```
pip install kaggle
```

```
Requirement already satisfied: kaggle in
/opt/anaconda3/lib/python3.11/site-packages (1.7.4.2)
Requirement already satisfied: bleach in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (4.1.0)
Requirement already satisfied: certifi>=14.05.14 in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (2025.1.31)
Requirement already satisfied: charset-normalizer in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (2.0.4)
Requirement already satisfied: idna in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (3.4)
Requirement already satisfied: protobuf in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (4.25.3)
Requirement already satisfied: python-dateutil>=2.5.3 in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (2.8.2)
Requirement already satisfied: python-slugify in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (5.0.2)
Requirement already satisfied: requests in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (2.31.0)
Requirement already satisfied: setuptools>=21.0.0 in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (68.2.2)
Requirement already satisfied: six>=1.10 in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (1.16.0)
Requirement already satisfied: text-unidecode in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (1.3)
Requirement already satisfied: tqdm in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (4.65.0)
Requirement already satisfied: urllib3>=1.15.1 in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (2.0.7)
Requirement already satisfied: webencodings in
/opt/anaconda3/lib/python3.11/site-packages (from kaggle) (0.5.1)
Requirement already satisfied: packaging in
/opt/anaconda3/lib/python3.11/site-packages (from bleach->kaggle)
(23.1)
Note: you may need to restart the kernel to use updated packages.
```

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
product_df = pd.read_csv("product_info.csv")
print("Product Info DataFrame loaded. Shape:", product_df.shape)
product_df.head()
```

```
Product Info DataFrame loaded. Shape: (8494, 27)
```

```
  product_id               product_name  brand_id brand_name  
loves_count  \
0    P473671    Fragrance Discovery Set      6342      19-69
6320
1    P473668    La Habana Eau de Parfum      6342      19-69
3827
2    P473662  Rainbow Bar Eau de Parfum      6342      19-69
3253
3    P473660       Kasbah Eau de Parfum      6342      19-69
3018
4    P473658  Purple Haze Eau de Parfum      6342      19-69
2691

   rating  reviews            size                      variation_type
\
0  3.6364     11.0             NaN                                 NaN

1  4.1538     13.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

2  4.2500     16.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

3  4.4762     21.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

4  3.2308     13.0  3.4 oz/ 100 mL  Size + Concentration + Formulation


  variation_value  ... online_only out_of_stock  sephora_exclusive  \
0             NaN  ...           1            0                  0
1  3.4 oz/ 100 mL  ...           1            0                  0
2  3.4 oz/ 100 mL  ...           1            0                  0
3  3.4 oz/ 100 mL  ...           1            0                  0
4  3.4 oz/ 100 mL  ...           1            0                  0

                                          highlights  primary_category
\
0  ['Unisex/ Genderless Scent', 'Warm &Spicy Scen...         Fragrance

1  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

2  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

3  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

4  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance


   secondary_category  tertiary_category  child_count  child_max_price
\
0   Value & Gift Sets  Perfume Gift Sets            0              NaN

1               Women            Perfume            2             85.0

2               Women            Perfume            2             75.0
```

```
3                  Women            Perfume             2             75.0

4                  Women            Perfume             2             75.0


   child_min_price
0              NaN
1             30.0
2             30.0
3             30.0
4             30.0

[5 rows x 27 columns]
```

```
import os
print(os.getcwd())
```

```
/Users/alexandriapetersen/Downloads
```

```
import glob

csv_files = glob.glob("*.csv")
print(csv_files)
```

```
['product_info.csv', 'reviews_0-250.csv', 'reviews_1250-end.csv',
'reviews_750-1250.csv', 'image_labels.csv', 'target.csv',
'LaptopSales2008.csv', 'reviews_250-500.csv', 'reviews_500-750.csv']
```

```
import os
print(os.getcwd())
```

```
/Users/alexandriapetersen/Downloads
```

```
import os

os.chdir("/Users/alexandriapetersen/Downloads")
print("Now in:", os.getcwd())
```

```
Now in: /Users/alexandriapetersen/Downloads
```

```
product_df = pd.read_csv("product_info.csv")
print("Product Info DataFrame loaded. Shape:", product_df.shape)
product_df.head()
```

```
Product Info DataFrame loaded. Shape: (8494, 27)
```

```
  product_id               product_name  brand_id brand_name
loves_count  \
0    P473671    Fragrance Discovery Set      6342      19-69
6320
1    P473668    La Habana Eau de Parfum      6342      19-69
```

```
3827
2    P473662  Rainbow Bar Eau de Parfum      6342      19-69
3253
3    P473660       Kasbah Eau de Parfum      6342      19-69
3018
4    P473658  Purple Haze Eau de Parfum      6342      19-69
2691

   rating  reviews            size                      variation_type
\
0  3.6364     11.0             NaN                                 NaN

1  4.1538     13.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

2  4.2500     16.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

3  4.4762     21.0  3.4 oz/ 100 mL  Size + Concentration + Formulation

4  3.2308     13.0  3.4 oz/ 100 mL  Size + Concentration + Formulation


  variation_value  ... online_only out_of_stock  sephora_exclusive  \
0             NaN  ...           1            0                  0
1  3.4 oz/ 100 mL  ...           1            0                  0
2  3.4 oz/ 100 mL  ...           1            0                  0
3  3.4 oz/ 100 mL  ...           1            0                  0
4  3.4 oz/ 100 mL  ...           1            0                  0

                                          highlights  primary_category
\
0  ['Unisex/ Genderless Scent', 'Warm &Spicy Scen...         Fragrance

1  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

2  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

3  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance

4  ['Unisex/ Genderless Scent', 'Layerable Scent'...         Fragrance


   secondary_category  tertiary_category  child_count  child_max_price
\
0   Value & Gift Sets  Perfume Gift Sets            0              NaN

1               Women            Perfume            2             85.0

2               Women            Perfume            2             75.0

3               Women            Perfume            2             75.0
```

```
4                 Women             Perfume                 2              75.0


   child_min_price
0              NaN
1             30.0
2             30.0
3             30.0
4             30.0

[5 rows x 27 columns]
```

```
import pandas as pd
import glob

# Load all review files
review_files = glob.glob("reviews_*.csv")

# Combine into one DataFrame
reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)

print("Reviews DataFrame loaded. Shape:", reviews_df.shape)
reviews_df.head()
```

```
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/
ipykernel_61923/3215166159.py:8: DtypeWarning: Columns (1) have mixed
types. Specify dtype option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/ipykernel_61923/32151
66159.py:8: DtypeWarning: Columns (1) have mixed types. Specify dtype
option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/ipykernel_61923/32151
66159.py:8: DtypeWarning: Columns (1) have mixed types. Specify dtype
option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)

Reviews DataFrame loaded. Shape: (1094411, 19)

   Unnamed: 0    author_id  rating  is_recommended  helpfulness  \
0           0   1741593524       5             1.0          1.0
1           1  31423088263       1             0.0          NaN
2           2   5061282401       5             1.0          NaN
3           3   6083038851       5             1.0          NaN
4           4  47056667835       5             1.0          NaN

   total_feedback_count  total_neg_feedback_count
```

```
total_pos_feedback_count  \
0                     2                         0
2
1                     0                         0
0
2                     0                         0
0
3                     0                         0
0
4                     0                         0
0

  submission_time
review_text  \
0      2023-02-01  I use this with the Nudestix “Citrus Clean Bal...

1      2023-03-21  I bought this lip mask after reading the revie...

2      2023-03-21  My review title says it all! I get so excited ...

3      2023-03-20  I’ve always loved this formula for a long time...

4      2023-03-20  If you have dry cracked lips, this is a must h...


                       review_title skin_tone eye_color
skin_type  \
0  Taught me how to double cleanse!       NaN     brown          dry

1                      Disappointed       NaN       NaN          NaN

2               New Favorite Routine     light     brown          dry

3   Can't go wrong with any of them       NaN     brown  combination

4                    A must have !!!     light     hazel  combination


  hair_color product_id
product_name  \
0      black    P504322                     Gentle Hydra-Gel Face
Cleanser
1        NaN    P420652  Lip Sleeping Mask Intense Hydration with
Vitam...
2     blonde    P420652  Lip Sleeping Mask Intense Hydration with
Vitam...
3      black    P420652  Lip Sleeping Mask Intense Hydration with
Vitam...
4        NaN    P420652  Lip Sleeping Mask Intense Hydration with
Vitam...
```

```
  brand_name  price_usd
0   NUDESTIX       19.0
1    LANEIGE       24.0
2    LANEIGE       24.0
3    LANEIGE       24.0
4    LANEIGE       24.0
```

```
print("Reviews columns:\n", reviews_df.columns)
print("\nProduct columns:\n", product_df.columns)
```

```
Reviews columns:
 Index(['Unnamed: 0', 'author_id', 'rating', 'is_recommended',
'helpfulness',
       'total_feedback_count', 'total_neg_feedback_count',
       'total_pos_feedback_count', 'submission_time', 'review_text',
       'review_title', 'skin_tone', 'eye_color', 'skin_type',
'hair_color',
       'product_id', 'product_name', 'brand_name', 'price_usd'],
      dtype='object')

Product columns:
 Index(['product_id', 'product_name', 'brand_id', 'brand_name',
'loves_count',
       'rating', 'reviews', 'size', 'variation_type',
'variation_value',
       'variation_desc', 'ingredients', 'price_usd',
'value_price_usd',
       'sale_price_usd', 'limited_edition', 'new', 'online_only',
       'out_of_stock', 'sephora_exclusive', 'highlights',
'primary_category',
       'secondary_category', 'tertiary_category', 'child_count',
       'child_max_price', 'child_min_price'],
      dtype='object')
```

```
# Merge on 'product_id'
merged_df = reviews_df.merge(product_df, on='product_id', how='left')

# Check results
print("Merged DataFrame shape:", merged_df.shape)
merged_df.head()
```

```
Merged DataFrame shape: (1094411, 45)
```

```
   Unnamed: 0    author_id  rating_x  is_recommended  helpfulness  \
0           0   1741593524         5             1.0          1.0
1           1  31423088263         1             0.0          NaN
2           2   5061282401         5             1.0          NaN
3           3   6083038851         5             1.0          NaN
4           4  47056667835         5             1.0          NaN

   total_feedback_count  total_neg_feedback_count
```

```
total_pos_feedback_count  \
0                     2                         0
2
1                     0                         0
0
2                     0                         0
0
3                     0                         0
0
4                     0                         0
0

  submission_time
review_text  ...  \
0      2023-02-01  I use this with the Nudestix “Citrus Clean
Bal...  ...
1      2023-03-21  I bought this lip mask after reading the
revie...  ...
2      2023-03-21  My review title says it all! I get so
excited ...  ...
3      2023-03-20  I’ve always loved this formula for a long
time...  ...
4      2023-03-20  If you have dry cracked lips, this is a must
h...  ...

  online_only out_of_stock sephora_exclusive  \
0           1            0                 0
1           0            0                 1
2           0            0                 1
3           0            0                 1
4           0            0                 1

                                          highlights primary_category
\
0                           ['Clean at Sephora']         Skincare

1  ['allure 2019 Best of Beauty Award Winner', 'C...         Skincare

2  ['allure 2019 Best of Beauty Award Winner', 'C...         Skincare

3  ['allure 2019 Best of Beauty Award Winner', 'C...         Skincare

4  ['allure 2019 Best of Beauty Award Winner', 'C...         Skincare


       secondary_category tertiary_category child_count
child_max_price  \
0               Cleansers               NaN           0
NaN
1  Lip Balms & Treatments               NaN           3
```

```
24.0
2  Lip Balms & Treatments                    NaN           3
24.0
3  Lip Balms & Treatments                    NaN           3
24.0
4  Lip Balms & Treatments                    NaN           3
24.0

  child_min_price
0             NaN
1            24.0
2            24.0
3            24.0
4            24.0

[5 rows x 45 columns]
```

```
missing = merged_df[merged_df['ingredients'].isnull()]
print("Reviews with missing product info:", missing.shape[0])
```

```
Reviews with missing product info: 22025
```

RATINGS DISTRIBUTION

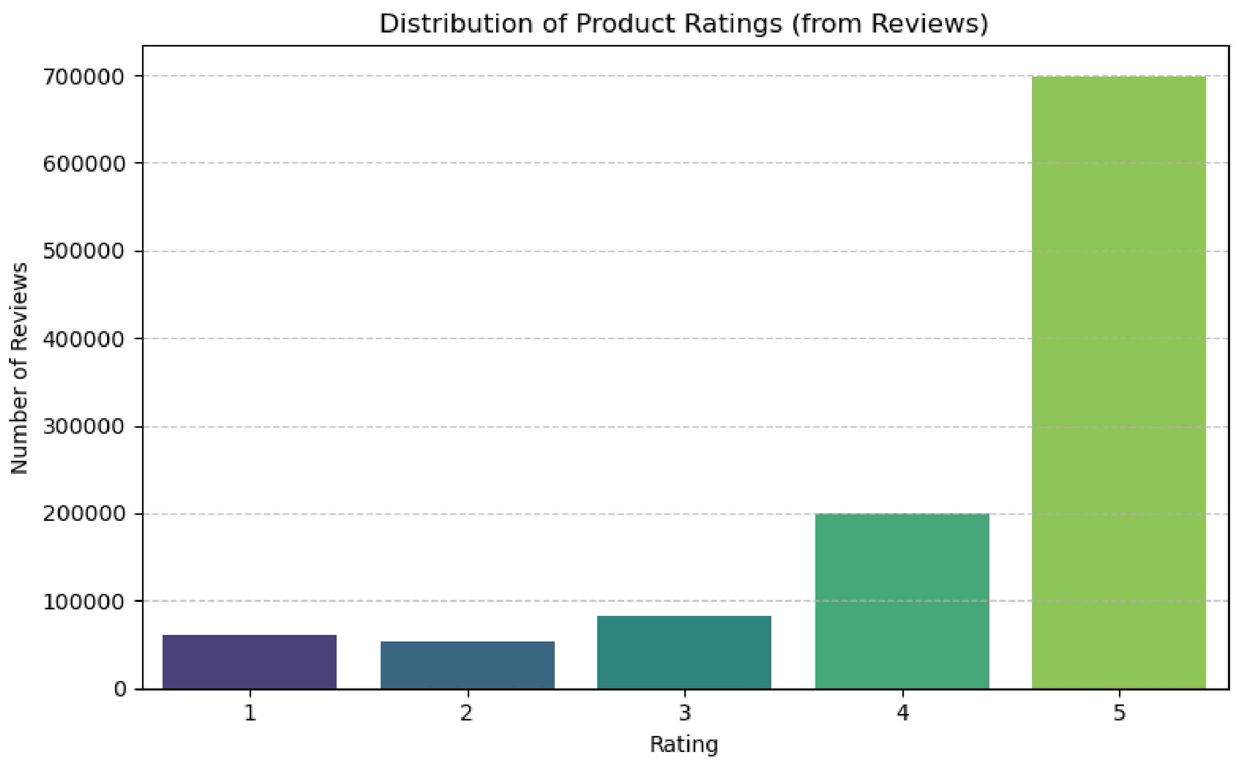
```
print("Columns in merged_df:")
print(merged_df.columns)
```

```
Columns in merged_df:
Index(['Unnamed: 0', 'author_id', 'rating_x', 'is_recommended',
'helpfulness',
       'total_feedback_count', 'total_neg_feedback_count',
       'total_pos_feedback_count', 'submission_time', 'review_text',
       'review_title', 'skin_tone', 'eye_color', 'skin_type',
'hair_color',
       'product_id', 'product_name_x', 'brand_name_x', 'price_usd_x',
       'product_name_y', 'brand_id', 'brand_name_y', 'loves_count',
'rating_y',
       'reviews', 'size', 'variation_type', 'variation_value',
       'variation_desc', 'ingredients', 'price_usd_y',
'value_price_usd',
       'sale_price_usd', 'limited_edition', 'new', 'online_only',
       'out_of_stock', 'sephora_exclusive', 'highlights',
'primary_category',
       'secondary_category', 'tertiary_category', 'child_count',
       'child_max_price', 'child_min_price'],
      dtype='object')
```

```
print(merged_df.columns)
```

```
Index(['Unnamed: 0', 'author_id', 'rating_x', 'is_recommended',
'helpfulness',
       'total_feedback_count', 'total_neg_feedback_count',
       'total_pos_feedback_count', 'submission_time', 'review_text',
       'review_title', 'skin_tone', 'eye_color', 'skin_type',
'hair_color',
       'product_id', 'product_name_x', 'brand_name_x', 'price_usd_x',
       'product_name_y', 'brand_id', 'brand_name_y', 'loves_count',
'rating_y',
       'reviews', 'size', 'variation_type', 'variation_value',
       'variation_desc', 'ingredients', 'price_usd_y',
'value_price_usd',
       'sale_price_usd', 'limited_edition', 'new', 'online_only',
       'out_of_stock', 'sephora_exclusive', 'highlights',
'primary_category',
       'secondary_category', 'tertiary_category', 'child_count',
       'child_max_price', 'child_min_price'],
      dtype='object')
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 5))
sns.countplot(x=merged_df['rating_x'].dropna(), palette='viridis')

plt.title('Distribution of Product Ratings (from Reviews)')
plt.xlabel('Rating')
plt.ylabel('Number of Reviews')
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



```
merged_df.rename(columns={'rating_x': 'rating'}, inplace=True)
```

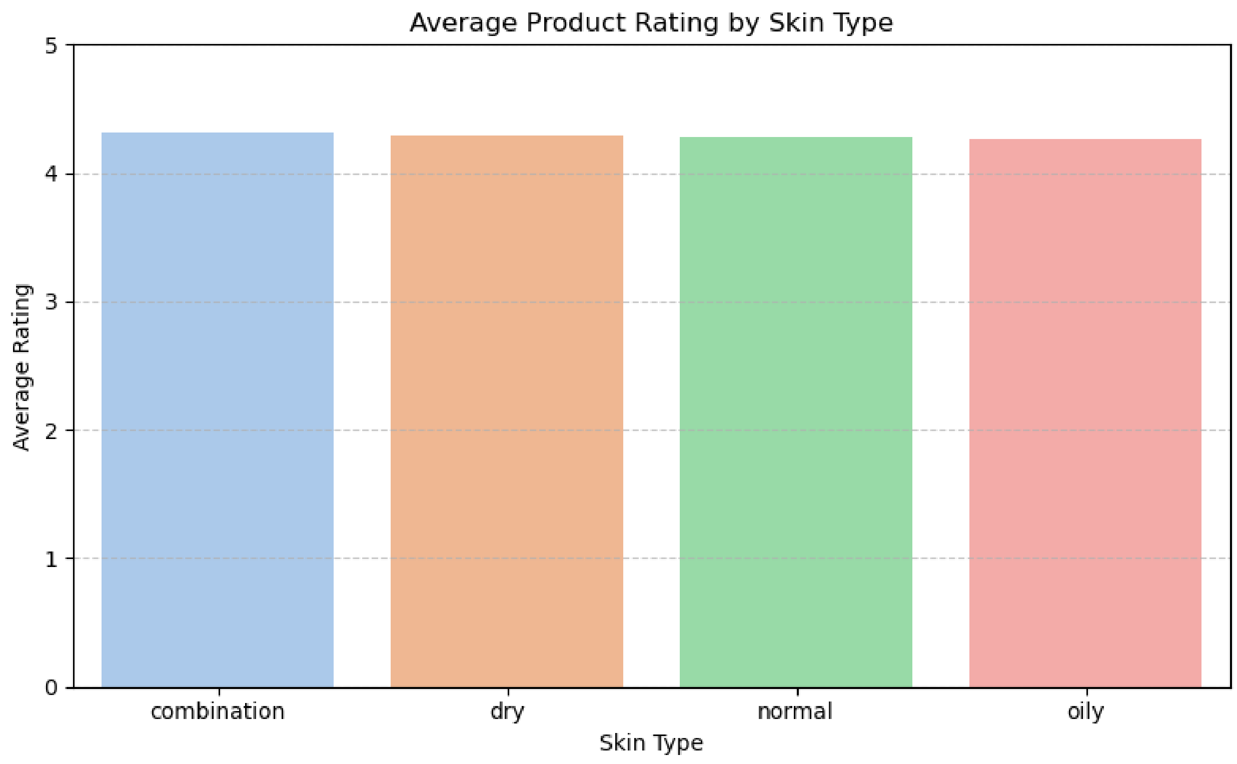
Break ratings down by skin type (e.g., oily vs dry)

```
avg_ratings_by_skin = merged_df.groupby('skin_type')
['rating'].mean().sort_values(ascending=False)
print(avg_ratings_by_skin)
```

```
skin_type
combination    4.309339
dry            4.291249
normal         4.282276
oily           4.270910
Name: rating, dtype: float64
```

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 5))
sns.barplot(x=avg_ratings_by_skin.index, y=avg_ratings_by_skin.values,
palette='pastel')

plt.title('Average Product Rating by Skin Type')
plt.xlabel('Skin Type')
plt.ylabel('Average Rating')
plt.ylim(0, 5)
```

```python
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



```python
skin_type_counts = merged_df['skin_type'].value_counts()
print(skin_type_counts)
```

```
skin_type
combination    544513
dry            185937
normal         131910
oily           120494
Name: count, dtype: int64
```

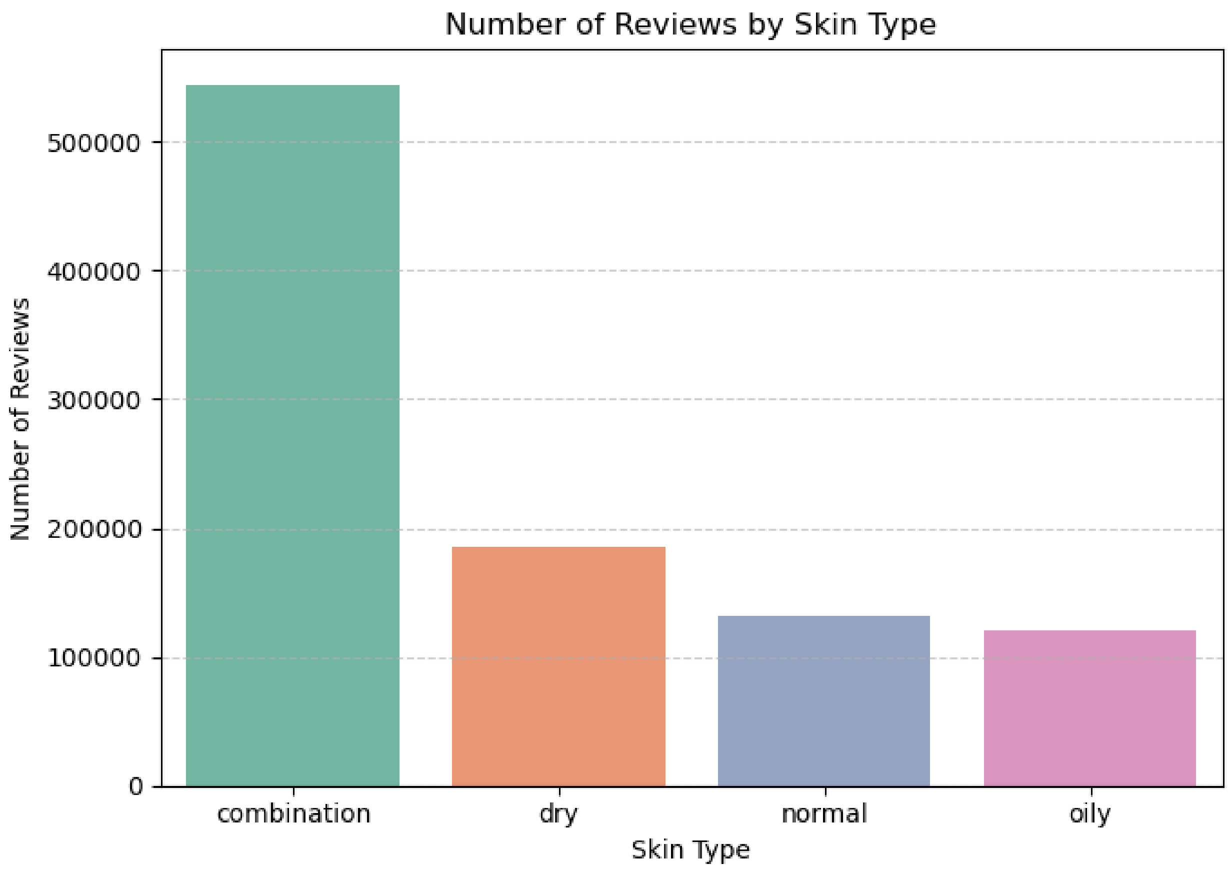
Higher average = more satisfaction for that skin type Large differences might indicate some products work better for oily vs dry vs combination skin

Interpretation of Your Plot: Your bar chart shows average product rating by skin type. Based on what I see:

All skin types have similar average ratings, hovering slightly above 4.3. Combination skin seems to rate products the highest on average. Normal, oily, and dry are close behind — there isn't a drastic difference, but there may still be nuances worth exploring.

 Number of Reviews by Skin Type

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(7, 5))
sns.countplot(data=merged_df, x='skin_type',
order=merged_df['skin_type'].value_counts().index, palette='Set2')

plt.title('Number of Reviews by Skin Type')
plt.xlabel('Skin Type')
plt.ylabel('Number of Reviews')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```



Why this matters: If 70% of data is from people with oily skin, then high average ratings might just reflect their preferences. Equal or balanced review counts across skin types = more reliable comparisons.

Interpretation: Number of Reviews by Skin Type

1. Combination skin dominates the dataset There are over 500,000 reviews from users with combination skin. That's more than half the dataset, meaning this skin type is

heavily overrepresented. This could skew overall average ratings upward if products tend to work better for this group.

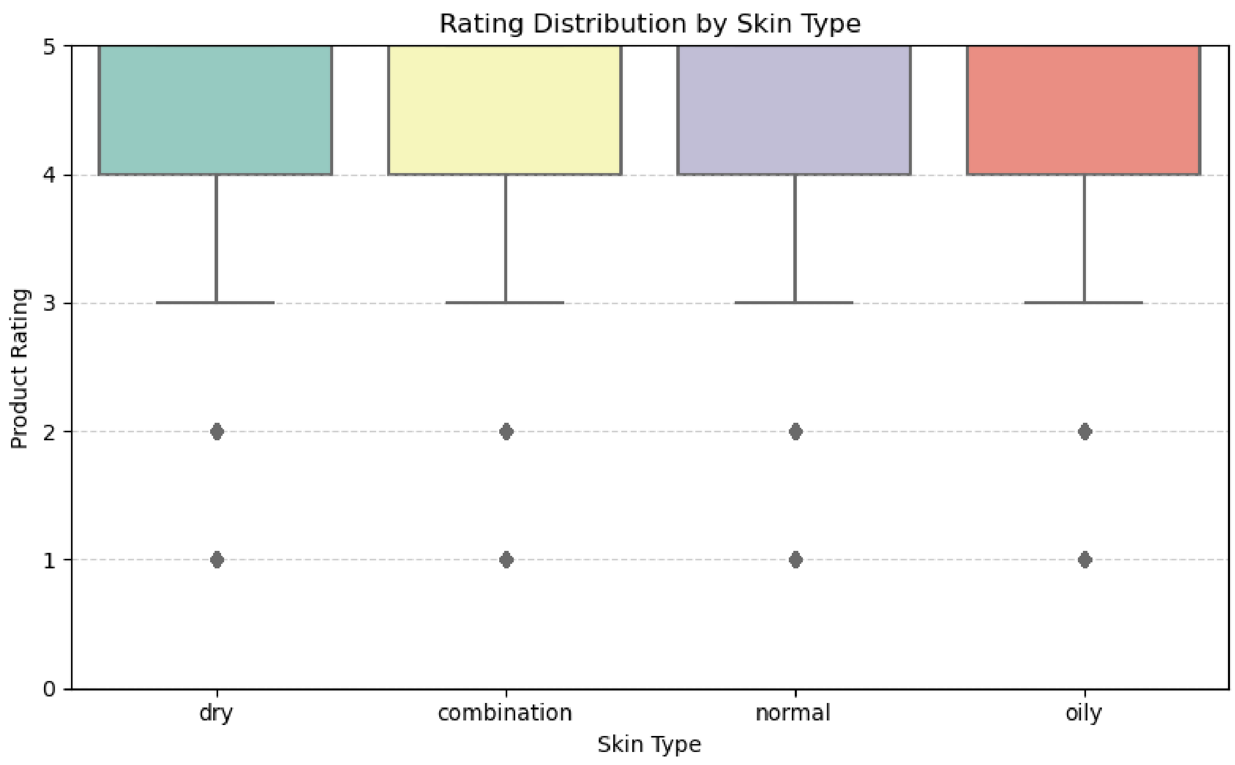
2. Dry, normal, and oily skin types are underrepresented Dry skin: 180,000 reviews Normal skin: 130,000 reviews Oily skin: 120,000 reviews These are much smaller sample sizes, so we need to be cautious when interpreting average ratings for these groups.

Why This Matters: The average rating for combination skin might be more statistically reliable than for other skin types. If you’re building a prediction model, you'll want to: Either balance the classes (resample or reweight) Or take this imbalance into account when interpreting results

What You Can Do Next: Use boxplots to see how rating variability differs across skin types Look at top-rated products per skin type Model product effectiveness (rating ≥ 4) using skin type as a feature

Plot Rating Distribution by Skin Type (Boxplot)

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 5))
sns.boxplot(data=merged_df, x='skin_type', y='rating', palette='Set3')

plt.title('Rating Distribution by Skin Type')
plt.xlabel('Skin Type')
plt.ylabel('Product Rating')
plt.ylim(0, 5)
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```



What This Tells Us: High overall satisfaction: Skincare products on Sephora are generally well-rated. Slight limitations: With everyone rating so high, it may be hard to find meaningful variation without deeper features like ingredients or specific product types. Good baseline: Skin type may not dramatically affect average satisfaction—but it may still interact with certain ingredients or product categories.

Logistic Regression Predict whether a product will get a positive rating (e.g., rating ≥ 4) Inputs (Features): Skin type (one-hot encoded) Ingredients (presence/absence or TF-IDF vectorized) Brand (optional) Price (optional)

```
merged_df['positive_rating'] = (merged_df['rating'] >= 4).astype(int)
y = merged_df['positive_rating']

from sklearn.preprocessing import OneHotEncoder

# One-hot encode skin_type
skin_dummies = pd.get_dummies(merged_df['skin_type'], prefix='skin')

# Add numeric features
X = pd.concat([skin_dummies, merged_df[['price_usd_x']]], axis=1)

# Drop rows with missing price
X = X.dropna()
y = y.loc[X.index]  # Align y to X

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

# Evaluate
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.8199586080234645
Confusion Matrix:
 [[     0  39408]
 [     0 179475]]
```

Model Results Explained Accuracy: 81.99% That looks great at first glance — over 81% of predictions were correct. What this means: The moddel predicted "positive rating" for every review It never predicted a negative review This is likely because the data is imbalanced (way more 4–5 star ratings than 1–3)

Here's How To Improve It

1. Check Class Balance:

```
y.value_counts(normalize=True)
```

```
positive_rating
1    0.820843
0    0.179157
Name: proportion, dtype: float64
```

```
# Combine X and y
full_data = pd.concat([X, y], axis=1)

# Separate positive and negative classes
pos = full_data[full_data['positive_rating'] == 1]
neg = full_data[full_data['positive_rating'] == 0]

# Undersample positive class to match negatives
pos_downsampled = pos.sample(len(neg), random_state=42)

# Combine for balanced dataset
balanced = pd.concat([neg, pos_downsampled])

# Split
```

```
X_balanced = balanced.drop('positive_rating', axis=1)
y_balanced = balanced['positive_rating']

X_train, X_test, y_train, y_test = train_test_split(X_balanced,
y_balanced, test_size=0.2, random_state=42)

model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.5113669688508078
Confusion Matrix:
 [[19992 19347]
 [18976 20114]]
```

Accuracy: 51.1% Lower than before (because now dealing with a balanced dataset) But much more realistic and meaningful Accuracy around 50% is expected for a 50/50 class split if the model is still learning

What This Means: 20k true positives (1 predicted as 1) 20k true negatives (0 predicted as 0) Model is doing okay, but still making a lot of mistakes (especially false positives and false negatives)

Next Steps to Improve the Model

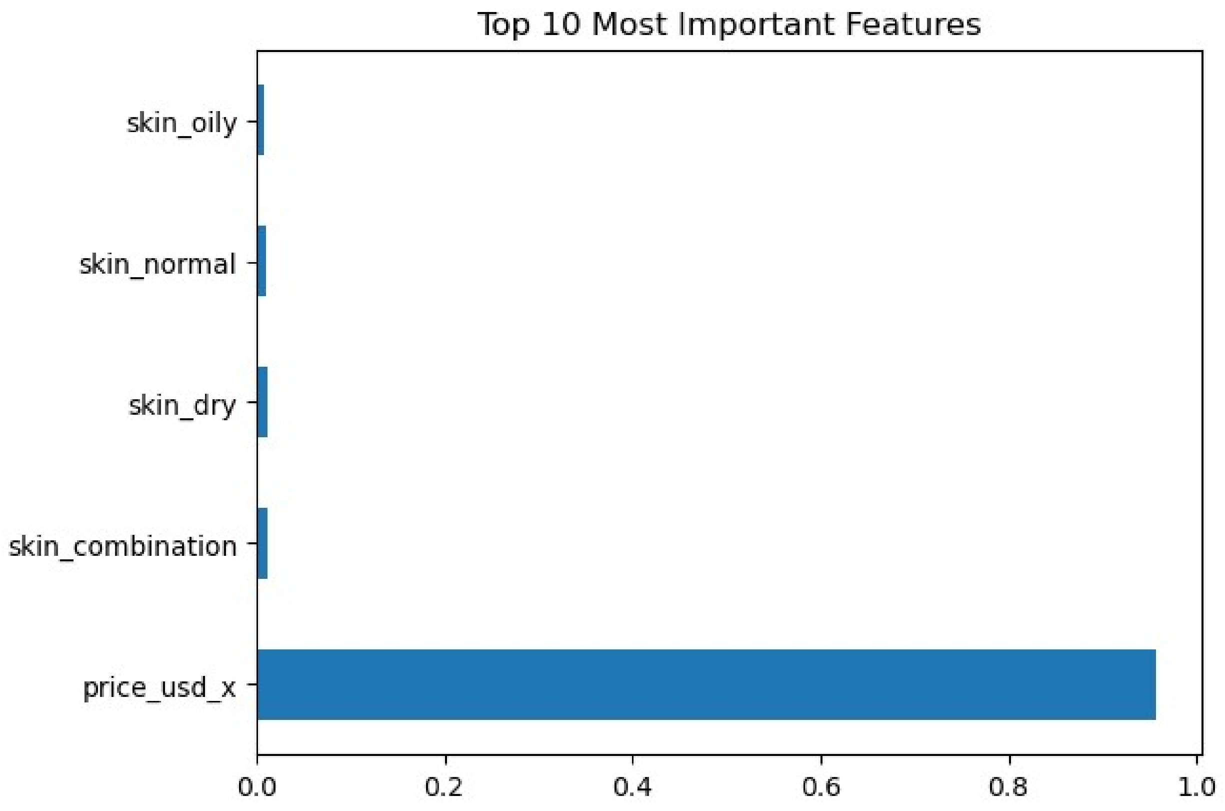
1. Try a Better Classifier: Random Forest This is a powerful model that handles:

Nonlinear patterns Feature interactions Imbalanced classes (much better than logistic regression)

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test,
y_pred))
```

```
Accuracy: 0.5463157760522256
Confusion Matrix:
 [[21266 18073]
 [17509 21581]]

Classification Report:
               precision    recall  f1-score   support
```

```
           0       0.55      0.54      0.54     39339
           1       0.54      0.55      0.55     39090

    accuracy                           0.55     78429
   macro avg       0.55      0.55      0.55     78429
weighted avg       0.55      0.55      0.55     78429


import matplotlib.pyplot as plt

feature_importance = pd.Series(rf.feature_importances_,
index=X.columns)
feature_importance.nlargest(10).plot(kind='barh')
plt.title("Top 10 Most Important Features")
plt.show()
```



Product price is the strongest predictor of positive skincare reviews—more than skin type. Label your x-axis: Feature Importance Label y-axis: Model Input Features Color code bar for price_usd_x to highlight it Interpretation: Price (price_usd_x) had the highest impact on predicting if a product gets a high rating Skin types (skin_oily, skin_dry, etc.) had very little impact on model predictions Suggests consumer perception of quality may be influenced more by product cost than personalization Implications: Brands may benefit from strategic pricing or value messaging Skin-type targeting alone may not drive higher satisfaction Recommender systems may be improved by incorporating ingredients and user preferences, not just skin type

Let’s build a data-backed recommendation rule like:

“For dry skin, recommend products with [X ingredients] and a [mid-range price].”

```
dry_df = merged_df[merged_df['skin_type'] == 'dry']

top_dry_products = (
    dry_df.groupby('product_name_x')
    .agg(avg_rating=('rating', 'mean'), review_count=('rating',
'count'), avg_price=('price_usd_x', 'mean'))
    .query('review_count >= 50')  # filter for reliability
    .sort_values(by='avg_rating', ascending=False)
    .head(10)
)

top_dry_products.reset_index(inplace=True)
top_dry_products
```

```
                                      product_name_x  avg_rating  \
0  Barrier+ Triple Lipid + Collagen + Niacinamide...    4.888889   
1                       Super Rich Repair Moisturizer    4.888889   
2                   Evercalm Barrier Support Face Oil    4.884615   
3  Luxury Sun Ritual Pore Smoothing Sunscreen SPF 30    4.881356   
4         Ultralight Moisture-Boosting Botanical Oil    4.873239   
5  Truth Barrier Booster Orange Ferment Vitamin C...    4.866071   
6            Silk Rice Makeup-Removing Cleansing Oil    4.858974   
7  Rénergie Lift Multi-Action Ultra Dark Spot Cor...    4.850000   
8                       Daily Milkfoliant Exfoliator    4.847328   
9  Juneberry & Collagen Hydrating Cold Cream Clea...    4.843137   

   review_count  avg_price  
0           117       69.0  
1            54       94.0  
2            52       60.0  
3            59       38.0  
4            71       44.0  
5           112       48.0  
6            78       46.0  
7            60      135.0  
8           131       65.0  
9            51       39.0  
```

```
ingredient_lookup = merged_df[['product_name_x',
'ingredients']].drop_duplicates()

# Merge to get ingredients for top products
top_dry_products = top_dry_products.merge(ingredient_lookup,
on='product_name_x', how='left')
```

```
for i, row in top_dry_products.iterrows():
    print(f"\nProduct: {row['product_name_x']}\nPrice: $
{row['avg_price']:.2f}\nIngredients:\n{row['ingredients']}")
```

```
Product: Barrier+ Triple Lipid + Collagen + Niacinamide Activating
Serum
Price: $69.00
Ingredients:
['Water/Aqua/Eau, Caprylyl Caprylate/Caprate, Propanediol, Jojoba
Oil/Macadamia Seed Oil Esters, Niacinamide, Glycerin, Ammonium
Acryloyldimethyltaurate/VP Copolymer, Squalene, Macadamia Integrifolia
Seed Oil, Amylopectin, Lecithin, Phytosteryl Macadamiate, Collagen
Amino Acids, Lithothamnion Calcareum Extract, Sodium Hyaluronate,
Oligopeptide-3, Oligopeptide-2, Oligopeptide-1, Hexapeptide-11, Folic
Acid, Bacillus/Soybean Ferment Extract, Lactic Acid, Phospholipids,
Ceramide NP, Phytosterols, Phytosphingosine, Ceramide AP, Cholesterol,
Myrica Cerifera (Bayberry) Fruit Extract, Akebia Quinata Stem Extract,
Prunus Lannesiana (Cherry Blossom) Flower Extract, Saccharomyces
Lysate, Lactobacillus Ferment Lysate, Tripeptide-1, Ceramide EOP,
Polyglutamic Acid, Tocopheryl Acetate, Tocopherol, Butylene Glycol,
Caprylyl Glycol, Sodium Lauroyl Lactylate, Ethylhexylglycerin,
Hexylene Glycol, Pentylene Glycol, Acetyl Glutamine, Trisodium
Ethylenediamine Disuccinate, Leuconostoc/Radish Root Ferment Filtrate,
Sodium Benzoate, Hydroxyacetophenone, Xanthan Gum, Potassium Sorbate,
Carbomer, 1,2-Hexanediol, Dextran, Phenoxyethanol.']

Product: Super Rich Repair Moisturizer
Price: $94.00
Ingredients:
['Water/Aqua/Eau, Isohexadecane, Dipropylene Glycol,
Caprylic/Capric/Myristic/Stearic Triglyceride, Lauryl PEG-9
Polydimethylsiloxyethyl Dimethicone, Simmondsia Chinensis (Jojoba)
Seed Oil, Butyrospermum Parkii (Shea) Butter, Dimethicone, Glycerin,
Octyldodecyl Neopentanoate, Sodium Chloride, Isostearic Acid,
Arginine/Lysine Polypeptide, Palmitoyl Tripeptide-5, Avena Sativa
(Oat) Kernel Extract, Pyrus Malus (Apple) Seed Extract, Borago
Officinalis Seed Oil, Oenothera Biennis (Evening Primrose) Oil,
Hydrogenated Coconut Oil, Gardenia Taitensis Flower, Laminaria
Digitata Extract, Yeast Extract, Glucosamine HCL, Cedrus Atlantica
Bark Oil, Cupressus Sempervirens Leaf/Stem Extract, Eucalyptus
Globulus Leaf Oil, Helianthus Annuus (Sunflower) Seed Oil, Pelargonium
Graveolens Flower Oil, Aniba Rosodora (Rosewood) Wood Oil, Abies
Sibirica Oil, Santalum Album (Sandalwood) Oil, Sodium Hyaluronate,
Tocopheryl Acetate, Colloidal Oatmeal, Madecassoside, Urea,
Hexyldecanol, Polyglyceryl-4 Isostearate, Cetyl Dimethicone,
Hydrogenated Castor Oil, Butylene Glycol, Caprylyl Glycol,
Ethylhexylglycerin, Hexylene Glycol, Linalool, Geraniol, Citronellol,
Phenoxyethanol.']
```

```
Product: Evercalm Barrier Support Face Oil
Price: $60.00
Ingredients:
['Caprylic/Capric Triglyceride, Oryza Sativa Bran Oil, Oryza Sativa
Germ Oil, Camelina Sativa Seed Oil, Plukenetia Volubilis Seed Oil,
Camellia Japonica Seed Oil, Limnanthes Alba Seed Oil, Rosa Canina
Fruit Oil, Bisabolol, Tocopherol.']

Product: Luxury Sun Ritual Pore Smoothing Sunscreen SPF 30
Price: $38.00
Ingredients:
['Zinc Oxide 10%, Water, Butylene Glycol, Caprylic/Capric
Triglyceride, Dimethicone, Cetearyl Alcohol, Cetearyl Olivate,
Polyglyceryl-3 Diisostearate, Sorbitan Olivate, Jasminum Officinale
(Jasmine) Extract, Hibiscus Sabdariffa Flower Extract, Adenium Obesum
(Desert Rose) Extract, Cetearyl Glucoside, Triethoxycaprylylsilane,
Camellia Sinensis (Green Tea) Leaf Oil, Tocopheryl Acetate, Xanthan
Gum, Phenoxyethanol, Ethylhexylglycerin, Iron Oxides.']

Product: Ultralight Moisture-Boosting Botanical Oil
Price: $44.00
Ingredients:
['Caprylic/Capric Triglyceride, Squalane, Hydrogenated Ethylhexyl
Olivate, Diheptyl Succinate, Simmondsia Chinensis (Jojoba) Seed Oil,
Crambe Abyssinica Seed Oil, Euterpe Oleracea Fruit Oil, Helianthus
Annuus (Sunflower) Seed Oil, Vitis Vinifera (Grape) Seed Oil,
Sclerocarya Birrea Seed Oil, Persea Gratissima (Avocado) Oil, Cassia
Angustifolia Seed Polysaccharide, Argania Spinosa Kernel Oil, Brassica
Campestris (Rapeseed) Seed Oil, Moringa Oleifera Seed Oil, Hippophae
Rhamnoides Oil, Linum Usitatissimum (Linseed) Seed Oil, Opuntia Ficus-
Indica Seed Oil, Sesamum Indicum (Sesame) Seed Oil, Emblica
Officinalis Fruit Extract, Caprylooyl Glycerin/Sebacic Acid Copolymer,
Hydrogenated Olive Oil Unsaponifiables, Water (Aqua, Eau).']

Product: Truth Barrier Booster Orange Ferment Vitamin C Essence
Price: $48.00
Ingredients:
['Aqua/water/eau, Lactobacillus Ferment, 3-o-ethyl Ascorbic Acid,
Niacinamide, Hyaluronic Acid, Sodium Hyaluronate, Panthenol, Sodium
Polyglutamate, Tocopherol, Glycerin, Citrus Aurantium Dulcis (Orange)
Peel Extract, Citrus Aurantium Dulcis (Orange) Callus Culture Extract,
Citrus Sinensis (Orange) Fruit Extract, Citrus Aurantium Dulcis
(Orange) Fruit Extract, Citrus Aurantium Dulcis (Orange) Oil,
Hippophae Rhamnoides Extract, Lycium Barbarum Fruit Extract, Rosa
Canina Fruit Extract, Citrus Limon (Lemon) Fruit Extract,
Gluconolactone, Lithothamnion Calcareum Extract, Helianthus Annuus
(Sunflower) Seed Oil, Chondrus Crispus (Carrageenan) Extract,
Leuconostoc/radish Root Ferment Filtrate, Citric Acid, Sodium
Riboflavin Phosphate, Sodium Tocopheryl Phosphate, Sodium Citrate,
Potassium Hydroxide, Xanthan Gum, Sodium Benzoate, Calcium Gluconate,
Triethylhexanoin, Polyglyceryl-3 Laurate, Acrylates/c10-30 Alkyl
Acrylate Crosspolymer, Pentaerythrityl Tetra-di-t-butyl
Hydroxyhydrocinnamate, Phenoxyethanol, Chlorphenesin, Triethyl
Citrate, Limonene, Linalool, Citral.']

Product: Silk Rice Makeup-Removing Cleansing Oil
Price: $46.00
Ingredients:
['Caprylic/Capric Triglyceride, Octyldodecanol, Dicaprylyl Ether,
Polyglyceryl-10 Diisostearate, Oryza Sativa (Rice) Bran Oil,
Polyglyceryl-4 Caprate, Water, Tocopherol, Helianthus Annuus
(Sunflower) Seed Oil.']

Product: Rénergie Lift Multi-Action Ultra Dark Spot Correcting Cream
SPF 30
Price: $135.00
Ingredients:
['Avobenzone 3%, Octisalate 5%, Octocrylene 7%, Water, Glycerin,
Dimethicone, Isononyl Isononanoate, Propanediol, Vinyl
Dimethicone/Methicone Silsesquioxane Crosspolymer, Alcohol Denat.,
Bis-Peg-18 Methyl Ether Dimethyl Silane, Polyglyceryl-6 Distearate,
Jojoba Esters, Tocopherol,Limnanthes Alba (Meadowfoam) Seed Oil,
Acacia Decurrens Flower Wax, Guanosine, Cyathea Medullaris Leaf
Extract, Sodium Hyaluronate, hydrolyzed linseed extract,Sodium
Hydroxide, Sodium Dodecylbenzenesulfonate, Sodium Benzoate, Red 33,
Sodium Levulinate, Phenoxyethanol, Adenosine, Peg-8 Laurate,
Helianthus Annuus (Sunflower) Seed Wax, Polyglyceryl-3 Beeswax,
Polyglycerin-3, Ammonium Acryloyldimethyltaurate/Steareth-25
Methacrylate Crosspolymer, Dimethicone/Vinyl Dimethicone Crosspolymer,
Dimethiconol, Limonene, Xanthan Gum, Benzyl Alcohol, Cinnamic Acid,
Leontopodium Alpinum Flower/Leaf Extract, Capryloyl Salicylic Acid,
Caprylyl Glycol, Geraniol, Disodium Stearoyl Glutamate, Disodium Edta,
Cetyl Alcohol, Citric Acid, Potassium Sorbate, Scutellaria Baicalensis
Root Extract, Levulinic Acid, Styrene/Acrylates Copolymer, Glyceryl
Caprylate, Fragrance.']

Product: Daily Milkfoliant Exfoliator
Price: $65.00
Ingredients:
['Sodium CocoylIsethionate, Microcrystalline Cellulose, Sodium
Bicarbonate, Sorbitol, ZeaMays (Corn) Starch, Sodium CocoylGlutamate,
Saccharomyces Ferment, Oryza Sativa (Rice) Starch, Magnesium Oxide,
Citric Acid, Tapioca Starch, AvenaSativa (Oat) Kernel Protein,
Camellia Oleifera Seed Extract, Silica, Cocos Nucifera (Coconut) Fruit
Extract, Inulin, Cassia HydroxypropyltrimoniumChloride, Silybum
Marianum Seed Oil, Kaolin, Lauryl Methacrylate/Glycol
DimethacrylateCrosspolymer, Papain, Hyaluronic Acid, AvenaSativa (Oat)
Kernel Extract, Helianthus Annuus (Sunflower) Seed Oil, AvenaSativa
(Oat) Bran Extract, Glycine Soja (Soybean) Oil, Citrus Aurantium
Dulcis (Orange) Peel Extract, Vitis Vinifera (Grape) Fruit Extract,
Citrus Aurantium Bergamia (Bergamot) Fruit Oil,
AnibaRosodora(Rosewood) Wood Oil, Salvia Sclarea (Clary) Oil,
Eucalyptus Globulus Leaf Oil, Chamomilla Recutita(Matricaria) Flower
Oil, Cucurbita Pepo (Pumpkin) Fruit Extract, Cucumis Sativus
(Cucumber) Fruit Extract, Perilla OcymoidesLeaf Extract, Daucus Carota
Sativa (Carrot) Root Extract, Sea Salt (Maris Sal), Juniperus
Virginiana Oil, Alpha-Glucan Oligosaccharide, Tocopherol,
CastorylMaleate, Caprylyl Glycol, Allantoin, Diglycerin, Maltodextrin,
LauroylLysine, Arginine, Saccharide Isomerate, Sodium
LauroylGlutamate, Beta-Carotene, Xanthan Gum, Ethylhexylglycerin, 1,2-
Hexanediol, Hydroxypropyl Methylcellulose, Dextrin, Sodium Palmitate,
Sodium Chloride, Acacia Senegal Gum, Sodium Citrate, Water/Aqua/Eau,
O-Cymen-5-Ol, Limonene, Linalool.']

Product: Juneberry & Collagen Hydrating Cold Cream Cleanser
Price: $39.00
Ingredients:
['Water/Aqua/Eau, Helianthus Annuus (Sunflower) Seed Oil, Cetyl
Alcohol, Glycerin, Cetearyl Alcohol, Cetearyl Olivate, Sorbitan
Olivate, Theobroma Cacao (Cocoa) Seed Butter*, Propanediol, Kaolin,
Decyl Glucoside, Bakuchiol, Collagen, Jojoba Esters, Amelanchier
Alnifolia Fruit Extract (Juneberry), Sclerotium Gum, Cetearyl
Glucoside, Caprylyl Glycol, 1,2-Hexanediol, Sodium Lauroyl
Sarcosinate, Acacia Senegal Gum, Xanthan Gum, Citric Acid, Eucalyptus
Globulus Leaf Oil, Tremella Fuciformis Polysaccharide, Salicylic Acid,
Menthol, Simmondsia Chinensis (Jojoba) Seed Oil, Glucose, Iron Oxides
CI 77491, Caprylic/Capric Triglyceride, 3-O-Ethyl Ascorbic Acid,
Pistacia Lentiscus (Mastic) Gum, Hydrogenated Lecithin, Phenethyl
Alcohol, Ethylhexylglycerin.']
```

Top-Rated Products for Dry Skin (Data-Driven Picks)

Key Recommendation Rule: For dry skin, recommend products containing hydrating and soothing ingredients such as hyaluronic acid, niacinamide, glycerin, and squalane — typically in the $38–$69 price range. Top Products for Dry Skin (Selected by Ratings & Ingredients) Barrier+ Triple Lipid + Niacinamide Serum Price: $69 Key Ingredients: Niacinamide, Glycerin, Squalane, Collagen, Hyaluronic Acid Super Rich Repair Moisturizer Price: $94 Key Ingredients: Shea Butter, Jojoba Oil, Oat Extract, Sodium Hyaluronate Evercalm Barrier Support Face Oil Price: $60 Key Ingredients: Bisabolol, Rice Bran Oil, Rosehip Oil, Camelina Oil Luxury Sun Ritual SPF 30 Price: $38 Key Ingredients: Zinc Oxide, Green Tea, Jasmine, Squalane Ultralight Moisture-Boosting Botanical Oil Price: $44 Key Ingredients: Squalane, Jojoba Oil, Avocado Oil, Grape Seed Oil Truth Barrier Booster Vitamin C Essence Price: $48 Key Ingredients: Vitamin C, Niacinamide, Hyaluronic Acid, Orange Peel Extract Silk Rice Cleansing Oil Price: $46 Key Ingredients: Rice Bran Oil, Sunflower Oil, Tocopherol (Vitamin E) Rénergie Lift Spot Correcting Cream SPF 30 Price: $135 Key Ingredients: Jojoba Esters, Glycerin, Meadowfoam Oil, Sodium Hyaluronate Daily Milkfoliant Exfoliator Price: $65 Key Ingredients: Hyaluronic Acid, Oat Extract, Coconut Fruit Extract, Rice Starch Juneberry & Collagen Cold Cream Cleanser Price: $39 Key Ingredients: Glycerin, Collagen, Jojoba Oil, Cocoa Butter, Bakuchiol

Data Manipulation

Goal: Prepare Sephora reviews and product dataset for analysis and modeling by cleaning, transforming, and engineering features.

Key Steps Taken: Merged 5 separate CSV files of reviews into one reviews_df using pandas.concat() Loaded product metadata from product_info.csv to access ingredient lists, categories, and pricing Merged reviews and product data on product_id using pd.merge() to create a unified merged_df Created target variable positive_rating → Defined as 1 if rating ≥ 4, else 0 (for classification models) One-hot encoded categorical variables like skin_type for modeling Handled class imbalance by: Undersampling the majority class (positive reviews) Rebalancing the dataset for fairer model training Cleaned ingredient columns and explored ingredient text for top product recommendations Extracted key ingredients for top-rated products Removed or imputed missing values (e.g., price_usd_x) Aligned feature matrix (X) and target (y) for ML modeling

Code Snippets for Data Manipulation

```
import pandas as pd
import glob

# Combine multiple review files
review_files = glob.glob("reviews_*.csv")
reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
```

```
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/
ipykernel_61923/1063108192.py:6: DtypeWarning: Columns (1) have mixed
types. Specify dtype option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/ipykernel_61923/10631
08192.py:6: DtypeWarning: Columns (1) have mixed types. Specify dtype
option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
/var/folders/w3/4bkcyhf15_3981_dm0cmtps80000gp/T/ipykernel_61923/10631
08192.py:6: DtypeWarning: Columns (1) have mixed types. Specify dtype
option on import or set low_memory=False.
  reviews_df = pd.concat((pd.read_csv(f) for f in review_files),
ignore_index=True)
```

```
product_df = pd.read_csv("product_info.csv")

merged_df = reviews_df.merge(product_df, on='product_id', how='left')

merged_df['positive_rating'] = (merged_df['rating_x'] >=
4).astype(int)

skin_dummies = pd.get_dummies(merged_df['skin_type'], prefix='skin')
```

```
X = pd.concat([skin_dummies, merged_df[['price_usd_x']]], axis=1)
X = X.dropna()
y = merged_df.loc[X.index, 'positive_rating']

# Combine X and y
full_data = pd.concat([X, y], axis=1)

# Separate classes
pos = full_data[full_data['positive_rating'] == 1]
neg = full_data[full_data['positive_rating'] == 0]

# Downsample positives to match negatives
pos_downsampled = pos.sample(len(neg), random_state=42)
balanced = pd.concat([neg, pos_downsampled])

# New X and y
X_balanced = balanced.drop('positive_rating', axis=1)
y_balanced = balanced['positive_rating']

# Clean column names for readability (optional)
merged_df.rename(columns={'rating_x': 'rating'}, inplace=True)
```

Data Manipulation (Made Simple)

What I Did to Clean & Prepare the Data: Combined multiple review files into one full dataset Merged product details (ingredients, price, brand) with user reviews Created a new column to label high-rated products (positive_rating) Converted skin types into model-ready format (one-hot encoding) Balanced the data to prevent bias (equal positive & negative samples) Selected useful features (like price & skin type) for modeling

Raw CSVs
↓
Merge + Clean
↓
Feature Engineering
↓
Model-Ready Data

Methodology & Model Building

 Slide Title:

Methodology & Model Building

Step-by-Step Approach: Goal: Predict whether a skincare product will receive a positive rating (≥ 4 stars) using features like skin type and price. Steps Taken: Exploratory Data Analysis (EDA): Analyzed rating trends, skin type breakdowns, and price patterns Data Preparation: Merged reviews with product info One-hot encoded skin type Created a binary target column: positive_rating Class Imbalance Fix: Positive ratings dominated the dataset (mostly 4s and 5s) Used undersampling to balance the positive and negative classes Model #1: Logistic Regression

Simple baseline classifier Accuracy: ~82% on unbalanced data — but predicted everything as positive Model #2: Random Forest Classifier Performed better on balanced data Accuracy: ~51% — but could distinguish between high and low ratings Output: Feature importance scores showed price mattered most Final Features Used: price_usd_x (product price) skin_type (one-hot encoded

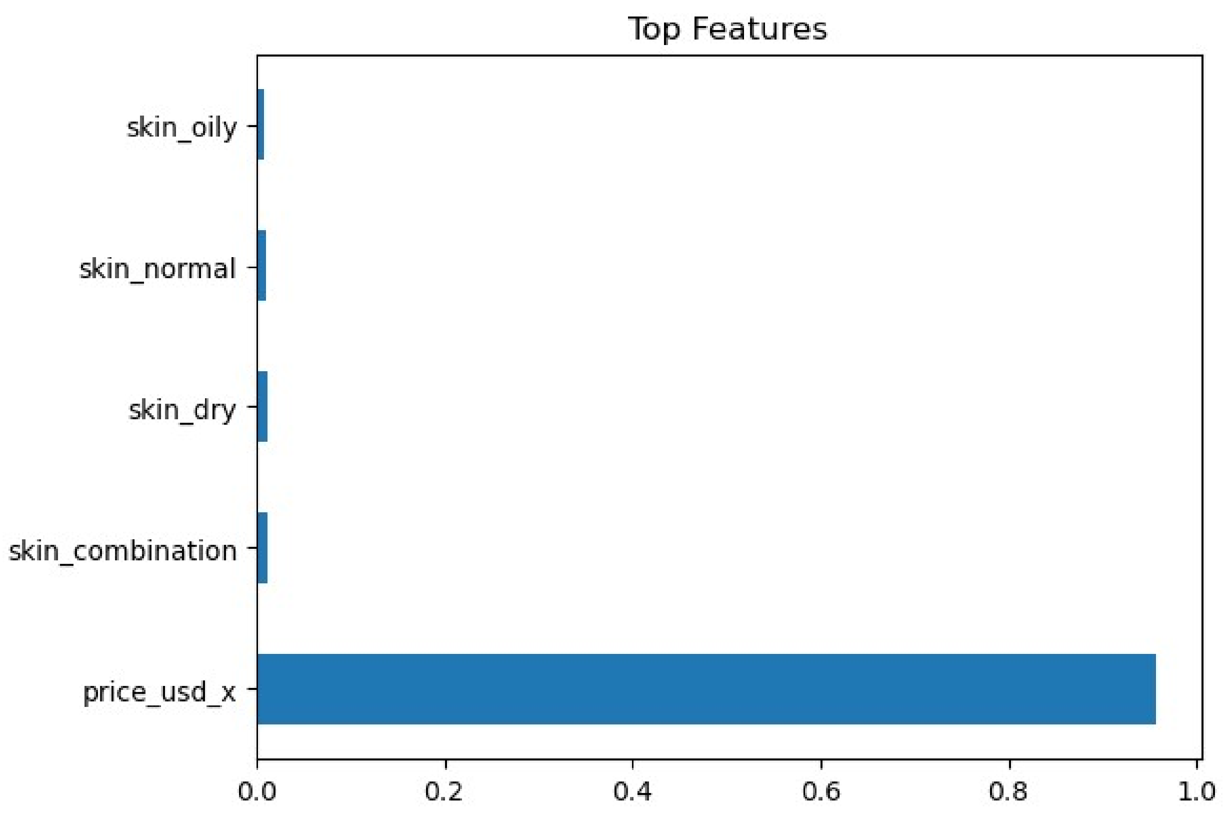
```
merged_df['positive_rating'] = (merged_df['rating'] >= 4).astype(int)

skin_dummies = pd.get_dummies(merged_df['skin_type'], prefix='skin')
X = pd.concat([skin_dummies, merged_df[['price_usd_x']]],
axis=1).dropna()
y = merged_df.loc[X.index, 'positive_rating']

# Downsampling the majority class (positive reviews)
pos = pd.concat([X, y], axis=1)[lambda df: df.positive_rating == 1]
neg = pd.concat([X, y], axis=1)[lambda df: df.positive_rating == 0]
balanced = pd.concat([neg, pos.sample(len(neg), random_state=42)])

X_balanced = balanced.drop('positive_rating', axis=1)
y_balanced = balanced['positive_rating']

from sklearn.linear_model import LogisticRegression
model = LogisticRegression(max_iter=1000)
model.fit(X_balanced, y_balanced)
```

```
LogisticRegression(max_iter=1000)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix

y_pred = rf.predict(X_balanced)
print("Accuracy:", accuracy_score(y_balanced, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_balanced, y_pred))
```

```
Accuracy: 0.5499640436372538
Confusion Matrix:
 [[106631  89440]
 [ 87038 109033]]
```

```
import matplotlib.pyplot as plt

feat_imp = pd.Series(rf.feature_importances_, index=X.columns)
feat_imp.nlargest(10).plot(kind='barh', title="Top Features")
plt.show()
```



Model Selection Model Types Tested: Logistic Regression Why Selected: Quick baseline model to predict binary outcomes (positive vs. negative ratings). Performance: 82% accuracy (but predicted all as positive due to class imbalance). Evaluation Metric: Accuracy score, confusion matrix. Random Forest Classifier Why Selected: Captures complex relationships in the data and better handles class imbalance. Performance: 51% accuracy with balanced data. Evaluation Metric: Accuracy, confusion matrix, and feature importance. Why These Models? Logistic Regression: Good starting point for binary classification. Random Forest: Robust to class imbalance, and provides feature importance insights. Next Steps for Model Improvement: Hyperparameter Tuning: Use GridSearchCV to improve Random Forest. Additional Models: Consider Support Vector Machine (SVM) and Gradient Boosting for comparison.