Assignment 01 Report

CS-2490, Advanced Machine Learning

Spring 2022

Kuber Shahi

Mar 9, 2022

Contents

1 Part 1: Recommendation System based on User vs Product matrix

I chose the **Beauty Products Dataset** to work on for part 1 and 2. The dataset is about reviews and ratings given by users on different beauty products. It has 5269 rows and 12 columns. The columns give information about the user, product, reviews and rating of the product, and time at which the reviews and ratings were provided.

	overall	verified	reviewTime	userID	productID	style	reviewerName	reviewText	summary	unixReviewTime	vote	image	One
0	5	True	09 1, 2016	A3CIUOJXQ5VDQ2	B0000530HU	{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice ...	Shelly F	As advertised. Reasonably priced	Five Stars	1472688000	NaN	NaN	1
1	5	True	11 14, 2013	A3H7T87S984REU	B0000530HU	{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice ...	houserules18	Like the oder and the feel when I put it on my...	Good for the face	1384387200	NaN	NaN	1
2	1	True	08 18, 2013	A3J034YH7UG4KT	B0000530HU	{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice ...	Adam	I bought this to smell nice after I shave. Wh...	Smells awful	1376784000	NaN	NaN	1

Figure 1: An overview of the dataset

Beside that the key thing to note about the dataset is that it has only only **991 unique users** and **85 unique products**.

1.1 Approach:

i) To get the user-product matrix, I used the pandas pivot table method. Users were placed in the rows and products were placed in the columns and each cell in the matrix counted to number of time the particular user had bought the particular product.

productID userID	B0000530HU	B00006L9LC	B00021DJ32	B0002JHI1I	B0006O10P4	B0009RF9DW	B000FI4S1E	B000FOI48G	B000FTYALG	B000GLRREU	..
A105A034ZG9EHO	0	0	0	0	0	1	1	0	0	0	..
A10JB7YPWZGRF4	0	0	0	0	0	1	1	0	0	0	..
A10M2MLE2R0L6K	0	0	0	0	0	0	0	0	0	0	..
A10P0NAKKRYKTZ	0	0	0	0	0	1	1	0	0	0	..
A10ZJZNO4DAVB	0	1	0	0	0	0	0	0	0	0	..
...	...	...	...	...	...	...	...	...	...	...	..
AZCOSCQG73JZ1	0	1	0	0	0	0	0	0	0	0	..
AZD3ON9ZMEGL6	0	0	0	0	0	1	1	0	0	0	..
AZFYUPGEE6KLW	0	1	0	0	0	0	0	0	0	0	..
AZJMUP77WBQZQ	0	1	0	0	0	0	0	0	0	0	..
AZRD4IZU6TBFV	0	0	0	0	0	1	1	0	0	0	..

991 rows × 85 columns

Figure 2: An overview of User-Product matrix

The dimension of the matrix is 991 x 85 which means the total number of cell is 84235 but only 4092 of them are filled which is around 5% of the data. The user-product matrix is really sparse as shownn in the 3

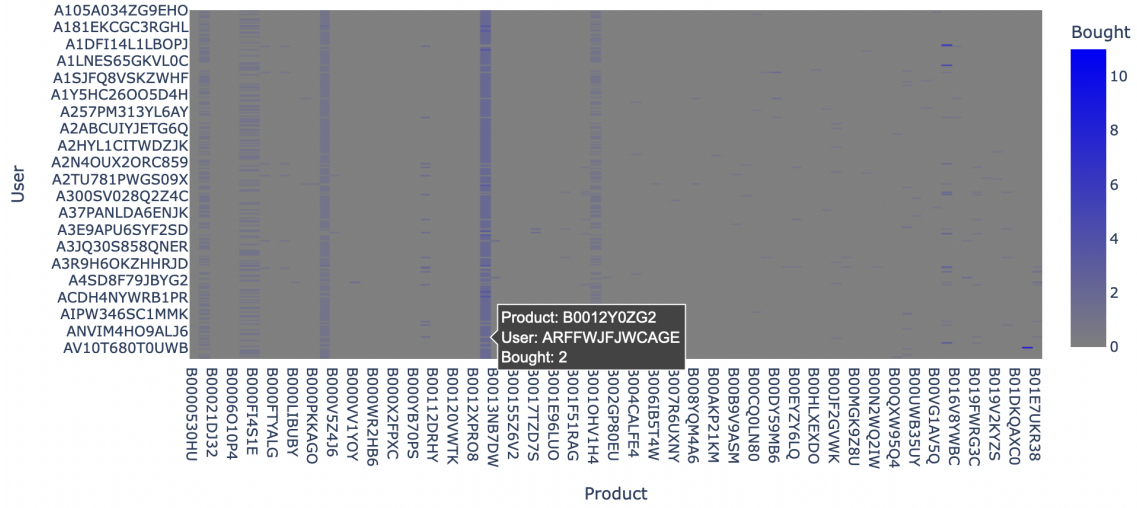


Figure 3: User-Product Matrix heatmap

The blue patches means the cell is filled and the gray patches mean the cell isn't filled for the corresponding user and the product.

ii) Then, I applied **PCA** to reduce the number products as dimension in the matrix and brought down the number of columns to 3.

	userID	f1	f2	f3
0	A105A034ZG9EHO	0.785026	-0.256283	-0.025307
1	A10JB7YPWZGRF4	0.785026	-0.256283	-0.025307
2	A10M2MLE2R0L6K	0.669168	1.120522	-0.902591
3	A10P0NAKKRYKTZ	0.785026	-0.256283	-0.025307
4	A10ZJZNO4DAVB	-0.775713	-0.017964	-0.005120
...	...	...	...	...
986	AZCOSCQG73JZ1	-0.775713	-0.017964	-0.005120
987	AZD3ON9ZMEGL6	0.785026	-0.256283	-0.025307
988	AZFYUPGEE6KLW	-0.775713	-0.017964	-0.005120
989	AZJMUP77WBQZQ	-0.775713	-0.017964	-0.005120
990	AZRD4IZU6TBFV	0.785026	-0.256283	-0.025307

991 rows × 4 columns

Figure 4: User-Product Matrix after PCA

iii) Next, I used KMeans clustering method to find the clusters. Using elbow method, I got 4 as the right number of clusters to take. The result of clustering in shown in figure 5.

iv) Finally, to recommend products to a user, we first find the products bought by the given user and all the products bought by other users from the cluster the given user belongs to and then, subtract both the sets.

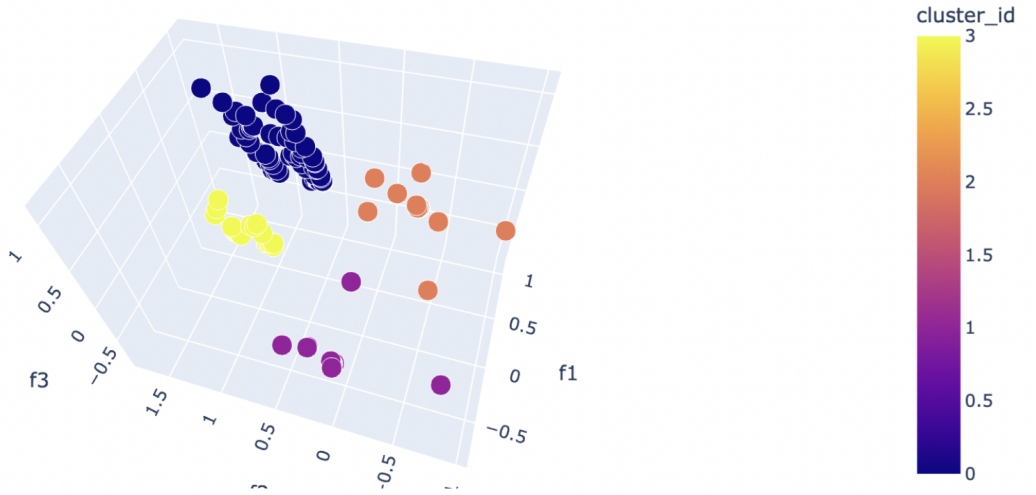


Figure 5: User-Product Cluster

1.2 Result and Recommendation

As shown in the picture 5, the users were segmented in four different clusters. The details of the clusters are as shown in the table below:

Cluster ID	Number of Users	Number of Unique Products
0	83	49
1	489	27
2	386	19
3	33	21

Table 1: User-Product Cluster Details

The number of products bought by users is clusters are [49, 27, 19, 21] which seems reasonable given there are 85 total products and the average number of products bought by users is around 5. Two of the clusters have high number of users - that is cluster 1 and cluster 2.

As suggested in the point (iv) of previous section (1.1), the recommendation for a user are the remaining products bought by other users from the cluster the given users belongs to. For instance, the recommendation for user **A105A034ZG9EHO** are 15 products with IDs [B01DKQAXC0, B000PKKAGO, B006WYJM8Y, B006IB5T4W, B00IJHY54S, B00112DRHY, B001E5PLCM, B000WYJTZG, B001OHV1H4, B00AKP21KM, B000X7ST9Y, B00CQ0LN80, B00021DJ32, B00HLXEXDO, B0014SQQ3M].

The clustering of the users based on the products they bought is the key to our recommendation system. Since KMeans is an unsupervised learning approach for clustering, the only way to evaluate the correctness of the clustering is the plot the points with the clusters. As you can see in the Figure 5, the clusters looks properly segregated giving some sense of robustness to our recommendation system.

2 Part 2: Recommendation System based on User vs Cluster matrix

For part 2, I followed the same approach as done in part 1. The key difference here is the method I used to obtain the final clustering on users. First, I clustered the products based on their description text and then for each user, I mapped the number of products the user has bought from each cluster giving us a final user-cluster matrix.
To cluster both, products based on their description text and then user based on the cluster matrix, I used the KMeans clustering technique.

2.1 Approach:

i) First, I got a new pandas dataframe with 85 unique product and their descriptions. There were 39 products with no description at all which is almost 46% of the products.

	productID	style
0	B0000530HU	{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice ...
1	B00006L9LC	{'Size:': ' 200ml/6.7oz'}
2	B00021DJ32	{'Color:': ' Super Orgasm'}
3	B0002JHI1I	NaN
4	B0006O10P4	{'Size:': ' 3 oz.', 'Scent Name:': ' Frankince...
...	...	...
80	B00UWB35UY	NaN
81	B00VARTPKS	NaN
82	B00VG1AV5Q	NaN
83	B019LAI4HU	NaN
84	B019V2KYZS	NaN

85 rows × 2 columns

Figure 6: Product-Description Dataframe

ii) Next, using NLTK package, I removed punctuation and stop words from the production description text.

```
85
["{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice Blue'}"
 "{'Size:': ' 200ml/6.7oz'}" "{'Color:': ' Super Orgasm'}" 'nan'
 "{'Size:': ' 3 oz.', 'Scent Name:': ' Frankincense & Myrrh'}"
 "{'Size:': ' Multiset'}" 'nan'
 "{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice Blue'}"
 "{'Size:': ' Ultra'}" 'nan' 'nan' 'nan' "{'Size:': ' 46'}" 'nan'
 "{'Color:': ' Black Natural and Navy'}"
 "{'Size:': ' 1 count', 'Color:': ' Apricot Cuticle Oil'}" 'nan'
 "{'Size:': ' 1 oz'}" "{'Size:': ' 3/4 Inch'}" "{'Size:': ' 32 oz'}" 'nan'
 "{'Size:': ' LG X 32'}" "{'Color:': ' Sea Sponge'}"
 "{'Scent Name:': ' Sandalwood'}" 'nan' "{'Size:': ' 2 oz'}" 'nan'
 "{'Color:': ' Blueberry'}" "{'Size:': ' 6.8 oz'}" "{'Size:': ' 7 Ounce'}"
 'nan' 'nan' 'nan' 'nan'
```

Figure 7: Product Description Cleaned Array

iii) Then, using TFIDF Vectorizer, I converted to description text into usable vector form. The vocabulary length of the data was 90. Hence, at the end I had a matrix of 85 × 90 dimension.

iv) Next, using KMeans, I clustered the products based on their description text into three clusters (number of clusters to be taken was found through elbow method).

	productID	cluster_id	style
0	B0000530HU	1	{'Size:': ' 7.0 oz', 'Flavor:': ' Classic Ice ...
1	B00006L9LC	1	{'Size:': ' 200ml/6.7oz'}
2	B00021DJ32	2	{'Color:': ' Super Orgasm'}
3	B0002JHI1I	0	NaN
4	B0006O10P4	1	{'Size:': ' 3 oz.', 'Scent Name:': ' Frankince...
...	...	...	...
80	B00UWB35UY	0	NaN
81	B00VARTPKS	0	NaN
82	B00VG1AV5Q	0	NaN
83	B019LAI4HU	0	NaN
84	B019V2KYZS	0	NaN

Figure 8: Product Description with cluster id

The first cluster had 39 products - all 39 products with no description text were clubbed together. The second and third clusters had 34 and 12 products respectively.

v) Then, to create the user-cluster matrix, for each user, I mapped the number of products the user had bought in each cluster. Since there were 3 clusters from previous part and 991 unique users. I got a final user-cluster matrix of 991×3 dimension.

	userID	0	1	2
0	A105A034ZG9EHO	1	3	0
1	A10JB7YPWZGRF4	1	3	0
2	A10M2MLE2R0L6K	0	0	4
3	A10P0NAKKRYKTZ	1	3	0
4	A10ZJZNO4DAVB	0	4	0
...	...	...	...	...
986	AZCOSCQG73JZ1	0	4	0
987	AZD3ON9ZMEGL6	1	3	0
988	AZFYUPGEE6KLW	0	4	0
989	AZJMUP77WBQZQ	0	4	0
990	AZRD4IZU6TBFV	1	3	0

991 rows × 4 columns

Figure 9: User-Cluster Matrix

vi) Next, I applied KMeans on user-cluster matrix and got three clusters (number of cluster to be taken was found through elbow method). The results of the clustering is shown in the Figure 10.

vii) Finally, as done in part 1, to recommend products to a user, we first find the products bought by the given user and also all the products bought by other users from the cluster the given user belongs to and then, subtract both the sets.

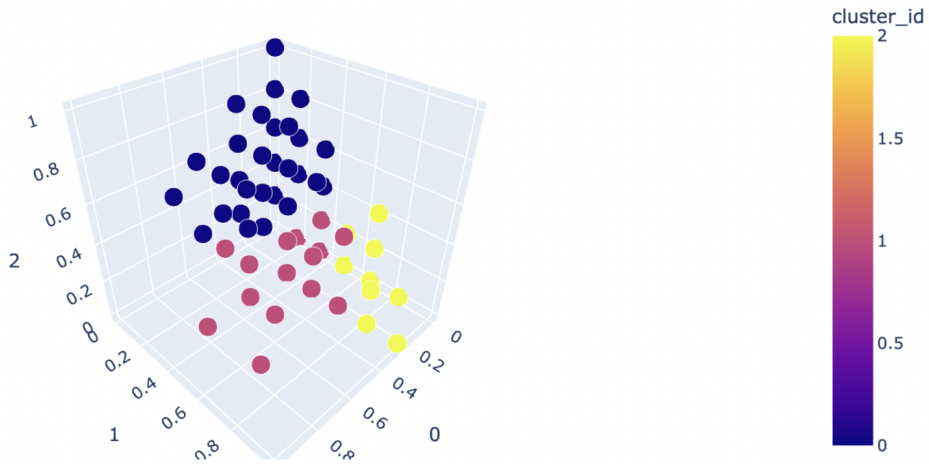


Figure 10: User-Cluster Cluster

2.2 Result and Recommendation

As shown in the picture 10, the users were segmented in three different clusters. The details of the clusters are as shown in the table below:

Cluster ID	Number of Users	Number of Unique Products
0	77	42
1	526	68
2	388	24

Table 2: User-Product Cluster Details

As mentioned above, the recommendation for a user is the rest of products bought by other users from the cluster the give users belongs to. For instance, the recommendation for user **A105A034ZG9EHO** are 64 products. In part 1, we got only 15 products as recommendation for the same product. It seems the clustering in the part 2 might be faulty.

The number of products bought by users is clusters are [42, 68, 24]. For cluster 0 and 2, the total number of products looks reasonable but for cluster 1, the total number of products is 68 which is distinctly high given it is nearly 3/4 of all the products. In average, each user has only bought 4 products which can never lead to multiple users buying 68 different products such that they can be clubbed together based on the products they bought. Therefore, it seems, in part 2, the final clusters contain a lot of dissimilar users. This can be further seen in Figure 10 where the points are evenly spread in all direction as there seems no distinct clusters as such.

Moreover, the final clustering of users is based on the clustering of product description text. 39 of the product which is 46% of total products didn't have any product description. Without any description text, there was no basis for finding similarity/dissimilarity between these product and hence they were clubbed together. This might have caused different dissimilar products to be clustered together, leading to a faulty result in the final user-cluster matrix clustering.

3 Part 3: BERT classifier for predicting ratings

3.1 Approach:

i) I created a new pandas dataframe with reviews and ratings

	review	rating
0	As advertised. Reasonably priced	5
1	Like the oder and the feel when I put it on my...	5
2	I bought this to smell nice after I shave. Wh...	1
3	HEY!! I am an Aqua Velva Man and absolutely lo...	5
4	If you ever want to feel pampered by a shampoo...	5
...	...	...
5264	I have genetic undereye darkness. Ive accepted...	5
5265	I absolutely love this eye gel.	5
5266	The eye gel is easy to apply and I use it morn...	5
5267	Ok this eye gel is good stuff.	5
5268	This is the first eye gel/cream that actually ...	5

5269 rows × 2 columns

Figure 11: User-Cluster Cluster

There are 5264 unique reviews and ratings but out of that rating of 5 has highest number of reviews nearing 89% of the total data available. The dataset is heavily skewed towards the ratings of 5.

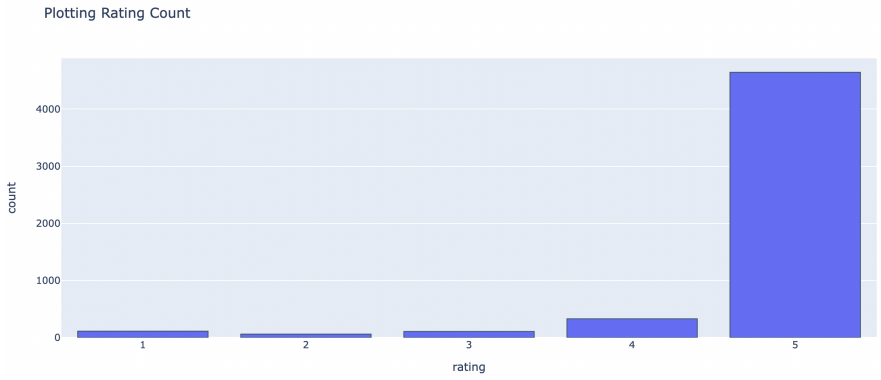


Figure 12: Rating Count

ii) To that end, I created another balanced dataset with equal number of all ratings from 1-5 but this lead to drastic reduction in training data - only 100 reviews for each label, a total of 500 examples for training.

iii) For both the dataset(unbalanced and balanced), reviews were taken as 'x' and ratings were taken as 'y'. The ratings were one-hot encoded for training.

iv) **Training:** I used a small Bert model that has 4 layers, 512 hidden units and 4 attention heads to build the classifier. I used the corresponding text preprocessor for this Bert

model to preprocess the text for inputs to the bert model.
At the output, I had a dropout layer with rate of 0.1 and then followed by a dense layer with 5 output units as there are 5 rating labels from 1-5.
The dataset was split into 70-30 ratio for training and testing data respectively. During training, a validation split of 10% was taken.

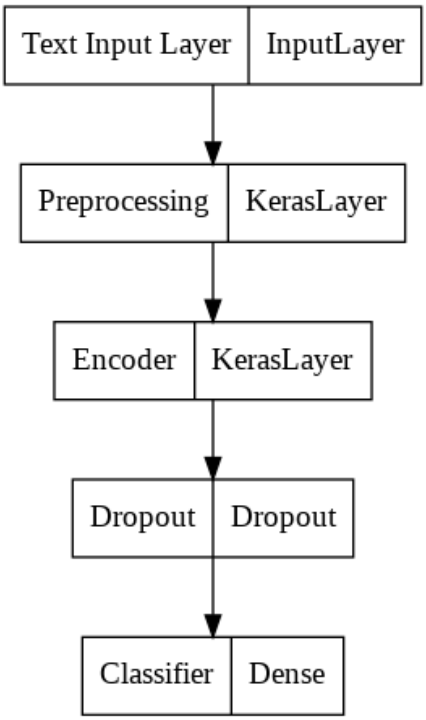
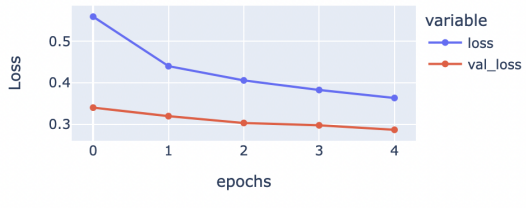


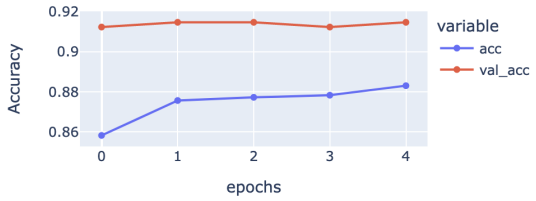
Figure 13: Model Layout

3.2 Results and Major Findings:

i) Training Results:

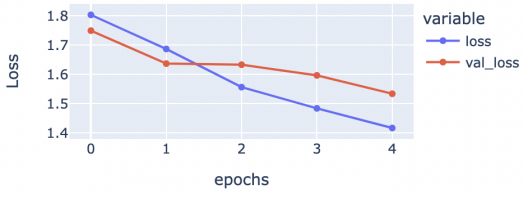


(a) Loss for unbalanced dataset

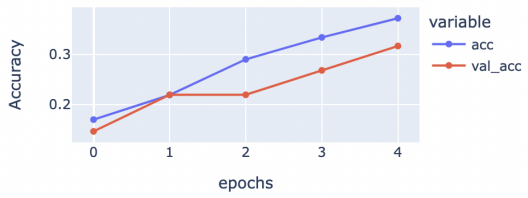


(b) Accuracy for unbalanced dataset

For the *unbalanced* dataset, as shown above, there is a huge disparity between training and validation metrics which implies that the model is overfitting.



(a) Loss for balanced dataset



(b) Accuracy for balanced dataset

For the *balanced* dataset, as shown above, the training and validation metrics are moving in the same direction with no huge disparity between them. This implies the model is learning appropriately.

ii) Testing Results:

On *unbalanced* dataset, the model has a testing accuracy of 89% but as we know this is obviously because of the high number of ratings of 5. This was evident when testing on a random dataset of 5 reviews.

```
sample_dataset = [
 'This is not a nice product. I hate it',
 'Awesome Product. Really Good',
 'The product is damaged. I am really dissappointed with the final product.',
 'Worth Buying. I loved it.',
 'I really liked the product. The product is great'
]

(np.argmax(classifier_model.predict(sample_dataset), axis = 1 ) + 1)
```

```
array([5, 5, 4, 5, 5])
```

Figure 16: Prediction result for unbalanced dataset trained classifier

As shown above, for all the reviews we are getting positive ratings.

On *balanced* dataset, the model has a testing accuracy of 47%. Though the accuracy was low, the model was able to correctly assign ratings to some of the reviews.

```
sample_dataset = [
 'This is not a nice product. I hate it',
 'Awesome Product. Really Good',
 'The product is damaged. I am really dissappointed with the final product.',
 'Worth Buying. I loved it.',
 'I really liked the product. The product is great'
]

np.argmax(classifier_model.predict(sample_dataset), axis = 1) + 1
```

```
array([1, 5, 4, 5, 4])
```

Figure 17: Prediction result for balanced dataset trained classifier

iii) **Explanation:** For the unbalanced dataset, the data was skewed towards a class label and that bias was also learned by the model. Hence, the bert-based rating predictor was of no use.

For the balanced dataset, though the skewness in the data was eliminated, the final dataset had very few examples (around 500) to learn any meaningful information from the dataset. Despite that the model trained on the balanced dataset is a better predictor than the one trained on the unbalanced dataset as clearly shown above.

4 Insights and Suggestions

Insights:

i) Out of the 85 products, only 46 products (54 %) had a description text and among them 30 of them having more than one field like *size, color* had a rating of 5. This implied better description text leads to a better reviews and ratings as users can have a good understanding of the product they are buying.

ii) The dataset had 5264 unique reviews and ratings out of which only 200 of them were bad. Inspect that products with bad reviews one could see that most of them lacked a description text. So, having no description text for products likely means getting a bad review and rating for the product.

iii) Similarly, there wasn't a much option provided with any of the categories. For shampoos and other categories, the products were similar or lacked range. This could have led to dissatisfied customers who ended up giving bad reviews and ratings.

Suggestions: Based on the insights, the suggestions for the beauty product manufacturer are evident.

i) Have detailed description of all the products that are up for the sale. Customers want to know as much as possible about the products.

ii) Don't put the products with no description text at all. This reflects badly on the product.

iii) Have diverse range of options for each category of cosmetic products that means in different sizes, color or with different use.

5 Conclusion

The Beauty dataset that was used to build models for this assignment was more or less a poor one. The dataset was riddled with missing rows and poor entries. Moreover, there was inconsistency in the data with lot of duplicate information. Without any meaningful information in the dataset, it was challenging to build reliable and robust models with it.