“SKINCARE PRODUCT RECOMMENDATION SYSTEM”

Capstone Project Final Report

Nisha Siddarama Gowda

Supervised by:

Dr. Humera Noor Minhas

Prof. Ebby George Varghese

Winter 2022-2022

Table of Contents

List of Figures

ABSTRACT

The demand for cosmetics has grown recently, especially in the area of skincare, around the globe. Consumers previously relied on top-selling items or in-store suggestions. But because everyone has a distinct type of skin, these are ineffective ways for determining if a product will work for a particular consumer. In this paper we propose two approaches to recommend a product, first collaborative filtering method by which a company can improve its online sales by making more accurate product recommendations for its customers. Second is a content-based recommendation system useful for customers to find skincare products that match their ingredient preferences and for skincare companies to understand which ingredients are more preferred by customers.

1. INTRODUCTION

Skincare is the fastest-growing market category in the beauty industry. In the US last year, sales of skincare products increased by 13% while those of makeup only increased by 1%. In that time, the growth of online beauty merchants was 24%, with skincare taking the lead. Net-a- The best-selling category in Porter's beauty section is skincare, which increased 40% from one year to the next. According to L'Oréal, skincare accounts for 40% of the beauty market but for roughly 60% of the growth in the global cosmetics sector. Consumers' pursuit of natural beauty and men's growing interest in skincare products are the key drivers of this growth.

As the skincare market is growing, new brands and products are emerging, making it a task for customers to choose what product to buy and that's where a product recommendation system comes to the rescue. Earlier people used to visit the counters to get product recommendations which were time-consuming, later with online websites they simply bought the highest-rated products, which may or may not work for them [3]. A product recommendation system will help consumers choose the right product for their concerns without wasting much time.

Different types of recommendation systems have been proposed by many researchers to resolve the information overloading problem and facilitate the selection process [1] in which the two commonly used methods are content-based filtering and collaborative filtering recently a hybrid method has been introduced which is a combination of these two techniques which increases the benefits while covering the flaws.

Which method is most effective in determining if a product is suitable for a given buyer is still up for debate. Even though each consumer has different skin conditions, many online cosmetics retailers still suggest bestsellers to their clients [2]. Therefore, there is a need for additional research and advancement in the recommender systems for personal care items.

This proposal presents two approaches of recommendation systems where one is beneficial for e-commerce companies whereas the other one is customer friendly. The first one uses collaborative filtering based on a correlation-based approach to make product recommendations. This approach uses a correlation matrix to identify products that are highly correlated with the products that customers have already bought. By recommending these highly correlated products, the company can increase the likelihood that customers will be interested in the recommended products and make a purchase. This model also uses Truncated SVD to decompose a large matrix of user-item interactions (such as ratings) into a smaller number of latent features, which can then be used to make recommendations.

The second approach is a content-based recommendation system where recommendations are made using the ingredients of the products. This approach uses TfidfVectorizer to convert a collection of ingredients into a matrix of TF-IDF features and the KMeans algorithm to cluster the products based on their TF-IDF features and then recommend products to a user based on an ingredient they have shown interest in.

2. DATA

The data was extracted from sephora.com, a website that sells beauty products from multiple brands. Among many categories of personal care items, we have considered fifteen of them which are Moisturizers, Face Serums, Face Wash & Cleansers, Face Masks, Eye Creams & Treatments, Toners, Face Oils, Face Sunscreen, Sheet Masks, Facial Peels, Skincare, Exfoliators, Face Sets, Anti-Aging, and For Face. The dataset consists of 1965 items which include information about id, product, brand, price, rating, details, and ingredients of each product.

3. PROJECT ARCHITECTUTRE

As illustrated in Figure 1, the proposed system offers content-based filtering based on the product name and content-based recommendation is made when a content(ingredient) of product is entered.

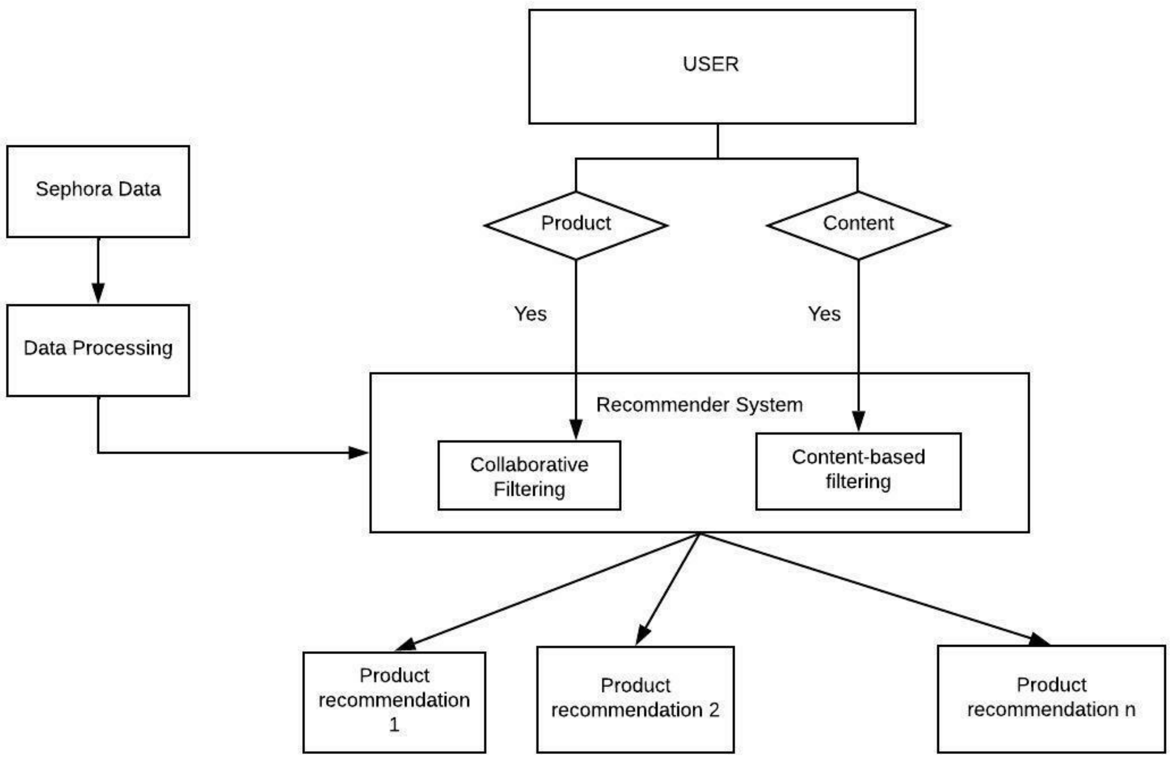

Figure 1: Framework for collaborative filtering and content-based recommendation system

4. METHODS AND IMPLEMENTATION

4.1. Matrix Factorization Technique using Truncated SVD (Singular Value Decomposition)

For Collaborative filtering, we use Truncated SVD. Truncated SVD is a type of matrix factorization technique that can be used to reduce the number of features in a dataset while preserving as much information as possible. It does this by decomposing the original matrix into three matrices: U, S, and V. Here the code is reducing the number of features in the data set X by performing dimensionality reduction using TruncatedSVD and preserving as much information as possible by keeping 10 components from the original matrix.

```
from sklearn.decomposition import TruncatedSVD

SVD = TruncatedSVD(n_components=10)
decomposed_matrix = SVD.fit_transform(X)
decomposed_matrix.shape
```

Figure 2: Matrix Factorization

The next step is to calculate the correlation coefficients between the columns of the decomposed matrix, and this can be used to identify the correlation between the different features in the decomposed matrix.

```
correlation_matrix = np.corrcoef(decomposed_matrix)
correlation_matrix.shape
```

Figure 3: Correlation Matrix

Then we identify a specific product in the database by its name, first by finding the index of the product name in a list of product names and then by accessing the specific row in the Data Frame that has that index label.

```
X.index[220]
```

```
'BioLumin-C Vitamin C Serum'
```

```
i = "Acid Mantle Repair Moisturizer With 250mg CBD and Ceramides"

product_names = list(X.index)
product_ID = product_names.index(i)
product_ID
```

```
64
```

Figure 4: Identifying specific product

The next step is to extract an array of correlation coefficients representing the correlation of the product ID with all the other products.

```
correlation_product_ID = correlation_matrix[product_ID]
correlation_product_ID.shape
```

Figure 5: Extracting correlation coefficients

Last step is to create a list of product IDs that have a correlation coefficient greater than 0.90 with the product ID i. This means that products in this list are highly correlated with product ID i, and therefore may be good recommendations for a user who has bought product ID I and then removes item i from the list of recommendations, since the user has already bought this item finally, slicing the list to get the first 9 items, which are the first 9 recommended items.

```
# Recommending top 10 highly correlated products in sequence

Recommend = list(X.index[correlation_product_ID > 0.90])

# Removes the item already bought by the customer
Recommend.remove(i)

Recommend[0:10]
```

Figure 6: Recommending products

4.2. Feature Extraction using TfidfVectorizer

For the content-based approach, first, we create an instance of the TfidfVectorizer class, which is used to convert a collection of raw documents into a matrix of TF-IDF features. The stop words parameter is set to 'english', which tells the vectorizer to ignore commonly used English words such as "the" and "and" that do not contain much meaningful information. The vectorizer is then fit to and transformed on the "details" column of the "ingredients1" data frame, resulting in a sparse matrix representation of the TF-IDF features, which is stored in the variable X1.

```
vectorizer = TfidfVectorizer(stop_words='english')
X1 = vectorizer.fit_transform(product_descriptions1["ingredients"])
X1
```

Figure 7: Feature extraction

4.3. K-means clustering

We have used k-means algorithm to cluster the sparse matrix of TF-IDF features, and then plotting the cluster assignments of the data points in a scatter plot.

```
# Fitting K-Means to the dataset
import matplotlib.pyplot as plt
X=X1

kmeans = KMeans(n_clusters = 20, init = 'k-means++')
y_kmeans = kmeans.fit_predict(X)
plt.plot(y_kmeans, ".")
plt.show()
```

Figure 8: Using k-means to cluster sparse matrix

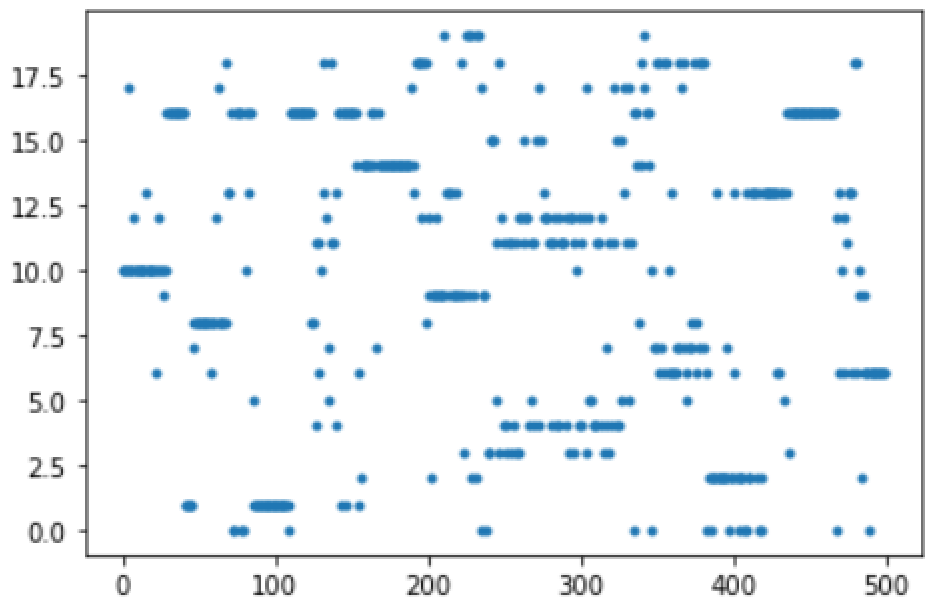

Figure 9: Scatter plot

Then we print the top 10 words for a given cluster number.

```
def print_cluster(i):
    print("Cluster %d:" % i),
    for ind in order_centroids[i, :10]:
        print(' %s' % terms[ind]),
    print
```

Figure 10: printing top 10 words in a cluster

Next step, we print the top 10 words for all clusters obtained by running k-means on the sparse matrix of TF-IDF features.

```
##Optimal clusters is

true_k = 20

model = KMeans(n_clusters=true_k, init='k-means++', max_iter=100, n_init=1)
model.fit(X1)

print("Top terms per cluster:")
order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names()
for i in range(true_k):
    print_cluster(i)
```

```
Top terms per cluster:
Cluster 0:
 skin
 vitamin
 spots
 free
 helps
 dark
 lines
 serum
 aging
 tone
```

Figure 11: Optimal cluster

The show_recommendations function uses the predict method on the previously trained KMeans model to predict the cluster that the input product belongs to and stores the result in the variable "prediction". The function then calls the previously defined "print_cluster" function and passes the first element of the "prediction" variable as the input argument. This will print the top 10 words of the cluster to which the input product belongs to. So this code is used to show the recommendations for a given product by identifying the cluster it belongs to and then printing the top 10 words of that cluster.

```
def show_recommendations(product):
    #print("Cluster ID:")
    Y = vectorizer.transform([product])
    prediction = model.predict(Y)
    #print(prediction)
    print_cluster(prediction[0])
```

```
show_recommendations("aloevera")
```

Figure 12: Show recommendations function

5. RESULTS

For the collaborative filtering approach, the recommendation system works as expected and it prints top 10 highly correlated products.

```
['Adaptogen Deep Moisture Cream with Ashwagandha + Reishi',
 'Beauty Sleep Power Peel',
 'Correct+™ Dark Spot Corrector',
 'Daily Vitamin C',
 'Face Mask - Lychee - Moisturizing',
 'Moisture Surge Hydrating Supercharged Concentrate Mini',
 'Nose Mask - Pineapple',
 'Perfectly Clean Multi-Action Foam Cleanser/Purifying Mask ',
 'Resveratrol Lift Face Lifting Soft Cream',
 'Rosehip Cleanser']
```

Figure 13: Collaborative filtering result

For the content-based filtering approach, the recommendation system prints top 10 words of the cluster to which the input(ingredient) product belongs to. When a cluster is found using the user's search terms, the recommendation system can show products from that cluster depending on the product's ingredients.

```
show_recommendations("niacinamide")
```

```
Cluster 15:
 oil
 skin
 extract
 vitamin
 appearance
 acid
 helps
 natural
 sodium
 wrinkles
```

Figure 14: Content-based recommendation result

6. CONCLUSION

This project implemented collaborative filtering and content-based approaches to make product recommendations based on user profiles and inputs. By implementing the first recommendation system, the company can improve its online sales by making more accurate product recommendations for its customers. This can lead to increased customer satisfaction and loyalty, which can result in increased sales and revenue for the company. Additionally, this can help the company to stand out in a crowded e-commerce market, by providing a better customer experience.

Whereas the content-based recommendation system helps to surface products that the customer is more likely to be interested in, which can save them time and effort in finding items that match their preferences. Additionally, by recommending related items, a content-based recommendation system can help to uncover products that the customer may not have been aware of, but that they may be interested in.

7. FUTURE WORK

Further research is required to achieve more accurate results because the data is only from a limited sample. The technique could be enhanced in the future by taking brand preferences or pricing into account when providing recommendations. The hybrid recommender system might also be attempted to be implemented with the right data set.

8. REFERENCES

[1] Ndengabaganizi Tonny James and K. Rajkumar. 2017. Product Recommendation Systems based on Hybrid Approach Technology. International Research Journal of Engineering and Technology 4, 8 (Aug. 2017). www.irjet.net

[2] X Ren. 2016. SKII Recommender System Design. http://xren615.github.io/post/ sk2_rs/

[3] Gyeongeun Lee, A Content-based Skincare Product Recommendation System. Association for Computing Machinery