' research [4] explores the effectiveness of TF-IDF in relation to other algorithms such as Naive Bayes and finds that Naive Bayes is a more well founded algorithm that tends to produce higher accuracies. Joachims' research [4] cites a similar problem that we explored when using TF-IDF in that there are a lot of different choices to be made in selecting features and there isn't much way to tell which choices are the correct one.

Additionally, it suggests a different PrTFIDF algorithm which helps you decide which features to include. The main advantage of this algorithm is that it isn't harder to understand or more computationally complex and still tends to perform better on average. Liu and Yang [3] seek to make improvements upon existing TF-IDF algorithms by using a new "in-class characteristic" parameter. This creates features vectors using the TF-IDF algorithm but instead uses term frequencies within a single class. Their models found significant improvements in accuracy, so they found them to be highly effective.

5 Results and Conclusions

While the accuracies did not vary drastically among the models, some did prove to be slightly better than others. Parameters that were included in our model included: lower case, dictionary size, C values, and stop words. We found that the best way to improve our Bag-of-words model was to increase the dictionary size until we reached about 10,000 words. Then there was little to no improvement of accuracy. Changing all letters to lowercase and removing stop words had little effect on the accuracy of our results, so we decided to not remove these from our text. Furthermore, we found that lower C values had improved accuracy, but we chose a more moderate C value to prevent overfitting. Furthermore, we chose a moderate amount of 10,000 for a dictionary size to prevent overfitting and make our model more general. Our Bag-of-words model is really simple to implement and understand while it doesn't account for document frequencies like TF-IDF does. This method could unintentionally give high weights to stop words and such which may not have any added semantic meaning.

TF-IDF worked better than Bag-of-Words because the TF-IDF model puts more weight on the more important words and less weight on the lesser important words. The Bag-of-Words just converted the words into frequency counts with no semantic information, whereas TF-IDF converted the words into vectors with weighted information.

While our TF-IDF Vectorizer (best model) has a seemingly low accuracy of 59.516%, we still believe it to be a good model given the large amount of labels in our dataset. For example, a suit could be for multiple occasions, "wedding" or "formal affair", but it would still be correctly classified in either category. However, our algorithm isn't developed enough to catch these small distinctions, so it could be labeled as a false positive.

While our results drastically beat our baseline model, we still believe that there are potentially better models to be explored. For example, deep learning was referenced as a potential solution to learn categorization tasks without being given labels. Future research could involve using Deep Learning to strengthen our results.

ACKNOWLEDGMENTS

Professor Julian McAuley

REFERENCES

[1] Rishabh Misra, Mengting Wan, and Julian McAuley. 2018. Decomposing fit semantics for product size recommendation in metric spaces. In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18). Association for Computing Machinery, New York, NY, USA, 422–426. https://doi.org/10.1145/3240323.3240398

[2] Abdul-Saboor Sheikh, Romain Guigourès, Evgenii Koriagin, Yuen King Ho, Reza Shirvany, Roland Vollgraf, and Urs Bergmann. 2019. A deep learning system for predicting size and fit in fashion e-commerce. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys '19). Association for Computing Machinery, New York, NY, USA, 110–118. https://doi.org/10.1145/3298689.3347006

[3] Liu, M. and Yang, J. (2012) "An improvement of TFIDF weighting in text categorization ," *2012 International Conference on Computer Technology and Science (ICCTS 2012)* [Preprint]. Available at: https://doi.org/10.7763/IPCSIT.2012.V47.9.

[4] Thorsten Joachims. 1997. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 143–151.