Final Report of Internship Program 2023

On

“Analysis of Chemical Components of Cosmetics”

MEDTOUREASY

SAJNA P

28th February 2023

ACKNOWLDEGMENTS

The internship opportunity that I had with MedTourEasy was a great chance for learning and understanding the intricacies of the subject of Data Analytics in Data Engineering and, for personal as well as professional development. I am very obliged to have a chance to interact with so many professionals who guided me throughout the internship project and made it a great learning curve for me.

Firstly, I express my deepest gratitude and special thanks to the Training & Development Team of MedTourEasy who gave me an opportunity to carry out my internship at their esteemed organization. Also, I express my thanks to the team for making me understand the details of the Data Analytics profile and training me in the same so that I can carry out the project properly and with maximum client satisfaction and for spearing his valuable time despite his busy schedule.

I would also like to thank the team of MedTourEasy and my colleagues who made the working environment productive and very conducive.

TABLE OF CONTENTS

ABSTRACT

In recent years, the interest of consumers in cosmetics has been increasing. Globally with a focus on skin care. In the past, consumers have Relies on best-seller products or in-store recommendations from the counter. However, everyone's skin condition is different, therefore these are not effective ways of judging the compatibility between a product and user.

Buying new cosmetic products is difficult. It can even be scary for those who have sensitive skin and are prone to skin trouble. The information needed to alleviate this problem is on the back of each product, but it's tough to interpret those ingredient lists unless you have a background in chemistry. Instead of buying and hoping for the best, we can use data science to help us predict which products may be a good fit for us. This proposal focuses on designing a skincare product recommendation system based on the user's skin type and ingredient composition of a product.

1. NTRODUCTION

1.1 About the Company

MedTourEasy, a global healthcare company, provides you with the informational resources needed to evaluate your global options. It helps you find the right healthcare solution based on specific health needs, affordable care while meeting the quality standards that you expect to have in healthcare.

MedTourEasy improves access to healthcare for people everywhere. It is an easy-to-use platform and service that helps patients to get medical second opinions and to schedule affordable, high-quality medical treatment abroad.

1.2 About the Project

Finding the right cosmetics is very difficult, so that here we can use data science to help us predict which products may be a good fit for us. The abundance of product information and reviews are perceived to be valuable. But at the same time, it prevents users from picking out desired information and making decisions based on their needs. Such difficulty has evoked the pressing need for personalized systems that could ease the access of data.

In this Project, we are going to create a ingredient-based recommendation engine where the 'ingredient' will be the chemical components of cosmetics. Specifically, you will process ingredient lists for 1472 cosmetics on Sephora via word embedding, then visualize ingredient similarity using a machine learning method called t-SNE and an interactive visualization library called Bokeh.

This method also allows users to input their desired beauty effect instead of a product name if they lack knowledge or have not found a product they like.

Additionally, MedTourEasy, being one of the globally foraying tele-medicine company in global healthcare, it is important for the firm to understand people's attitude towards cosmetics their skin care routine, and cosmetics is a fastest-growing category globally. Furthermore, based on the results of the analysis, it can be used to enhance their market presence and capacity planning.

1.3 Objectives and Deliverables

We proposed an ingredient-based recommendation engine that helps us to choose the content, where the 'ingredient' will be the chemical components of cosmetics. Specifically, we will process ingredient lists for 1472 cosmetics on Sephora via word embedding, then visualize ingredient similarity using t-SNE and Bokeh.

Our data set contains, product names, Ingredients, skin preference and price. This data set will be used specifically to evaluate the efficiency of this method after the implementation of the ingredient-based recommendation engine.

2. METHODOLOGY

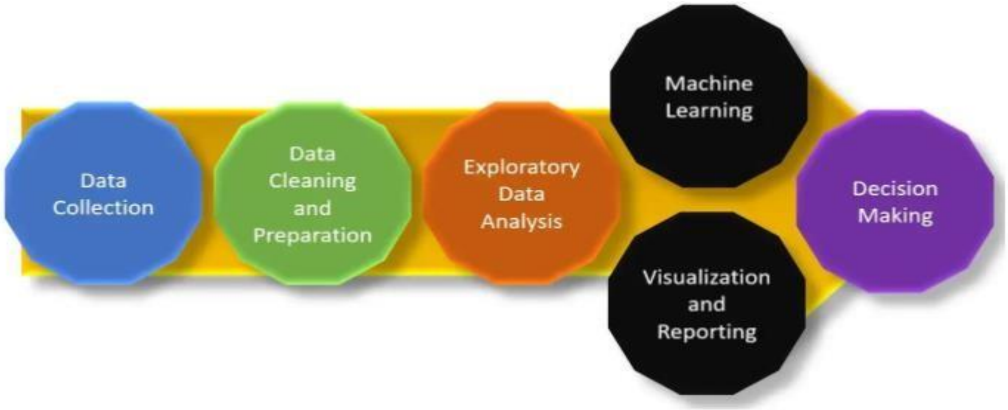
2.1 Importing Dataset

The data was scraped from sephora.com, a website that offers beauty products from multiple brands. Among many categories of personal care items, only six were extracted to focus on skincare products. These six categories include moisturizing cream, facial treatments, cleanser, facial mask, eye treatment, and sun protection. The data set consists of 1472 items which includes information about the brand, name, price, rank, skin types, and chemical components of each product.

Additionally, star ratings for all 1472 items will be extracted from sephora.com along with the reviewers' skin types. The extraction will be done using a tool called Scrapestorm1 that allows data mining from different websites. This data set will be used specifically to evaluate the efficiency of this method after the implementation of the ingredient-based recommendation engine.

2.2 Exploratory Data Analysis

Exploratory Data Analysis(EDA) refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. EDA is an approach to analyze data using visual techniques to obtain desired output.


Data Collection
Data Cleaning and Preparation
Exploratory Data Analysis
Machine Learning
Visualization and Reporting
Decision Making

Exploratory Data Analysis (EDA) is the primary building block of any data-centric project. The above figure shows the process flow from data collection to decision making.

Generally, EDA falls into two categories:-

- **The univariate analysis** involves analyzing one feature, such as summarizing and finding the feature patterns.
- **The multivariate analysis** technique shows the relationship between two or more features using cross-tabulation or statistics.

2.3 Language and Platform Used

2.3.1 Language: Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. It was created by Guido van Rossum and released in 1991.It is used for web development (server-side),software development, mathematics, system scripting. The most recent major version of Python is Python 3.

Python is the most widely used programming language today. When it comes to solving data science tasks and challenges, Python never ceases to surprise its users. Most data scientists are already leveraging the power of Python programming every day. Python has been built with extraordinary Python libraries for data science that are used by programmers every day in solving problems.

Why Python is so popular with developers?

- Python can be used on a server to create web applications. It works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.).
- Versatility, Efficiency, Reliability, Speed, and easy to learn and use.
- Big data, machine learning and cloud computing. and perform complex mathematics.

- Python can connect to database systems. It can also read and modify files.
- Python can be used for rapid prototyping, or for production-ready software development.

Why is Python opted for data analysis?

- Hundreds of Python libraries and frameworks which are valuable for analytics and complex calculations.
- There are several ways you can integrate python data analytics into your existing business intelligence and analytics tools. It includes time series and more complex data structures such as merging, pivoting, and slicing tables to create new views and perspectives on existing sets.
- Python combined with libraries such as iPython and NumPy itself, these tools can form the foundation of a powerful data analytics suite.

2.3.2 Python Libraries

A Python library is a reusable chunk of code that you may want to include in your programs/ projects. The Python Standard Library is a collection of exact syntax, token, and semantics of Python. It comes bundled with core Python distribution. It is written in C and handles functionality like I/O and other core modules. All this functionality together makes Python the language it is.

More than 200 core modules sit at the heart of the standard library. This library ships with Python. But in addition to this library, you can also access a growing collection of several thousand components from the Python Package Index (PyPI). Here we used some Important Python Libraries for Data Analysis:-

1. Pandas

Pandas (Python data analysis) are a must in the data science life cycle. It is themost popular and widely used Python library for data science, along with NumPy in matplotlib.

It is heavily used for data analysis and cleaning. Pandas provides fast, flexible data structures, such as data frame CDs, which are designed to work with structured data very easily and intuitively.

Features:

- Eloquent syntax and rich functionalities that give you the freedom to deal with missing data.
- Enables you to create your own function and run it across a series of data.
- High-level abstraction.
- Contains high-level data structures and manipulation tools.

Applications:

- General data wrangling and data cleaning.
- ETL (extract, transform, load) jobs for data transformation and data storage, as it has excellent support for loading CSV files into its data frame format.
- Used in a variety of academic and commercial areas, including statistics, finance, and neuroscience.
- Time-series-specific functionality, such as date range generation, moving window, linear regression and date shifting.

2. NumPy

NumPy (Numerical Python) is the fundamental package for numerical computation in Python; it contains a powerful N-dimensional array object. It’s a general-purpose array-processing package that provides high-performance multidimensional objects called arrays and tools for working with them. NumPy also addresses the slowness problem partly by providing these multidimensional arrays as well as providing functions and operators that operate efficiently on these arrays.

Features:

- Provides fast, precompiled functions for numerical routines.
- Array-oriented computing for better efficiency.
- Supports an object-oriented approach.
- Compact and faster computations with vectorization.

Applications:

- Extensively used in data analysis
- Creates powerful N-dimensional array
- Forms the base of other libraries, such as SciPy and scikit-learn.
- Replacement of MATLAB when used with SciPy and matplotlib.

3. Scikit-learn

Simple and efficient tools for predictive data analysis. Accessible to everybody, and reusable in various contexts. Scikit-learn, a machine learning library that provides almost all the machine learning algorithms you might need. Scikit-learn is designed to be interpolated into NumPy and SciPy.

Features:

- Datasets. Scikit-learn comes with several inbuilt datasets such as the iris dataset, house prices dataset, diabetes dataset, etc.
- Data Splitting.
- XG Boost.
- Machine learning algorithms.

Applications:

- Clustering
- Classification
- Regression
- Model selection
- Dimensionality reduction
- Preprocessing

2.3.3 t-SNE Algorithm

T-distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions.

t-SNE finds patterns in the data based on the similarity of data points with features, the similarity of points is calculated as the conditional probability that a point A would choose point B as its neighbour.

It then tries to minimize the difference between these conditional probabilities (or similarities) in higher-dimensional and lower-dimensional space for a perfect representation of data points in lower-dimensional space.

The algorithm computes pairwise conditional probabilities and tries to minimize the sum of the difference of the probabilities in higher and lower dimensions. This involves a lot of calculations and computations. So, the algorithm takes a lot of time and space to compute. t-SNE has a quadratic time and space complexity in the number of data points.

t-SNE could be used to investigate, learn, or evaluate segmentation. Often, we select the number of segments prior to modeling or iterating after results. t-SNE can often show clear separation in the data. This can be used prior to using your segmentation model to select a cluster number or after to evaluate if your segments hold up. t-SNE however is not a clustering approach since it does not preserve the inputs like PCA and the values may often change between runs so it's purely for exploration.

2.3.4 Bokeh library

Bokeh is a data visualization library in Python that provides high performance interactive charts and plots. Bokeh output can be obtained in various mediums like notebook, html, and server.
Bokeh provides two visualization interfaces to users:

- bokeh.models : A low level interface that provides high flexibility to application developers.
- bokeh.plotting : A high level interface for creating visual glyphs.

Bokeh primarily focuses on converting the data source into JSON format which then uses as input for BokehJS. Some of the best features of Bokeh are:

- Flexibility: Bokeh provides simple charts and customs charts too for complex use-cases.

- Productivity: Bokeh has an easily compatible nature and can work with Pandas and Jupyter notebooks.
- Styling: We have control of our chart, and we can easily modify the charts by using custom JavaScript.
- Open source: Bokeh provides many examples and ideas to start with and it is distributed under Berkeley Source Distribution (BSD) license.

2.3.5 Platform: Jupyter Notebook

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

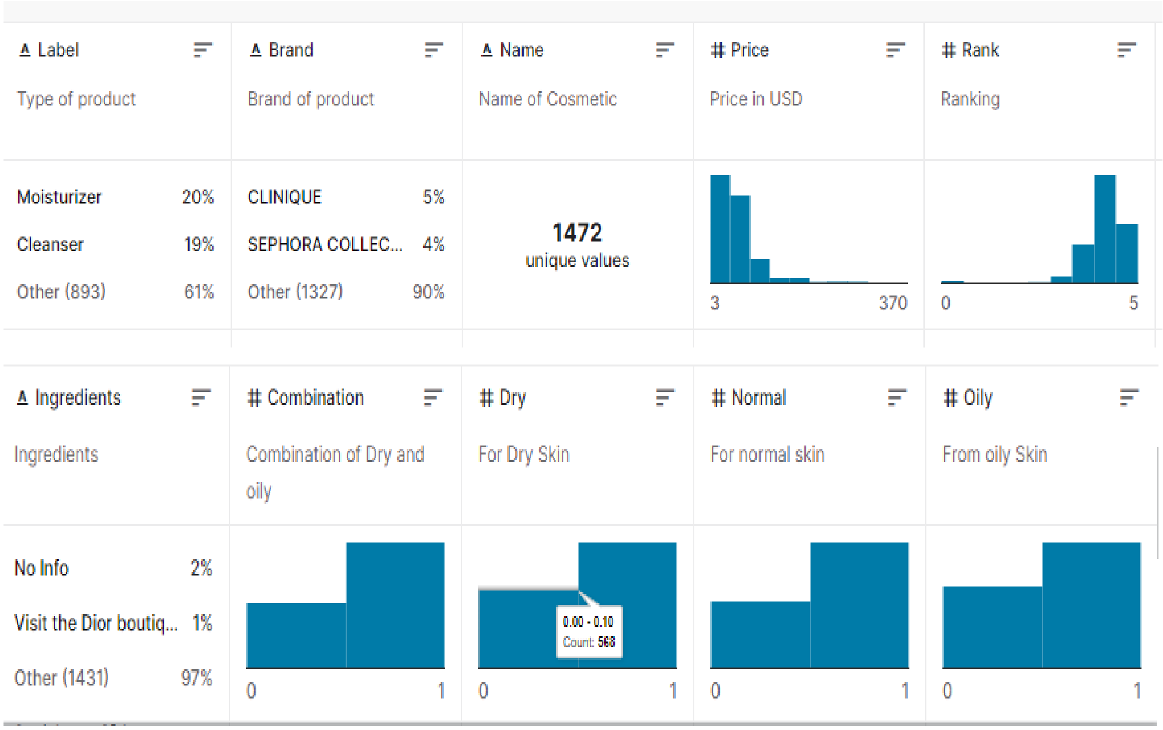
Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R.

3. IMPLEMENTATIONS

3.1 Gathering Requirements and Defining Problem Statement

It is the first step in which requirements are gathered from the resources and required applications are installed followed by defining a problem statement which is to be followed during the development of the project.

Data collection is a systematic approach for gathering and measuring information from a source to choose users for their required skin care product. It helps us to address specific questions, determine outcomes and forecast future probabilities and patterns. This Data set contains product names, Ingredients, skin preference and price. This dataset is stored as a comma separated values) file. Open Jupyter Notebook through Anaconda prompt then Inspect cosmetics.data file.



Label	Brand	Name	# Price	# Rank
Type of product	Brand of product	Name of Cosmetic	Price in USD	Ranking
Moisturizer 20% Cleanser 19% Other (893) 61%	CLINIQUE 5% SEPHORA COLLEC... 4% Other (1327) 90%	1472 unique values	3 370	0 5

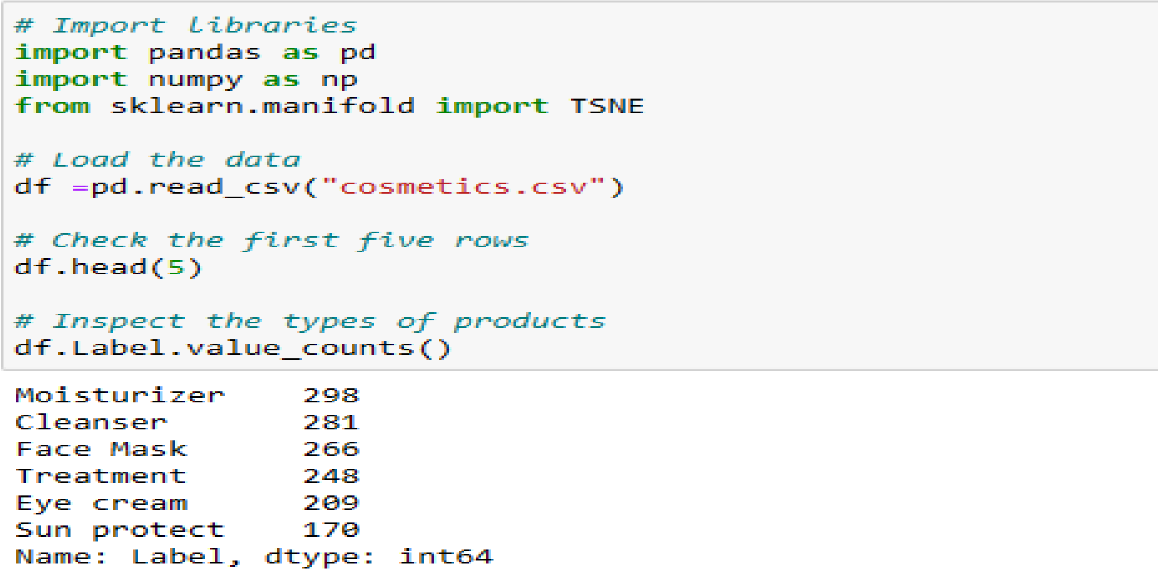
Ingredients	# Combination	# Dry	# Normal	# Oily
Ingredients	Combination of Dry and oily	For Dry Skin	For normal skin	From oily Skin
No Info 2% Visit the Dior boutiq... 1% Other (1431) 97%	0 1	0.00 - 0.10 Count: 568 0 1	0 1	0 1

Data importing is referred to as uploading the required data into the coding environment from internal sources (computer) or external sources (online websites and data repositories).

This data can then be manipulated, aggregated, filtered according to the requirements, and needs of the project. Once the data is imported in the environment, it is converted into a data frame using python library pandas through read.csv (). It is a wrapper function for read.table() that mandates a comma as separator and uses the input file's first line as header that specifies the table's column names. which makes it easy to maintain the data in the form of a table.

First, open Jupyter Notebook for inspecting the dataset and then import the dataset. Once the data is imported into the environment, then import the required libraries for Exploratory Data Analysis. We proceed to load the data into memory and understand the attribute information.

The dataset looks like every column in our DataFrame has the numeric type, which is exactly what we want when building a machine learning model. First, we do Data cleaning, it is an important step in data preprocessing. Checking null values and attribute types are essential.



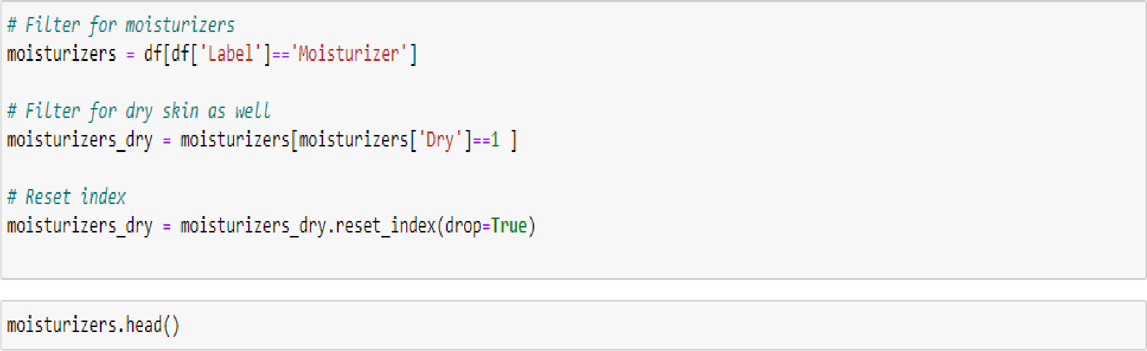
```
# Import Libraries
import pandas as pd
import numpy as np
from sklearn.manifold import TSNE

# Load the data
df =pd.read_csv("cosmetics.csv")

# Check the first five rows
df.head(5)

# Inspect the types of products
df.Label.value_counts()
```

```
Moisturizer     298
Cleanser        281
Face Mask       266
Treatment       248
Eye cream       209
Sun protect     170
Name: Label, dtype: int64
```

The data set consists of 1472 items which includes information about the brand, name, price, rank, skin types, and chemical components of each product. Additionally, star ratings for all 1472 items will be extracted from sephora.com along with the reviewers’ skin types.

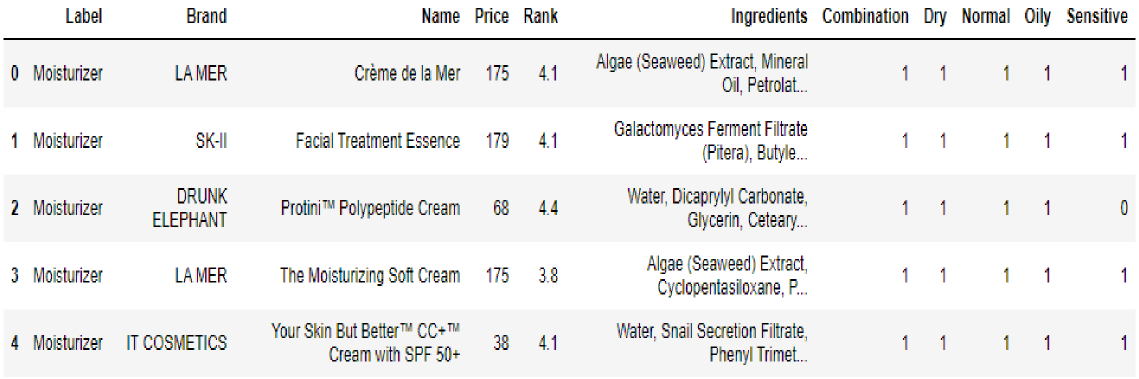
3.2 Focus on one product category and one skin type

There are six categories of product in our data (moisturizers, cleansers, face masks, eye creams, and sun protection) and there are five different skin types (combination, dry, normal, oily, and sensitive). Because individuals have different product needs as well as different skin types, let's set up our workflow so its outputs (a t-SNE model and a visualization of that model) can be customized. First focus on moisturizers for those with dry skin by filtering the data accordingly.



```
# Filter for moisturizers
moisturizers = df[df['Label']=='Moisturizer']

# Filter for dry skin as well
moisturizers_dry = moisturizers[moisturizers['Dry']==1 ]

# Reset index
moisturizers_dry = moisturizers_dry.reset_index(drop=True)
```

```
moisturizers.head()
```



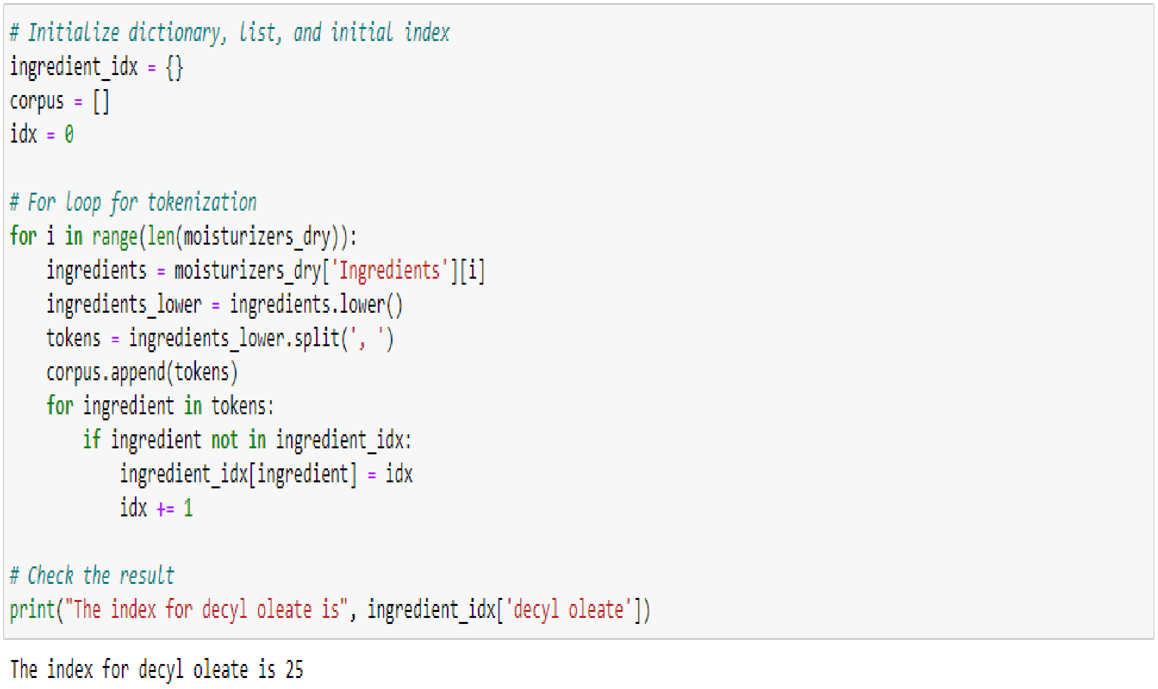
	Label	Brand	Name	Price	Rank	Ingredients	Combination	Dry	Normal	Oily	Sensitive
0	Moisturizer	LA MER	Crème de la Mer	175	4.1	Algae (Seaweed) Extract, Mineral Oil, Petrolat...	1	1	1	1	1
1	Moisturizer	SK-II	Facial Treatment Essence	179	4.1	Galactomyces Ferment Filtrate (Pitera), Butyle...	1	1	1	1	1
2	Moisturizer	DRUNK ELEPHANT	Protini™ Polypeptide Cream	68	4.4	Water, Dicaprylyl Carbonate, Glycerin, Ceteary...	1	1	1	1	0
3	Moisturizer	LA MER	The Moisturizing Soft Cream	175	3.8	Algae (Seaweed) Extract, Cyclopentasiloxane, P...	1	1	1	1	1
4	Moisturizer	IT COSMETICS	Your Skin But Better™ CC+™ Cream with SPF 50+	38	4.1	Water, Snail Secretion Filtrate, Phenyl Trimet...	1	1	1	1	1

3.3 Tokenizing the ingredients

To get to our end goal of comparing ingredients in each product, we first need to do some preprocessing tasks and bookkeeping of the actual words in each product's ingredients list. The first step will be tokenizing the list of ingredients in Ingredients column. After splitting them into tokens, we'll make a binary bag of words.

Then we will create a dictionary with the tokens, ingredient_idx, which will have the following format:

{ "ingredient": index value, … }

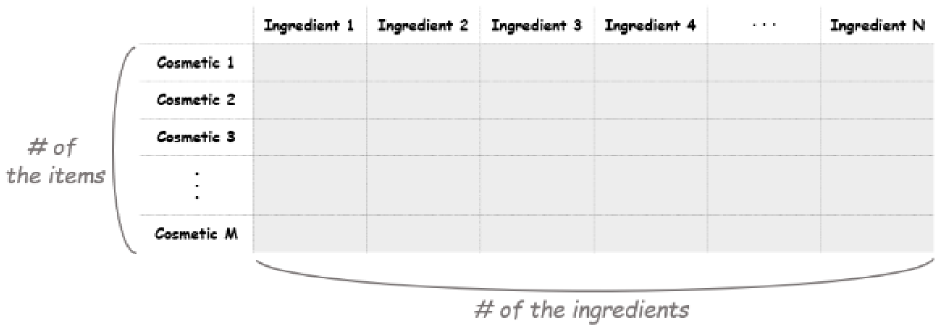


```
# Initialize dictionary, list, and initial index
ingredient_idx = {}
corpus = []
idx = 0

# For loop for tokenization
for i in range(len(moisturizers_dry)):
    ingredients = moisturizers_dry['Ingredients'][i]
    ingredients_lower = ingredients.lower()
    tokens = ingredients_lower.split(', ')
    corpus.append(tokens)
    for ingredient in tokens:
        if ingredient not in ingredient_idx:
            ingredient_idx[ingredient] = idx
            idx += 1

# Check the result
print("The index for decyl oleate is", ingredient_idx['decyl oleate'])
```

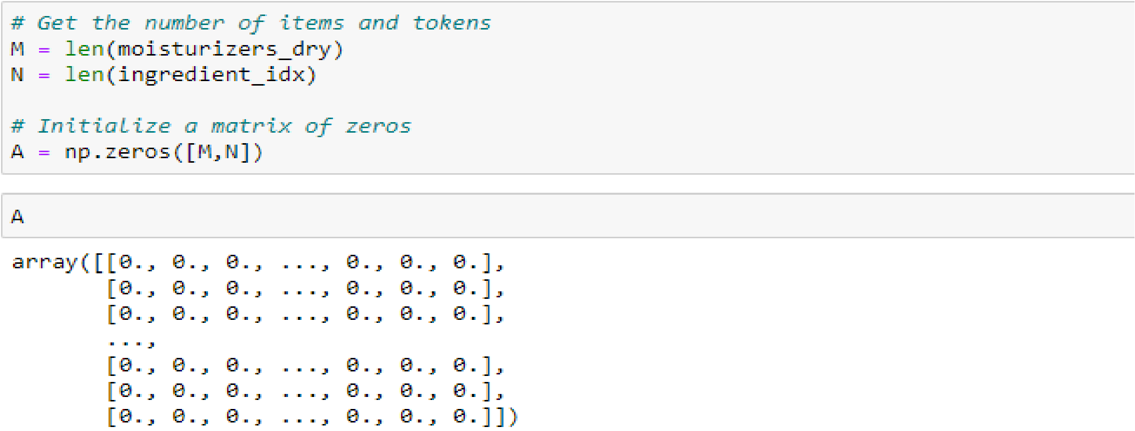
```
The index for decyl oleate is 25
```

3.4 Initializing a document-term matrix (DTM)

The next step is making a document-term matrix (DTM). Here each cosmetic product will correspond to a document, and each chemical composition will correspond to a term. This means we can think of the matrix as a “cosmetic-ingredient” matrix. The size of the matrix should be as the picture shown below.



To create this matrix, we'll first make an empty matrix filled with zeros. The length of the matrix is the total number of cosmetic products in the data. The width of the matrix is the total number of ingredients. After initializing this empty matrix, we'll fill it in the following tasks.



```
# Get the number of items and tokens
M = len(moisturizers_dry)
N = len(ingredient_idx)

# Initialize a matrix of zeros
A = np.zeros([M,N])
```

```
A
```

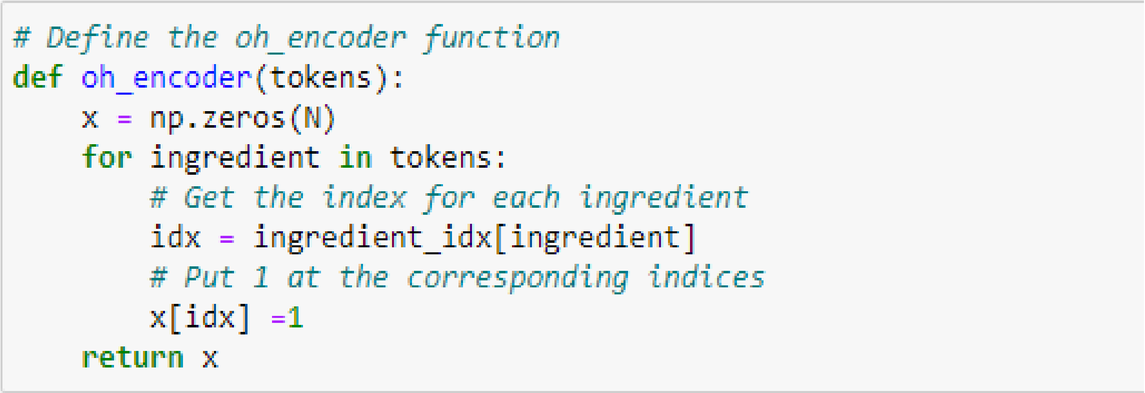
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

3.5 Creating a counter function

Before we can fill the matrix, let's create a function to count the tokens (i.e., an ingredients list) for each row. Our end goal is to fill the matrix with 1 or 0: if an ingredient is in a cosmetic, the value is 1. If not, it remains 0. The name of this function, oh_encoder, will become clear next.

One hot encoding is a process of converting categorical data variables so they can be provided to machine learning algorithms to improve predictions. One hot encoding is a crucial part of feature engineering for machine learning.

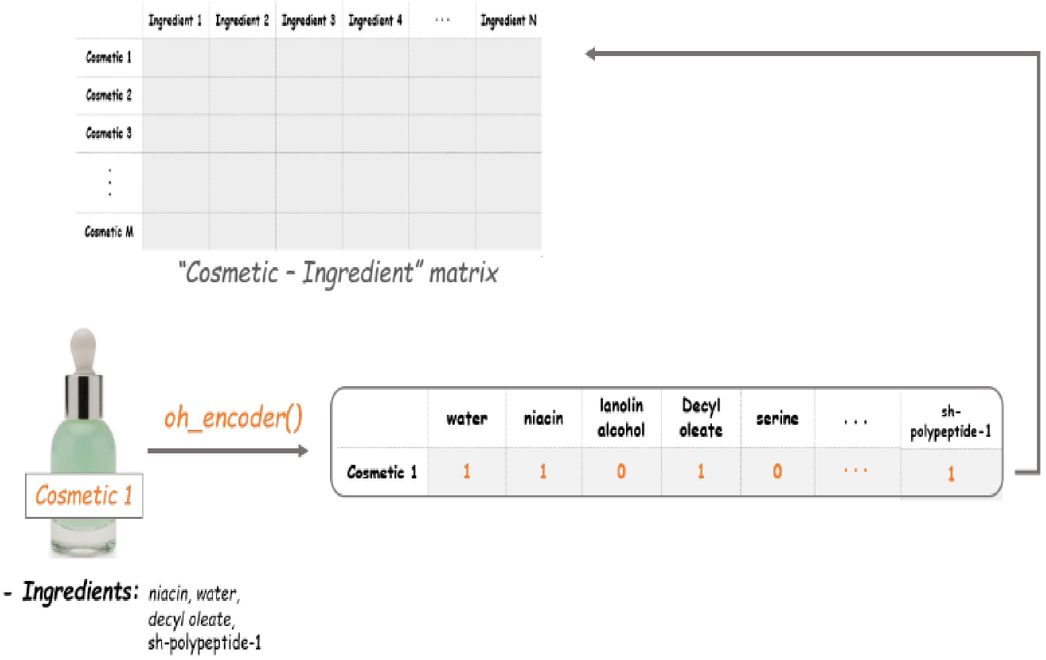
One hot encoding is useful for data that has no relationship to each other. Machine learning algorithms treat the order of numbers as an attribute of significance. In other words, they will read a higher number as better or more important than a lower number. One hot encoding makes our training data more useful and expressive, and it can be rescaled easily. By using numeric values, we more easily determine a probability for our values.



```python
# Define the oh_encoder function
def oh_encoder(tokens):
    x = np.zeros(N)
    for ingredient in tokens:
        # Get the index for each ingredient
        idx = ingredient_idx[ingredient]
        # Put 1 at the corresponding indices
        x[idx] =1
    return x
```

3.6 The Cosmetic-Ingredient matrix

Now we'll apply the oh_encoder() function to the tokens in corpus and set the values at each row of this matrix. So, the result will tell us what ingredients each item is composed of. For example, if a cosmetic item contains water, niacin, decyl aleate and sh-polypeptide-1, the outcome of this item will be as follows. This is what we call one-hot encoding. By encoding each ingredient in the items, the Cosmetic-Ingredient matrix will be filled with binary values.



	Ingredient 1	Ingredient 2	Ingredient 3	Ingredient 4	···	Ingredient N
Cosmetic 1						
Cosmetic 2						
Cosmetic 3						
⋮						
Cosmetic M						

"Cosmetic - Ingredient" matrix

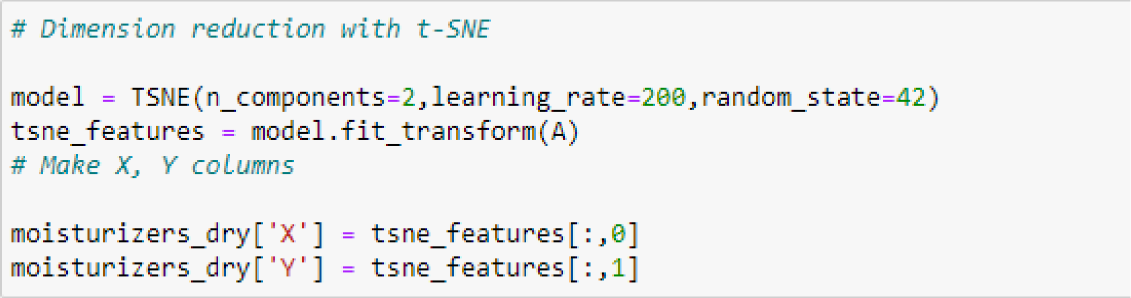
	water	niacin	lanolin alcohol	Decyl oleate	serine	···	sh-polypeptide-1
Cosmetic 1	1	1	0	1	0	···	1

oh_encoder()
Cosmetic 1
- Ingredients: niacin, water, decyl oleate, sh-polypeptide-1

3.7 Dimension reduction with t-SNE

The dimensions of the existing matrix are (190, 2233), which means there are 2233 features in our data. For visualization, we should downsize this into two dimensions. We'll use t-SNE for reducing the dimension of the data here.

T-distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear dimensionality reduction technique that is well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. Specifically, this technique can reduce the dimension of data while keeping the similarities between the instances. This enables us to make a plot on the coordinate plane, which can be said to be vectorizing. All these cosmetic items in our data will be vectorized into two-dimensional coordinates, and the distances between the points will indicate the similarities between the items.

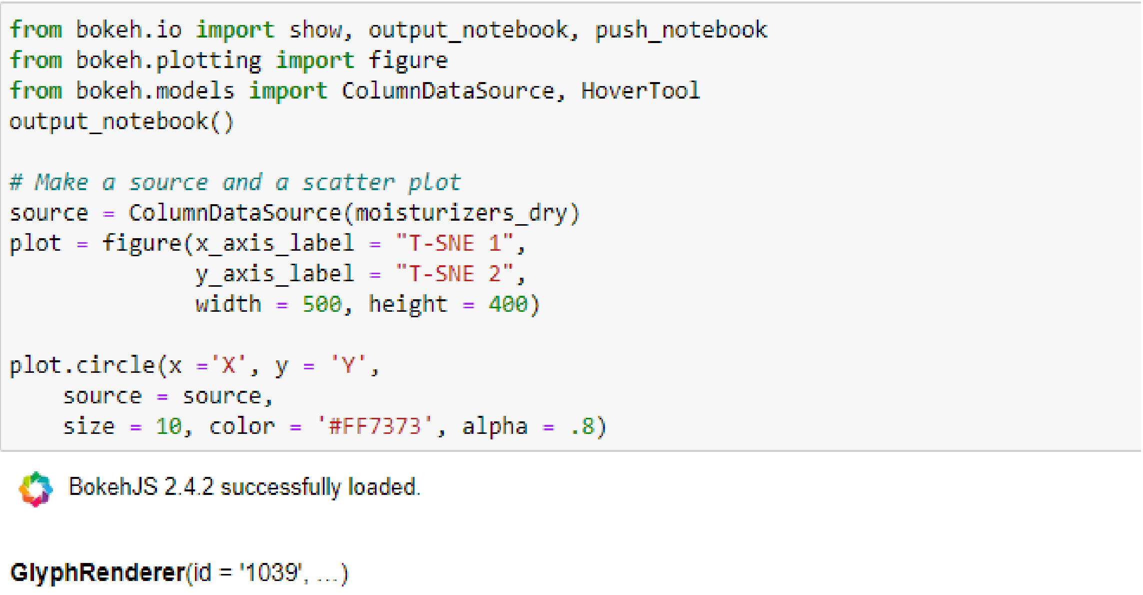


```
# Dimension reduction with t-SNE

model = TSNE(n_components=2,learning_rate=200,random_state=42)
tsne_features = model.fit_transform(A)
# Make X, Y columns

moisturizers_dry['X'] = tsne_features[:,0]
moisturizers_dry['Y'] = tsne_features[:,1]
```

3.8 Map the items with Bokeh

With a wide array of widgets, plot tools, and UI events that can trigger real Python callbacks, the Bokeh server is the bridge that lets you connect these tools to rich, interactive visualizations in the browser.

With the t-SNE values, we can plot all our items on the coordinate plane. And the coolest part here is that it will also show us the name, the brand, the price, and the rank of each item. Make a scatter plot using Bokeh and add a hover tool to show that information.

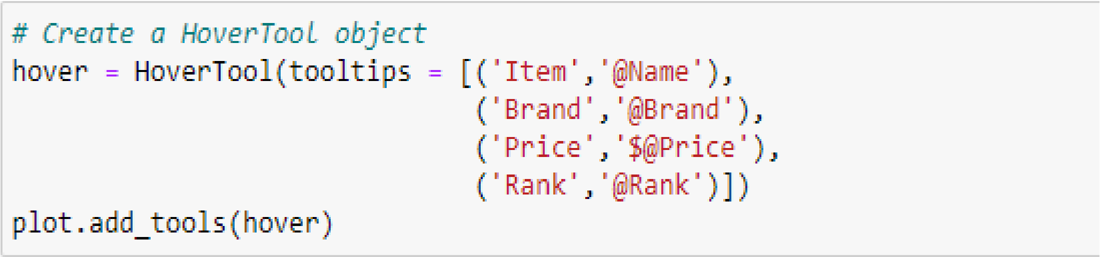


```
from bokeh.io import show, output_notebook, push_notebook
from bokeh.plotting import figure
from bokeh.models import ColumnDataSource, HoverTool
output_notebook()

# Make a source and a scatter plot
source = ColumnDataSource(moisturizers_dry)
plot = figure(x_axis_label = "T-SNE 1",
              y_axis_label = "T-SNE 2",
              width = 500, height = 400)

plot.circle(x ='X', y = 'Y',
    source = source,
    size = 10, color = '#FF7373', alpha = .8)
```

```
BokehJS 2.4.2 successfully loaded.

GlyphRenderer(id = '1039', …)
```

3.9 Adding a hover tool

Adding a hover tool allows us to check the information of each item whenever the cursor is directly over a glyph. We'll add tooltips with each product's name, brand, price, and rank (i.e., rating).

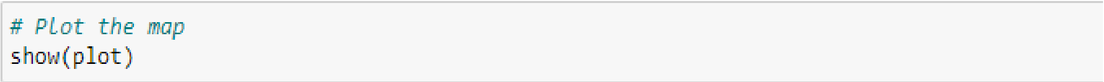


```
# Create a HoverTool object
hover = HoverTool(tooltips = [('Item','@Name'),
                              ('Brand','@Brand'),
                              ('Price','$@Price'),
                              ('Rank','@Rank')])
plot.add_tools(hover)
```

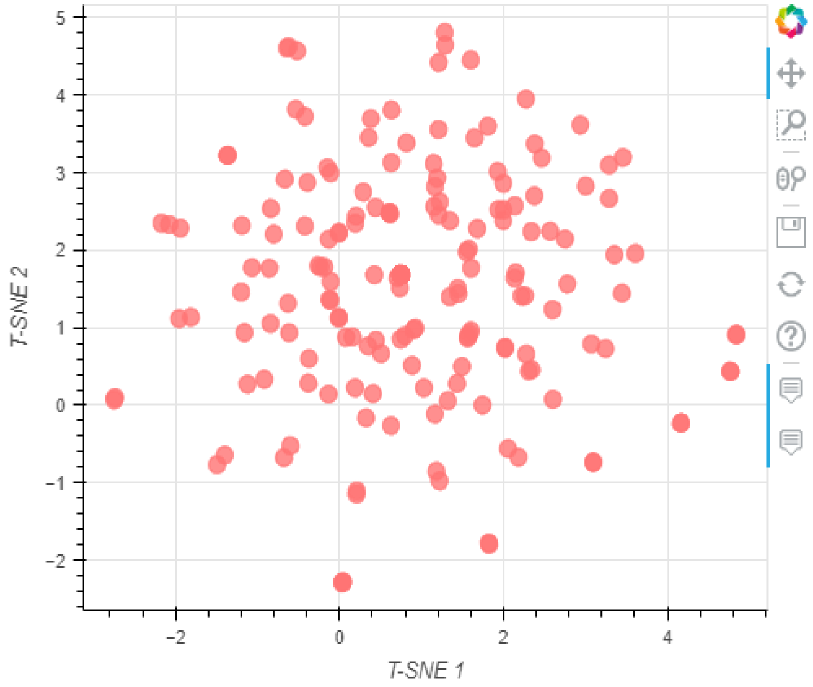
3.10 Mapping the cosmetic items

Each point on the plot corresponds to the cosmetic items. Then what do the axes mean here? The axes of a t-SNE plot aren't easily interpretable in terms of the original data. Like mentioned above, t-SNE is a visualizing technique to plot high-dimensional data in a low-dimensional space. Therefore, it's not desirable to interpret a t-SNE plot quantitatively.

Instead, what we can get from this map is the distance between the points (which items are close, and which are far apart). The closer the distance between the two items is, the more similar the composition they have. Therefore, this enables us to compare the items without having any chemistry background.



```
# Plot the map
show(plot)
```



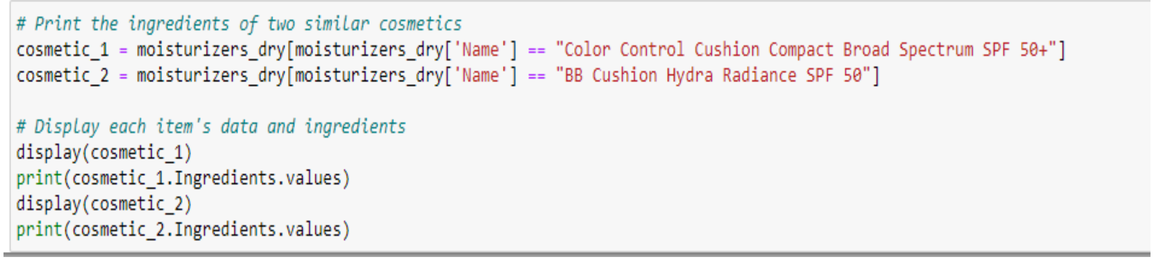
3.11 Comparing two products

Since there are so many cosmetics and so many ingredients, the plot doesn't have many super obvious patterns that simpler t-SNE plots can have (example). Our plot requires some digging to find insights. AmorePacific's Color Control Cushion Compact Broad Spectrum SPF 50+.

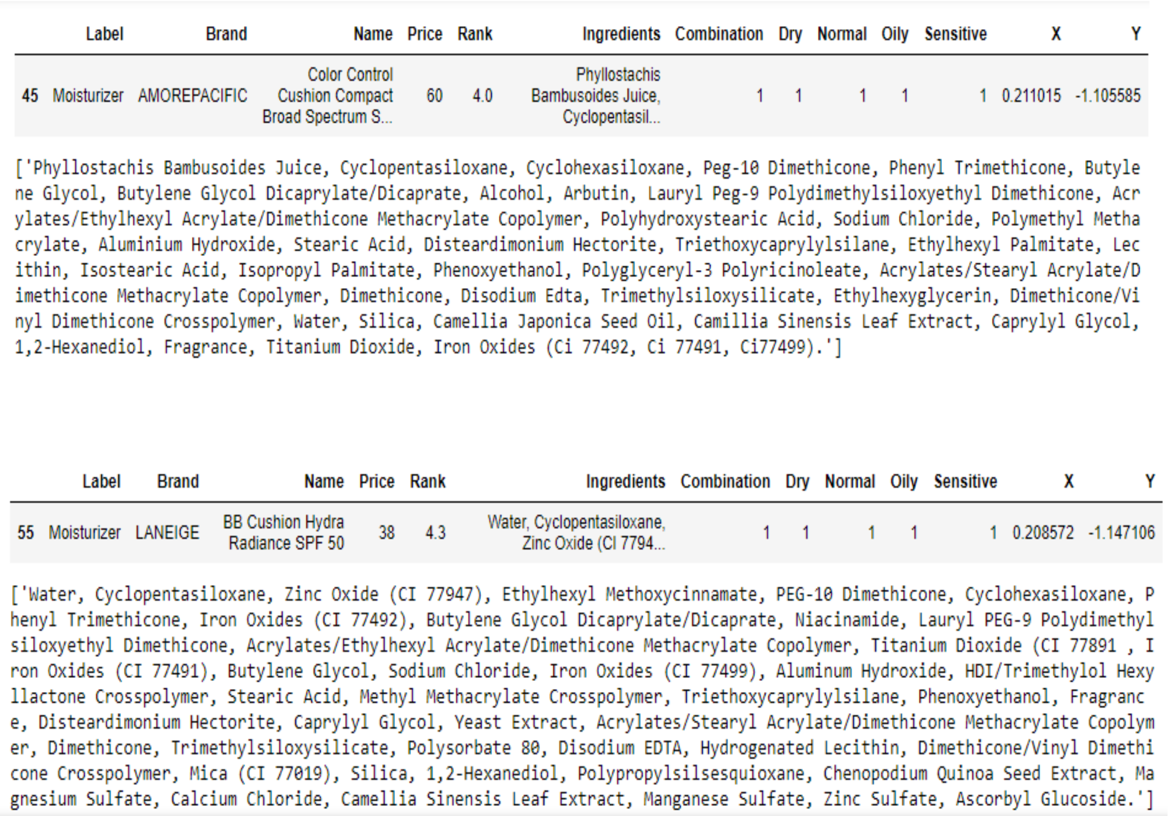
We could find this product on the plot and see if a similar product(s) exists. And it turns out it does! If we look at the points furthest left on the plot, we see LANEIGE's BB Cushion Hydra Radiance SPF 50 essentially overlaps with the AmorePacific product.

By looking at the ingredients, we can visually confirm the compositions of the products are similar (though it is difficult to do, which is why we did this analysis in the first place!), plus LANEIGE's version is $22 cheaper and has higher ratings.

It's not perfect, but it's useful. In real life, we can use our little ingredient-based recommendation engine to help us make educated cosmetic purchase choices.



```python
# Print the ingredients of two similar cosmetics
cosmetic_1 = moisturizers_dry[moisturizers_dry['Name'] == "Color Control Cushion Compact Broad Spectrum SPF 50+"]
cosmetic_2 = moisturizers_dry[moisturizers_dry['Name'] == "BB Cushion Hydra Radiance SPF 50"]

# Display each item's data and ingredients
display(cosmetic_1)
print(cosmetic_1.Ingredients.values)
display(cosmetic_2)
print(cosmetic_2.Ingredients.values)
```



	Label	Brand	Name	Price	Rank	Ingredients	Combination	Dry	Normal	Oily	Sensitive	X	Y
45	Moisturizer	AMOREPACIFIC	Color Control Cushion Compact Broad Spectrum S...	60	4.0	Phyllostachis Bambusoides Juice, Cyclopentasil...	1	1	1	1	1	0.211015	-1.105585

```
['Phyllostachis Bambusoides Juice, Cyclopentasiloxane, Cyclohexasiloxane, Peg-10 Dimethicone, Phenyl Trimethicone, Butyle
ne Glycol, Butylene Glycol Dicaprylate/Dicaprate, Alcohol, Arbutin, Lauryl Peg-9 Polydimethylsiloxyethyl Dimethicone, Acr
ylates/Ethylhexyl Acrylate/Dimethicone Methacrylate Copolymer, Polyhydroxystearic Acid, Sodium Chloride, Polymethyl Metha
crylate, Aluminium Hydroxide, Stearic Acid, Disteardimonium Hectorite, Triethoxycaprylylsilane, Ethylhexyl Palmitate, Lec
ithin, Isostearic Acid, Isopropyl Palmitate, Phenoxyethanol, Polyglyceryl-3 Polyricinoleate, Acrylates/Stearyl Acrylate/D
imethicone Methacrylate Copolymer, Dimethicone, Disodium Edta, Trimethylsiloxysilicate, Ethylhexyglycerin, Dimethicone/Vi
nyl Dimethicone Crosspolymer, Water, Silica, Camellia Japonica Seed Oil, Camillia Sinensis Leaf Extract, Caprylyl Glycol,
1,2-Hexanediol, Fragrance, Titanium Dioxide, Iron Oxides (Ci 77492, Ci 77491, Ci77499).']
```

	Label	Brand	Name	Price	Rank	Ingredients	Combination	Dry	Normal	Oily	Sensitive	X	Y
55	Moisturizer	LANEIGE	BB Cushion Hydra Radiance SPF 50	38	4.3	Water, Cyclopentasiloxane, Zinc Oxide (CI 7794...	1	1	1	1	1	0.208572	-1.147106

```
['Water, Cyclopentasiloxane, Zinc Oxide (CI 77947), Ethylhexyl Methoxycinnamate, PEG-10 Dimethicone, Cyclohexasiloxane, P
henyl Trimethicone, Iron Oxides (CI 77492), Butylene Glycol Dicaprylate/Dicaprate, Niacinamide, Lauryl PEG-9 Polydimethyl
siloxyethyl Dimethicone, Acrylates/Ethylhexyl Acrylate/Dimethicone Methacrylate Copolymer, Titanium Dioxide (CI 77891 , I
ron Oxides (CI 77491), Butylene Glycol, Sodium Chloride, Iron Oxides (CI 77499), Aluminum Hydroxide, HDI/Trimethylol Hexy
llactone Crosspolymer, Stearic Acid, Methyl Methacrylate Crosspolymer, Triethoxycaprylylsilane, Phenoxyethanol, Fragranc
e, Disteardimonium Hectorite, Caprylyl Glycol, Yeast Extract, Acrylates/Stearyl Acrylate/Dimethicone Methacrylate Copolym
er, Dimethicone, Trimethylsiloxysilicate, Polysorbate 80, Disodium EDTA, Hydrogenated Lecithin, Dimethicone/Vinyl Dimethi
cone Crosspolymer, Mica (CI 77019), Silica, 1,2-Hexanediol, Polypropylsilsesquioxane, Chenopodium Quinoa Seed Extract, Ma
gnesium Sulfate, Calcium Chloride, Camellia Sinensis Leaf Extract, Manganese Sulfate, Zinc Sulfate, Ascorbyl Glucoside.']
```

4. CONCLUSION

This proposal presents an ingredient-based recommendation engine which assesses the similarity of the composition of the ingredients within the products. Instead of being recommended within the same category, the new system recommends products in different categories to allow more effective recommendations for one skin type. It also gives an option to users to provide minimal input to receive skin care product suggestions. The system will be validated by ingredients of each product and comparing them with the results.

In the future, the system can be improved by incorporating brand, personal profile or price preferences when making recommendations. With a suitable data set, you can also try to implement a hybrid recommender system.

5. REFERENCES

Data Collection

Input data and statistics:

a. https://www.sephora.com/data
b. https://www.academia.edu/63711006/A_Content_based_Skincare_Product_Recommendation_System
c. https://www.realsimple.com/beauty-fashion/skincare/how-to-choose-skin-care-product
d. https://www.researchgate.net/publication/271387744_Hazardous_Ingredients_in_Cosmetics_and_Personal_Care_Products_and_Health_Concern_A_Review

Programming References

The following websites have been referred for Python coding and Jupyter Notebook tutorials:

a. https://datascienceplus.com/category/programming

b. https://www.python.org/

c. https://bokeh.org/

d. https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

e. https://www.coursera.org/learn/data-analysis-with-python

f. https://www.educative.io/blog/one-hot-encoding

g. https://jupyter.org/

h. https://www.anaconda.com/products/distribution