CYOProject

Joon Young Jang

2024-03-17

Introduction

Consumer reviews are an important part of the e-commerce experience and their impact on consumer behavior is notable. That is, while positive reviews may encourage the consumers to purchase an item, negative reviews may forestall that decision. This report therefore will use a publicly available dataset to predict consumer ratings of items from Sephora, a personal care product retailer. The data archive collected and assembled by Raghad Alharb contains 9169 rows (9168 items total) and 21 columns offering relevant information for each item. To prevent overfitting, 80% of the dataset was partitioned into the train set and the remaining 20% was reserved for the test set. This data table will be visualized into graphs and charts to further study its features. Thereafter, knn, random forest and support vector machine was adopted to model ratings of these beauty products.

Data Visulization

Even though the dataset is polished for machine learning and comes with clean values, the dataset was tailored and attuned to fit the needs of this project. Columns including ingredients and size that are not useful for the prediction of ratings were removed. Limited time offer was also deleted as only three products were indeed offered on a limited time basis. The remaining columns, or predictors, that will be utilized to predict ratings are the following: brand, category, number of reviews, love, price, value price, online only, exclusive, and limited edition. The name column will be kept to distinguish each item.

There are 9168 rows (reviews) and 11 columns (9 predictors, 1 rating, 1 name). The dataset comprises 324 brands and 143 categories. These statistics provide an indication of the substantial volume and variation of data that must be processed.

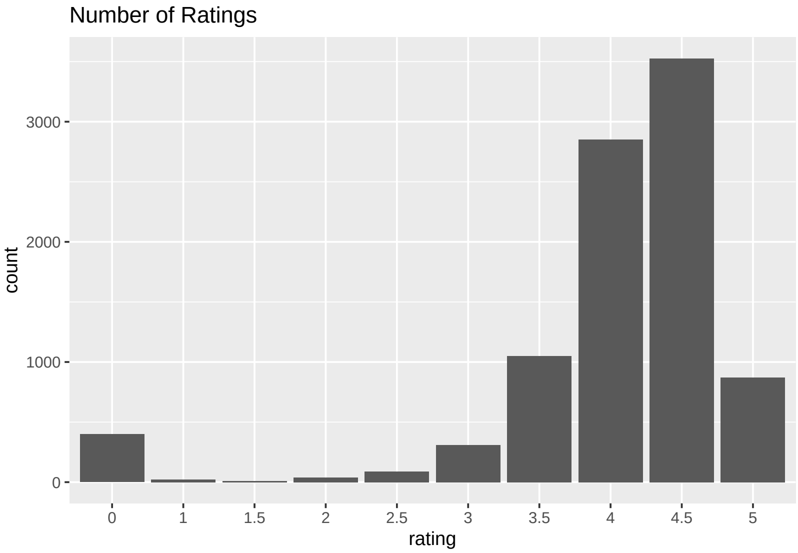
The reviews dataset now looks like such:

```
## # A tibble: 6 x 11
##   brand          category name  rating number_of_reviews  love price value_price
##   <fct>          <fct>    <chr> <fct>              <dbl> <dbl> <dbl>       <dbl>
## 1 Acqua Di Parma Fragran~ Blu ~ 4                      4  3002    66          75
## 2 Acqua Di Parma Cologne  Colo~ 4.5                   76  2700    66          66
## 3 Acqua Di Parma Perfume  Aran~ 4.5                   26  2600   180         180
## 4 Acqua Di Parma Perfume  Mirt~ 4.5                   23  2900   120         120
## 5 Acqua Di Parma Fragran~ Colo~ 3.5                    2   943    72          80
## 6 Acqua Di Parma Perfume  Fico~ 4.5                   79  2600   180         180
## # i 3 more variables: online_only <fct>, exclusive <fct>, limited_edition <fct>
```

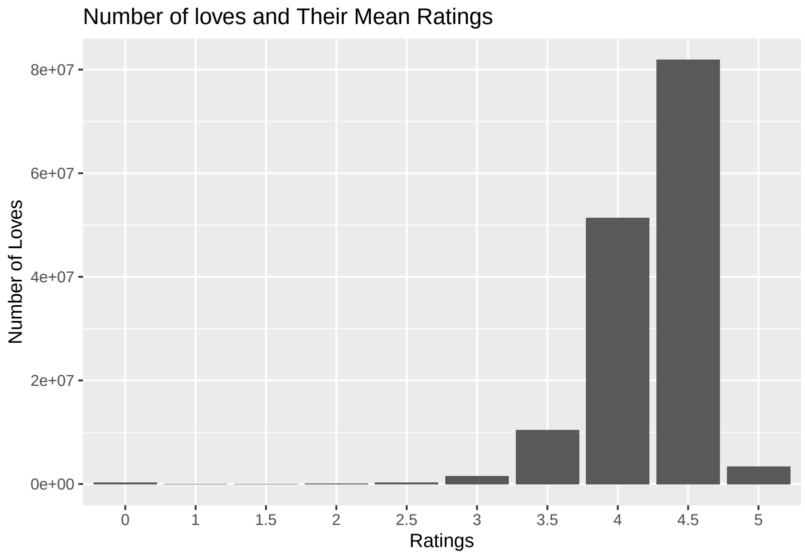
The brand column contains the names of the brands each items are from. The category column groups the products into categories such as perfume, face masks or shampoo. The number of reviews column contains the number of reviews each product has. The love column counts how many customers bookmarked an item for potential future purchase. The price value shows the original product price while value price is the final value of the product after discounts. Online only, exclusive and limited edition are factor columns that are either 1 or 0. If the column name applies to an item, that item is a 1 and a 0 if not so. For example, if an item is exclusive only on Sephora, that item is a 1 in the exclusive column. Most importantly, the ratings column is also of a factor class with the following levels: 1, 1.5, 2, 2.5, 3, 3.5, 4.5, and 5. 5 indicates that customers were extremely satified with the product while 1 is left by a customer to express discontent and disappointment.

```
## List of 11
##  $ brand            : int(0)
##  $ category         : int(0)
##  $ name             : int(0)
##  $ rating           : int(0)
##  $ number_of_reviews: int(0)
##  $ love             : int(0)
##  $ price            : int(0)
##  $ value_price      : int(0)
##  $ online_only      : int(0)
##  $ exclusive        : int(0)
##  $ limited_edition  : int(0)
```

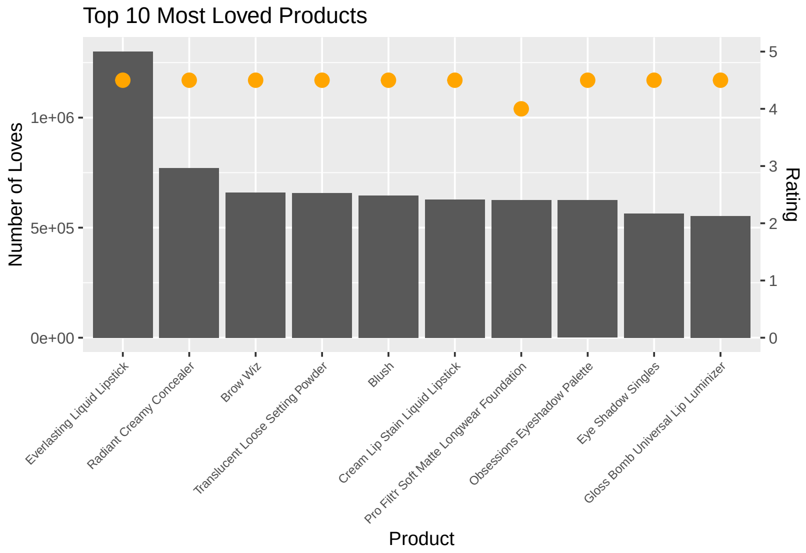
As such, there aren't any missing values in the final dataset and is ready to be visuzlied and modeled.



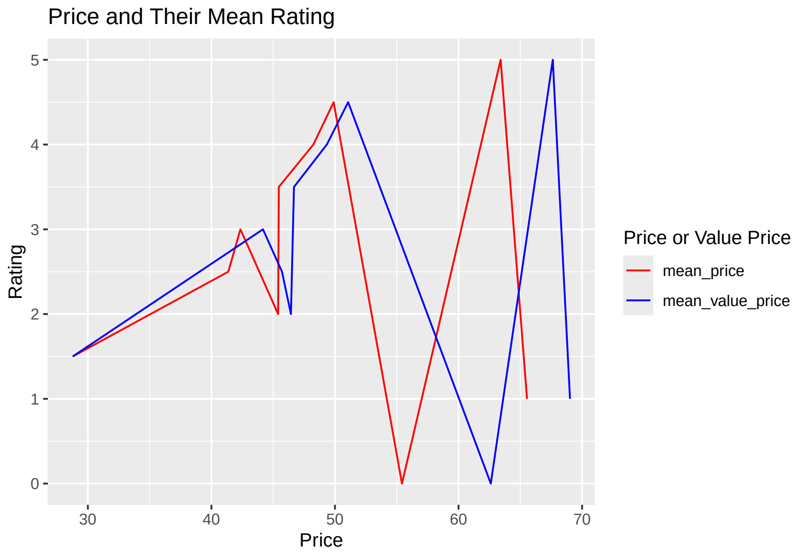
The plot illustrates the overall distribution of the total number of ratings, with the mode situated high up approximately between 4 and 4.5.



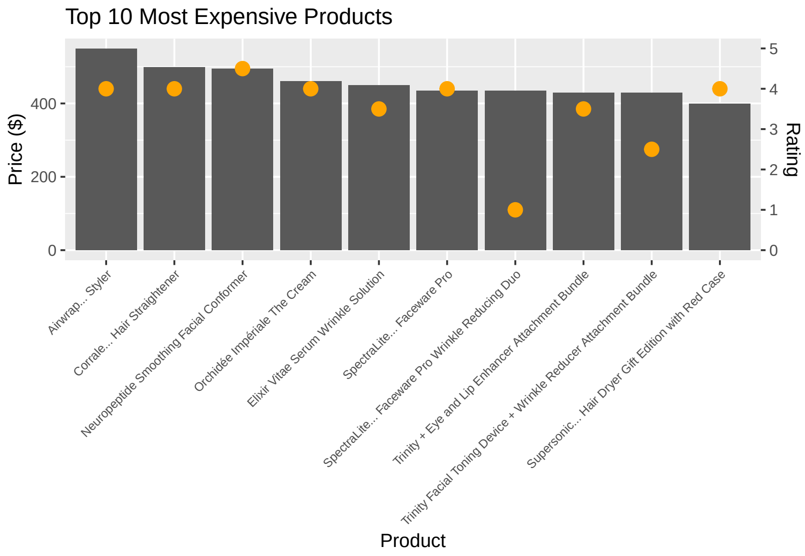
The plot describes the total number of loves for rating class. The mode is located again at 4~4.5.



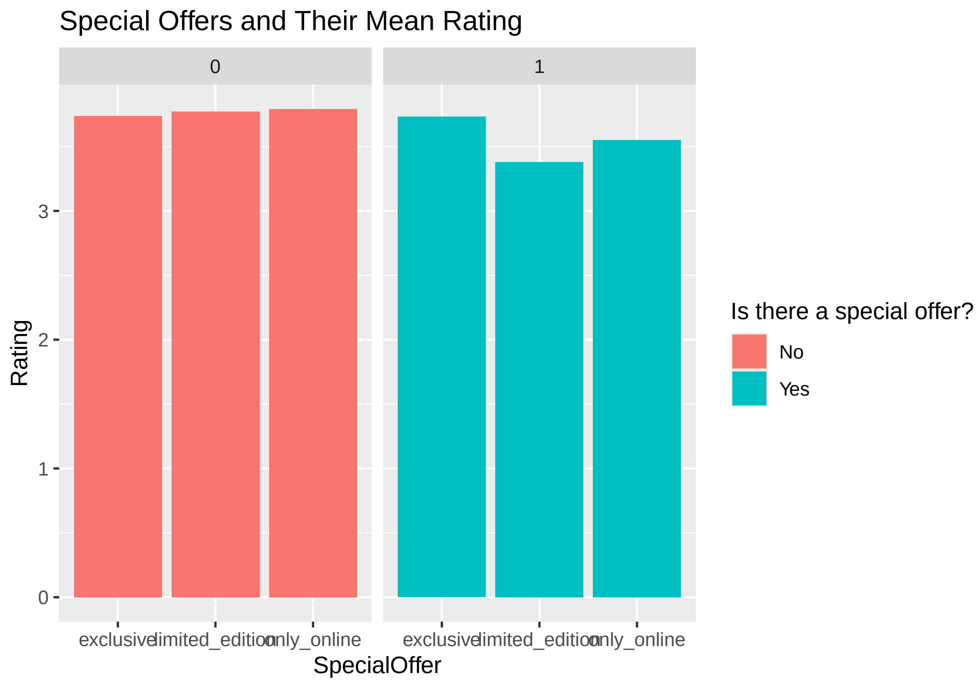
The most loved beauty products were consistently rated at 4 or higher which is suggestive of a greater correlation between these two variables.



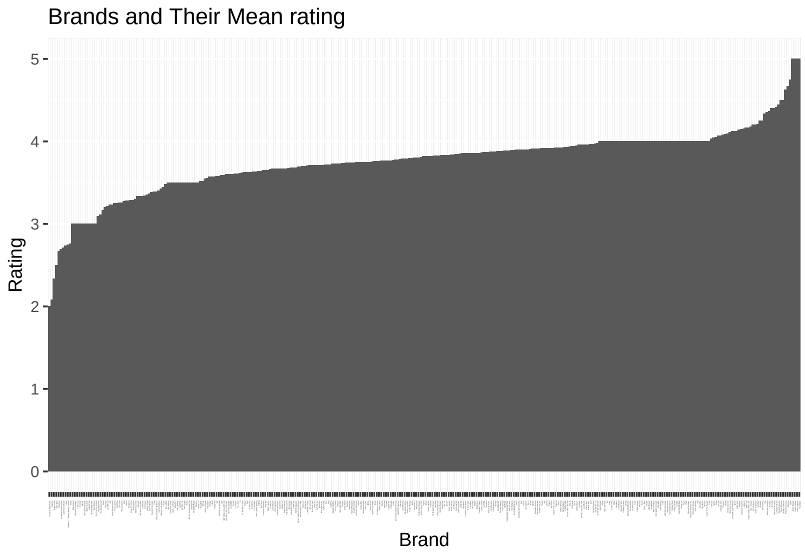
The price plots tell a similar story as well with a similar distribution and a peak at once again 4~4.5 stars. Price and value price both seem to have a similar relationship with ratings as well.



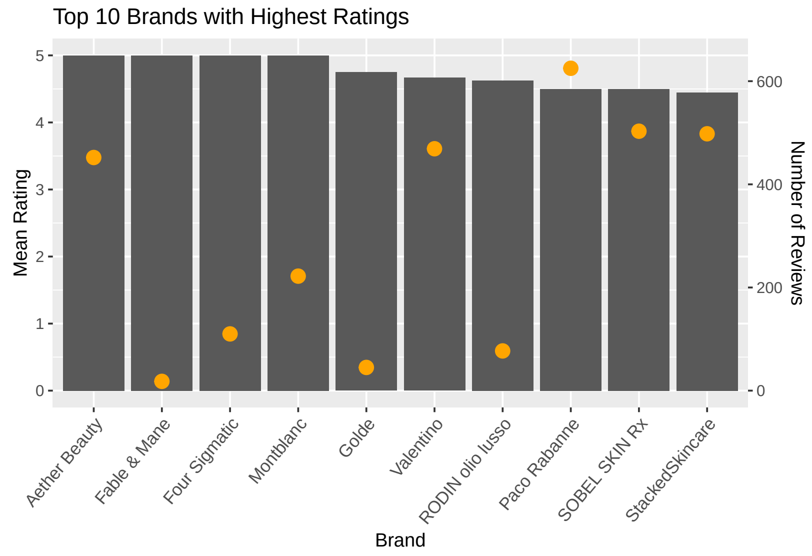
In contrast to the top 10 most loved products, a consistent high rating could not be observed as only one product was rated 4.5 and the rest were rated below 4.



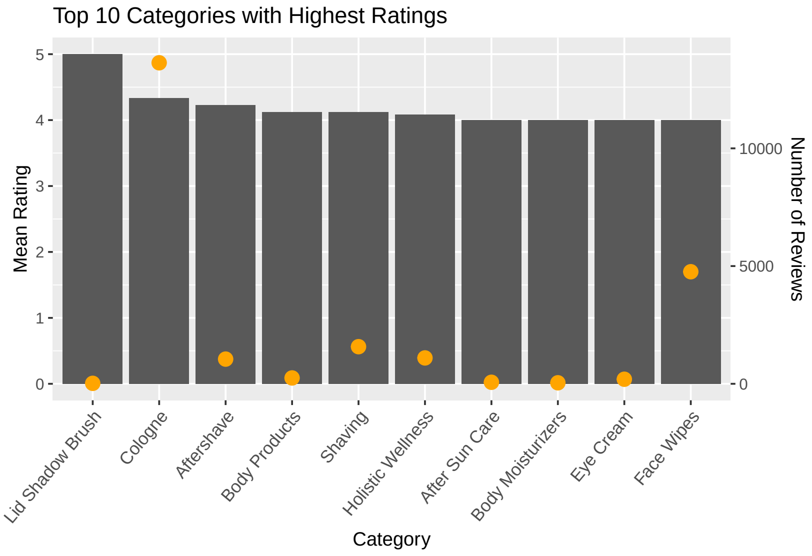
The bar graph above depicts the relationship between special promotions and ratings. Counter-intuitively, the ratings were higher when the special promotion was not available.



This plot helps us visualize how each brands were rated according to customers.



In a similar vein, this plot zooms into the top 10 brands with the highest mean ratings were ranked along with the number of reviews they received each. The top four brands all had perfect ratings but Fable & Mane only received 18 reviews while Aether Beauty had 452.



Cologne ranked second with an impressive rating of 4.34 while also being fairly popular with 13635 reviews. Though Lip Shadow Brush came in first place, it seems to be a niche category with only 20 reviews.

```
#Set seed and partition into train/test sets
reviews <- reviews |> mutate(brand = str_replace(reviews$brand, "SEPHORA COLLECTION", "'Sephora Collecti

set.seed(3)
test_ind <- createDataPartition(reviews$rating, times = 1, p = 0.2, list = FALSE)
test_set <- reviews[test_ind,]
train_set <- reviews[-test_ind,]

#Remove rows that are unique to the training set
unmatched_brand <- anti_join(train_set, test_set, by = "brand")
unmatched_category <- anti_join(train_set, test_set, by = "category")
```

```
train_set <- train_set[-(which(train_set$brand %in% unmatched_brand$brand)),]
train_set <- train_set[-(which(train_set$category %in% unmatched_category$category)),]

#Remove rows that are unique to the test set
unmatched_brand_test <- anti_join(test_set, train_set, by = "brand")
unmatched_category_test <- anti_join(test_set, train_set, by = "category")

test_set <- test_set[-(which(test_set$brand %in% unmatched_brand_test$brand)),]
test_set <- test_set[-(which(test_set$category %in% unmatched_category_test$category)),]
```

Model/Anlaysis

The dataset was divided into a training set and a test set to evaluate the models performance and to avoid over-fitting. A seed value of two was assigned to ensure the reproducibility of the R code.

KNN Model

```
#Set parameters for models
control <- trainControl(method = "cv", number = 10)
```

```
#Model 1 - KNN
model_knn <- train(rating ~ brand + category + number_of_reviews + love + price + value_price + online_
                   method = "knn",
                   data = train_set,
                   trControl = control,
                   tuneLength = 10)

model_knn$results
```

```
##     k  Accuracy     Kappa AccuracySD    KappaSD
## 1   5 0.3783514 0.1186149 0.01702923 0.01999458
## 2   7 0.3813011 0.1138181 0.01931446 0.02740972
## 3   9 0.3821444 0.1108041 0.01430182 0.01994049
## 4  11 0.3887481 0.1157910 0.02198190 0.03080165
## 5  13 0.3914174 0.1192613 0.01506337 0.02114013
## 6  15 0.3996998 0.1283682 0.01523889 0.02147119
## 7  17 0.3957639 0.1215902 0.02017133 0.02857172
## 8  19 0.4022255 0.1299498 0.02154089 0.03158877
## 9  21 0.4055940 0.1339353 0.02000986 0.02917576
## 10 23 0.4034851 0.1307675 0.02002216 0.02862929
```

```
accuracy_knn <- confusionMatrix(predict(model_knn, test_set), test_set$rating)$overall["Accuracy"]
accuracy_knn
```

```
##  Accuracy
## 0.3816293
```

```
accuracy_list <- tibble(model = "KNN", accuracy = accuracy_knn)
```

The knn model yielded a disappointing accuracy of 0.3816293.

Random Forest Model

```
#Model 2 - Random Forest
#This model may take a few minutes to run.
model_rf <- train(rating ~ brand + category + number_of_reviews + love + price + value_price + online_o
                  data = train_set,
                  trControl = control,
                  method = "rf")

model_rf$results
```

```
##   mtry  Accuracy     Kappa   AccuracySD    KappaSD
## 1    2 0.3849514 0.0000000 0.0008146673 0.00000000
## 2  210 0.4889951 0.2599361 0.0191602093 0.02998147
## 3  419 0.4856189 0.2560489 0.0184192513 0.02929924
```

```
accuracy_rf <- confusionMatrix(predict(model_rf, test_set), test_set$rating)$overall["Accuracy"]
accuracy_rf
```

```
##  Accuracy
## 0.4980864
```

```
accuracy_list <- accuracy_list |> add_row(model = "RF", accuracy = accuracy_rf)
```

The random forest model fared better at an accuracy of 0.4980864.

```
#Model 3 - Support Vector Machine
#This model may take a few minutes to run.
model_svm <- train(rating ~ brand + category + number_of_reviews + love + price + value_price + online_
                  data = train_set,
                  trControl = control,
                  method = "svmRadial")

model_svm$results
```

```
##          sigma    C  Accuracy      Kappa  AccuracySD    KappaSD
## 1 6.584124e-07 0.25 0.4015239 0.10022186 0.008601176 0.01090774
## 2 6.584124e-07 0.50 0.3960460 0.09620946 0.010007514 0.01490198
## 3 6.584124e-07 1.00 0.3940848 0.09850515 0.012394824 0.01906640
```

```
accuracy_svm <- confusionMatrix(predict(model_svm, test_set), test_set$rating)$overall["Accuracy"]
accuracy_svm
```

```
##  Accuracy
## 0.4029524
```

```
accuracy_list <- accuracy_list |> add_row(model = "svm", accuracy = accuracy_svm)
```

The accuracy for the support vector machine model is 0.4029524.

Result

```
## # A tibble: 3 x 2
##   model accuracy
##   <chr>    <dbl>
## 1 KNN      0.382
## 2 RF       0.498
## 3 svm      0.403
```

The cumulative table concludes that the random forest model is superior algorithm when it come to accuracy in predicting ratings compared to other models.

Conclusion

This report explored the Sephora product dataset and implemented various non-linear algorithms for rating prediction. This public dataset was first studied and dissected the given predictors before priming the table for data training. The remaining columns were visualized onto graphs, illuminating their complexities. The seemingly unrelated nature of variables observed during this step of the process could have served as cautionary signs that the chosen features may not be strong predictors of customer ratings. In addition, computational limitations during this study restricted the exploration of more complex models. All in all, the accuracy of these two models are underperforming and suboptimal. We recommend future research to explore the relationships between the chosen variables and customer ratings in more detail. This could involve feature engineering to create new features or investigating the use of more advanced machine learning techniques. Despite the underperformance of the models in this study, this report serves to inform other data scientists about the potential of this dataset and to promote a deeper understanding of consumer behavior in the beauty product market.

Credits to Alharbi (n.d.), Chen et al. (2022) and Rafael (2019) for providing the data set and further context throughout the report.

References

Alharbi, Raghad. n.d. "Sephora Website." https://www.kaggle.com/datasets/raghadalharbi/all-products-available-on-sephora-website.

Chen, Tao, Premaratne Samaranayake, XiongYing Cen, Meng Qi, and Yi-Chen Lan. 2022. "The Impact of Online Reviews on Consumers' Purchasing Decisions: Evidence From an Eye-Tracking Study." *Frontiers in Psychology* 13 (June). https://doi.org/10.3389/fpsyg.2022.865702.

Rafael, Irizarry. 2019. *Advanced Data Science.* https://rafalab.dfci.harvard.edu/dsbook-part-2/.