Carine Wong
Hajera Laique
Isabella Gonzales
Libby Amir
ECS 111

ECS 111 Final Project: Sephora Product Recommendation System

Introduction:

We have all experienced buying products that turn out to not be the best purchase for us. This occurs often in the makeup industry, where there are numerous products available, but few that work best for our features. A product that is meant for oily skin may not be the best purchase for a person that has dry skin. We decided to create a model that can help users find products to purchase based on their specific features. Our project is a recommendation system which takes in the user attributes and a product of interest and then generates a list of ten product recommendations. Our model will also give a rating for each of these products based on how other people with similar features rated that product. The recommendation system we set out to make would be able to benefit both the customer and the seller (in this case Sephora). Customers may not be aware of all the products available and recommendations help them discover alternatives they might not have found otherwise. Providing recommendations saves customers time and effort and makes it so they can compare features more easily. From the seller perspective, by helping customers find products that meet their needs, the company can build loyalty, encouraging customers to return for future purchases. When the recommendation results provide relevant alternative options, it could increase the likelihood that customers purchase, thus increasing revenue. Overall, this model will provide aid to both the customer and company by providing several recommendations of a desired product based on user attributes.

Dataset Description:

The dataset we used came from a Kaggle dataset called Sephora Products and Skincare Reviews which was collected via a Python web scraper in March of 2023. Our data had two data frames: one for products and one for customer reviews. Here is the metadata of the features we used:

Products Data

- *Product ID:* The unique identifier for the product on the website
- *Brand Name*: Associated brand
- *Rating*: The average rating of the product based on user reviews
- *Highlights*: A list of tags or features that highlight the product's attributes
- *Primary Category*: General type of product (Haircare, Skincare, Makeup, etc)
- *Secondary Category*: Subcategory of product (Moisturizer, Cleanser, Lip Balm, etc)

Reviews Data

- *Author ID*: The unique identifier for the author of the review on the website
- *Product ID*: The unique identifier for the product on the website
- *Rating*: The rating given by the author for the product on a scale of 1 to 5
- *Review Text*: The main text of the review written by the author
- *Skin Tone*: Author's skin tone (e.g. fair, tan, etc.)
- *Eye Color*: Author's eye color (e.g. brown, green, etc.)
- *Skin Type*: Author's skin type (e.g. combination, oily, etc.)

Exploratory Data Analysis:

Our data includes information about over 8,000 beauty products from the Sephora online store as well as about a million user reviews for over 2,000 of the products. Before we began building a model, we wanted to conduct exploratory data analysis to explore the product categories and the review data. Because we knew we wanted to do a model related to recommendations based on the product review data, we made a bar plot to observe the ratings (figure 1). We saw that the ratings are skewed toward 5 stars which we will take into account when interpreting the model.

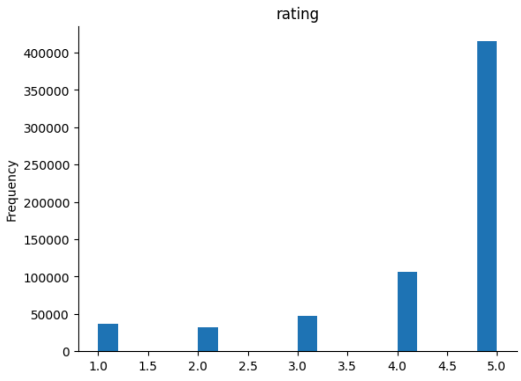

Figure 1. Rating Histogram

This was also reconfirmed by a word cloud we made of the review text (figure 2) which shows that words like “love” had high frequencies. However, the large amount of positive ratings and reviews could also be because users left reviews only on products they liked rather than the ones they disliked. It is important to keep this information in mind when we are making conclusions of our results.

Figure 2. Word Map

Because our model will also be taking into account user characteristics, we made a heatmap to compare the skin types and the skin tones (figure 3). We found that the most common combination is light/combination skin and the least common skin tones are olive, porcelain, and ebony.

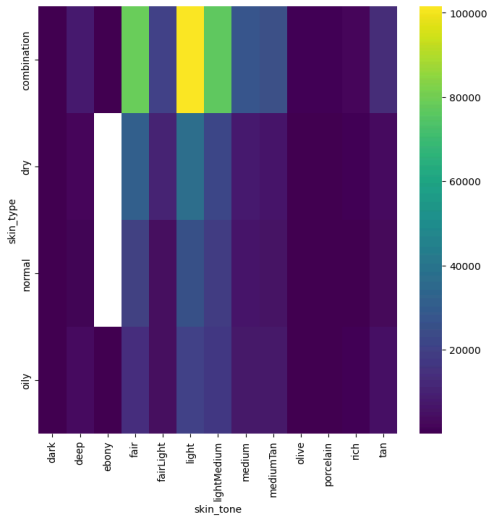

Figure 3. Skin tone/Skin type heat map

Data preprocessing:

To preprocess our data, we did quite a bit of filtering. We removed NA values to avoid unnecessary errors and promote consistency. Luckily, the dataset we chose was very large, so we

could remove this excess “noise” and still have plenty of data to run with. This also meant that our model runtime started out pretty lengthy: we filtered reviews by removing those written by authors who have written less than 10 reviews, under the assumption that they are less credible. Sephora also sells various products like fragrances, and we chose to focus only on makeup and skincare for our recommendation system. Through exploratory data analysis, we noticed that many products had different shades, mini versions, and travel sizes. To reduce redundancy of recommendations, we thought to filter out these duplicate products. However, after further observation, we noticed that filtering by secondary product type eliminated this concern.

We used TF-IDF to process the review text, which required some preprocessing. We completed this through tokenization, removing stop words/non-words/emojis/numbers, lowercasing, removing punctuation, and stemming. We also used TF-IDF on the highlights text, but the concise formatting of that column did not need to be changed. Finally, we wanted to create a regression model using user features, so we one-hot encoded the eye color, skin type, and skin tone features.

Building the Algorithm:

Starting out, we were unsure what the best approach would be for summarizing the information in our data set. Initially, we knew we wanted to convey how certain products work better for certain people, but there was a lot of uncertainty in how we could manipulate the data to show that. We initially thought of a system that would take in a person’s features, and then output products that are recommended for those features. Then we realized that this would be a little difficult to achieve, and we wanted to start with a more feasible approach. That is when we decided on our starting point: a model that inputs a product name and returns 10 products that are recommended based on that product.

Our first implementation was relatively simple. After cleaning our data, we obtained a TF-IDF matrix (a natural language processing method that numerically captures the importance of words in a text) of the highlights text associated with each product. This attribute of the product data was a good initial variable because it summarizes some of the most important characteristics of products without extraneous filler words, unlike the written reviews. After we had the highlights TF-IDF matrix, we used cosine similarity (a measure of distance) as the metric in our main function. The get_recommendations function takes a product name as input and then

returns the 10 products with the highest cosine similarity scores relative to that inputted product. While this simple model works in theory, it does not consider enough information to make accurate, personalized recommendations. After this initial step, we immediately thought of several ways to improve the algorithm, and that led us to our next iterations.

```
# when calling for recommendations, acknowledge that spelling and grammar are crucial
get_recommendations("Lip Sleeping Mask Intense Hydration with Vitamin C", cosine_similarity)
```

```
297          Clinique iD Custom-Blend Hydrator Collection
1230                  Hydro Grip Hydrating Makeup Primer
2008             Hyaluronic Acid 2% + B5 Hydrating Serum
2206           Mini Superberry Hydrate + Glow Dream Mask
779                      Vanish Flash Highlighting Stick
840              Mini Limitless Lash Lengthening Mascara
1014                     Tinted Face Oil Comfy Skin Tint
1244               Sunshine Vitamin C + Squalane Face Oil
2205    Superberry Hydrate + Glow Dream Night Mask wit...
1716       Synchro Skin Self-Refreshing Foundation SPF 30
Name: product_name, dtype: object
```

Figure 4. Iteration 1 example

The second version of the algorithm was pretty similar to our first, only with a minor change to the code that gave us much better results. After running our first model, we noticed the products recommended were not all obviously related. Inputting a lip balm would return results of face creams and makeup, and that was not what we had in mind. To improve this, we used the already existing categories of the products to filter out unrelated products. The data set has several columns of categories, but we went with the secondary categories because they are relatively specific but not overly niche. Our second version of the function, get_recommendations2, only considers the products in the same secondary category as the input, and therefore returns 10 relevant product recommendations.

```
# when calling for recommendations2, we can compare the results with recommendations
get_recommendations2("Lip Sleeping Mask Intense Hydration with Vitamin C", cosine_similarity
```

```
1101                                    Lip Glowy Balm
1956                                 The Kissu Lip Mask
160                         Squalane+ Rose Vegan Lip Balm
616              Sugar Recovery Lip Mask Advanced Therapy
617              Sugar Mint Rush Freshening Lip Treatment
1394                  Pout Preserve Peptide Lip Treatment
1105                                 Lip Treatment Balm
607                    Sugar Lip Balm Hydrating Treatment
596     Sugar Advanced Lip Balm Intense Hydration Trea...
1667                              Clean Lip Balm & Scrub
Name: product_name, dtype: object
```

Figure 5. Iteration 2 Example

Our third iteration was also focused on the practicality of our algorithm. While the product names can be lengthy and descriptive, they do not include the brand of the product which can lead to confusion. Especially now that the recommendations are more related to each

other. We found that a result of 10 lip gloss names which are all similar is not very useful if you don't know the brands of the products. Therefore, we implemented a third version, get_recommendations3, that prints the names of the products as well as their brand names. This small change makes our output much more interpretable, and also gives more insight into product versus brand similarities.

```
# when calling for recommendations3, we can now see the brand of every product
get_recommendations3("Lip Sleeping Mask Intense Hydration with Vitamin C", "LANEIGE", cosine_similarity)
```

```
Recommendations for Lip Sleeping Mask Intense Hydration with Vitamin C:
```

	product_name	brand_name
1101	Lip Glowy Balm	LANEIGE
1956	The Kissu Lip Mask	Tatcha
160	Squalane+ Rose Vegan Lip Balm	Biossance
616	Sugar Recovery Lip Mask Advanced Therapy	fresh
617	Sugar Mint Rush Freshening Lip Treatment	fresh
1394	Pout Preserve Peptide Lip Treatment	OLEHENRIKSEN
1105	Lip Treatment Balm	LANEIGE
607	Sugar Lip Balm Hydrating Treatment	fresh
596	Sugar Advanced Lip Balm Intense Hydration Trea...	fresh
1667	Clean Lip Balm & Scrub	SEPHORA COLLECTION

Figure 6. Iteration 3 Example

After making some small edits to our already existing model, we wanted to really improve the quality of our recommendations. Our fourth implementation not only includes the highlights from before, but performs a similar operation using the product reviews. This would likely be an improvement from the previous iteration as it factors in the description of the product by the brand and also the perception/description by the customer. The review text is much messier, and thus required a lot more data cleaning and preprocessing before creating a TF-IDF matrix. We were also working with a lot more data now, and we wanted to improve the efficiency of our algorithm by filtering out reviews that we deemed less useful. For example, with the amount of reviews data we have, we were able to remove all reviews by authors who have written less than 10 product reviews and still have more than enough remaining data. We did this with the assumption that people who have reviewed more products are more likely to write informational and accurate reviews. Using this subset of reviews theoretically translates to more accurate recommendations, and we confirmed this later on by getting MSE values for the regression model in iteration 5. The subset resulted in a lower MSE versus using the whole data set, so this decision did in fact lead to higher accuracy. Using a similar approach to the highlights

text, we combined the highlights and review texts into one column and used TF-IDF and cosine similarity to get similar products to the input. Combining this with the existing highlights method, we got some new and improved recommendations, again with both product names and brands printed. The difference in products shows that the reviews text is being taken into account and the reviews still make logical sense.

```
# when calling for recommendations4, we get recommendations based on highlights and their reviews
get_recommendations4("Lip Sleeping Mask Intense Hydration with Vitamin C", "LANEIGE")
```

```
Recommendations for Lip Sleeping Mask Intense Hydration with Vitamin C:
```

	product_name	brand_name
486	Lip Glowy Balm	LANEIGE
489	Lip Treatment Balm	LANEIGE
814	The Kissu Lip Mask	Tatcha
295	Sugar Recovery Lip Mask Advanced Therapy	fresh
43	Squalane+ Rose Vegan Lip Balm	Biossance
203	Lippe Balm	Drunk Elephant
719	Brazilian Kiss Cupuaçu Lip Butter	Sol de Janeiro
666	Clean Lip Balm & Scrub	SEPHORA COLLECTION
234	Honey Butter Beeswax Lip Balm	Farmacy
18	Willow & Sweet Agave Plumping Lip Mask	alpyn beauty

Figure 7. Iteration 4 Example

Our fifth iteration added brand new information to our recommendation system: a predicted rating. We wanted a numerical metric to be displayed and evaluated, as well as integrate an element of personalization. For these reasons, we created a linear regression model that takes into account the attributes in choosing specific products that are better for each individual. To execute this, we made a separate predict_rating function that takes attribute inputs of skin tone, eye_color, and skin_type, as well as a product name. It calculates a predicted rating for that product for an individual with those characteristics based on ratings by individuals with similar attributes. In practical usage, this personalized metric gives the viewer much more information than just the 10 recommended products. Our inspiration for this feature came from our personal experiences and background knowledge about customer segmentation in the cosmetic industry. Attributes like skin tone can entirely change the satisfaction of a customer with a makeup product, or someone with dry skin might love a moisturizer that a person with oily skin finds too rich. The predicted ratings can help give consumers insight into the specific products that might work better for their needs.

```
get_recommendations5('tan', 'brown', 'dry', "Lip Sleeping Mask Intense Hydration with Vitamin C", "LANEIGE")
```

```
Recommendations for Lip Sleeping Mask Intense Hydration with Vitamin C:
```

	product_name	brand_name	predicted_rating
486	Lip Glowy Balm	LANEIGE	4.347574
489	Lip Treatment Balm	LANEIGE	4.148530
814	The Kissu Lip Mask	Tatcha	3.741183
295	Sugar Recovery Lip Mask Advanced Therapy	fresh	4.691435
43	Squalane+ Rose Vegan Lip Balm	Biossance	3.829175
203	Lippe Balm	Drunk Elephant	3.716643
719	Brazilian Kiss Cupuaçu Lip Butter	Sol de Janeiro	4.096331
666	Clean Lip Balm & Scrub	SEPHORA COLLECTION	3.216920
234	Honey Butter Beeswax Lip Balm	Farmacy	4.218792
18	Willow & Sweet Agave Plumping Lip Mask	alpyn beauty	4.516091

Figure 8. Iteration 5 Example

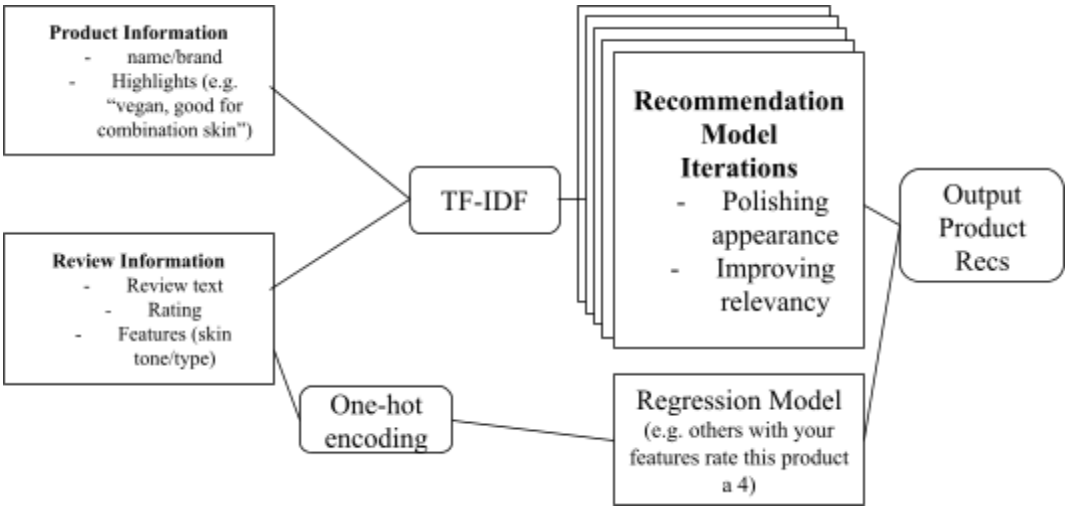

Figure 9. Model Visualization

If we were to build on this algorithm, we could add even more personalization. If the product data included attributes like cost, we could implement another optimization factor to recommend products within a customer's desired budget. We are also aware that while TF-IDF and cosine similarity are effective for identifying textual similarities, they may overlook nuances and context, potentially limiting the accuracy of conclusions drawn from complex text analyses. A further step could be to use BERT score to evaluate similarity instead as it is a pre-trained model that can better understand the context and take into account word order. Lastly, our final iteration outputs the list of recommendations from most similar to the original product to least similar, but in a future iteration we could take the predicted rating into account and optimize the similarity and predicted rating.

Evaluating the results:

It is difficult to evaluate recommendation model accuracy objectively, so our approach was to make as many improvements/iterations as possible. We can, however, measure the accuracy of our regression model created in Iteration 5. Our model takes in the user's features as inputs, one-hot encodes those inputs, and outputs a floating point number as a review estimate.

To evaluate our predict_rating() function, we trained our model on 80% of our review data, and tested on the remaining 20%. The MSE of our model is 0.929: while this isn't as low as we might've liked, we recognize that human ratings have biases and think that this value is relatively good. We manually tested the regression model on a few products using different user features to see if the results were as expected. One product was an oil based serum with an average rating of 4.02 and when our model was tested for someone with dry skin the predicted rating was 4.5, indicating that it is more suitable for dry skin types. Another product was a sunscreen known to leave a slight white cast on darker skin and when the predicted rating for someone with fair skin was 3.92 and for dark skin it was 3.87. These examples have small differences between them but this could be because of the skewed distribution of the ratings.

To evaluate our recommendation model accuracy, we performed "sanity checks." As a demonstrative example, we compared our recommendations for the Laneige Lip Sleeping mask to the Sephora website. Many of the suggestions are the same as what shows up in our model. Some of the potential explanations for differences are our limited dataset and these suggestions could be powered by AI already and browsing history. For example, our data set cannot account for new product releases instantly and limited edition items–and the first suggestion in our product query was a product that falls into both of these categories.

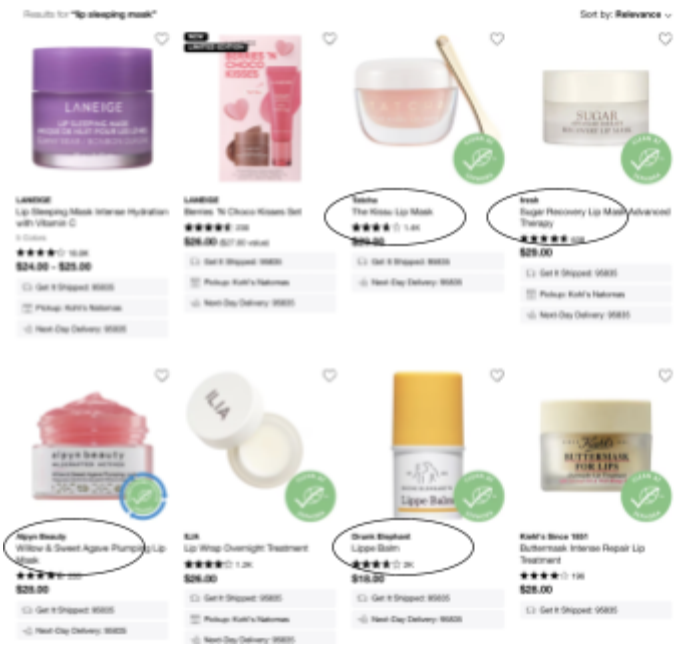

Figure 10. Sephora Website Example

To further check our model we tested it on several other products. For the retinol product below, retinol was mentioned in all recommended products. This fact indicates that our 'highlights' column holds meaningful information about the product contents, such as active ingredients like retinol. This indicates that the TF-IDF matrix places retinol as a highly important word in our data. It also demonstrates some accuracy in the model as all products seem logically related.

Recommendations for Retinol Anti-Aging Serum:

	product_name	brand_name	predicted_rating
515	Retinol Youth Renewal Serum	Murad	4.667177
205	A-Passioni Retinol Cream	Drunk Elephant	4.092556
584	CLINICAL 1% Retinol Treatment	Paula's Choice	4.674557
677	Retinol Reform Treatment Serum	Shani Darden Skin Care	4.288439
242	1% Vitamin A Retinol Serum	Farmacy	4.737337
271	FAB Skin Lab Retinol Serum 0.25% Pure Concentrate	First Aid Beauty	3.681529
598	Retinol Face Stick	Peace Out	4.570142
431	Micro-Dose Anti-Aging Retinol Serum with Ceram...	Kiehl's Since 1851	4.451425
763	A+ High-Dose Retinol Serum	Sunday Riley	3.970229
406	Argan Beta Retinol Pink Algae Serum	Josie Maran	4.307496

Figure 11. Additional product example

For a second example, we observed our model's recommendations of a gentle hydrating cleanser. The word "Gentle" was mentioned in many suggestions and "Hydrating" is acknowledged by hydrating ingredients mentioned like fulvic acid, squalane, and aloe. This demonstrates that important features of products are being accounted for by our model. Since one of our inputs is

“dry” where hydration is important, predicted ratings are all pretty high, indicating some level of accuracy of our predicted ratings. The last recommendation is a scrub, which is a different specific type of cleanser, demonstrating that our recommendation model does produce less and less relevant recommendations as they are listed out in order. Though there are potential improvements we could make to the model the results seem logical and recommendations fit what we would generally expect.

```
Recommendations for Soy Hydrating Gentle Face Cleanser:
```

	product_name	brand_name	predicted_rating
394	Confidence in a Cleanser Hydrating Facial Clea...	IT Cosmetics	4.223640
40	Squalane + Amino Aloe Gentle Pore-Minimizing C...	Biossance	4.146985
908	Superfood Antioxidant Cleanser	Youth To The People	4.165284
846	Fulvic Acid Brightening Cleanser	The INKEY List	4.728240
847	Mini Fulvic Acid Brightening Cleanser	The INKEY List	4.824334
315	Blueberry Bounce Gentle Cleanser	Glow Recipe	3.887933
377	Keep It Clean Hydrating Gel Cleanser with Cera...	iNNBEAUTY PROJECT	4.568328
77	Vinoclean Gentle Foam Cleanser	Caudalie	4.333074
803	The Rice Wash Skin-Softening Cleanser	Tatcha	4.344725
903	Yo Glow AHA & BHA Facial Enzyme Scrub	Wishful	4.042546

Figure 12. Additional Product Example

Conclusion and Discussion:

In conclusion, our exploration into recommendation systems revealed their complexity and the inherent challenges in evaluating them objectively. Despite the lack of a concrete accuracy metric, our spot checks indicated that our model produces reasonable outcomes. Our model could be used to enhance personalized recommendations and add to the traditional methods like those of Sephora, which focus on popularity, price, and category. Although our model does not currently prioritize results based on these personalized predictions, it effectively incorporates personal attributes to generate relevant prediction ratings. Finally, our algorithm can be used for customer segmented marketing. When brands release new products, they can use this to anticipate which segment of the market is most likely to get satisfaction from their product and advertise accordingly.

Contributions:

Carine: Data preprocessing, iteration 2, iteration 4

Hajera: EDA, joining iteration 5 to iteration 4, results testing
Isabella: EDA, regression model for iteration 5, results testing
Libby: Data preprocessing, iteration 1, iteration 3

We all contributed to the report, check-ins, and discussed methodology together. We also had joint coding sessions for a couple of the iterations where we all provided input and worked together to build the model. We all contributed to creating the slides for the presentation.

References:

https://www.kaggle.com/datasets/nadyinky/sephora-products-and-skincare-reviews