Data Preprocessing

```
# Impo
import
import
import
import
```

Run this cell to mount your Google Drive.
Learn more

Dismiss

```
# Load the dataset
file_path = "/content/reviews_0-250.csv"
df = pd.read_csv(file_path, on_bad_lines='warn')
```

```
<ipython-input-1-87f83597872c>:9: DtypeWarning: Columns (1) have mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv(file_path, on_bad_lines='warn')
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
# Preview the data
df.head()
```

	Unnamed: 0	author_id	rating	is_recommended	helpfulness	total_feedback_count	total_neg_feedback_count	total_pos_feedback_count
0	0	1741503524	5	1.0	1.0	2	0	2
1	1	[illegible]	[illegible]	0.0	NaN	0	0	0
2	2	5061282401	5	1.0	NaN	0	0	0
3	3	6083038851	5	1.0	NaN	0	0	0
4	4	47056667835	5	1.0	NaN	0	0	0

Run this cell to mount your Google Drive.
Learn more

```
df.info()
df.describe(include='all')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 327015 entries, 0 to 327014
Data columns (total 19 columns):
 #   Column                     Non-Null Count   Dtype
---  ------                     --------------   -----
 0   Unnamed: 0                 327015 non-null  int64
 1   author_id                  327015 non-null  object
 2   rating                     327015 non-null  int64
 3   is_recommended             256887 non-null  float64
 4   helpfulness                142866 non-null  float64
 5   total_feedback_count       327015 non-null  int64
 6   total_neg_feedback_count   327015 non-null  int64
 7   total_pos_feedback_count   327015 non-null  int64
 8   submission_time            327015 non-null  object
 9   review_text                326415 non-null  object
 10  review_title               237160 non-null  object
 11  skin_tone                  266015 non-null  object
 12  eye_color                  245743 non-null  object
 13  skin_type                  285501 non-null  object
 14  hair_color                 245090 non-null  object
 15  product_id                 327014 non-null  object
 16  product_name               327014 non-null  object
 17  brand_name                 327014 non-null  object
 18  price_usd                  327014 non-null  float64
dtypes: float64(3), int64(5), object(11)
memory usage: 47.4+ MB
```

Run this cell to mount your Google Drive.
Learn more

	Unnamed: 0	author_id	rating	is_recommended	helpfulness	total_feedback_count	total_neg_feedback_count	total_po
count	327015.000000	327015	327015.000000	256887.000000	142866.000000	327015.000000	327015.000000	
unique	NaN	230798	NaN	NaN	NaN	NaN	NaN	
top	NaN	1288462295	NaN	NaN	NaN	NaN	NaN	
freq	NaN	32	NaN	NaN	NaN	NaN	NaN	
mean	163507.000000	NaN	4.333076	0.839801	0.779456	3.521438	0.734183	
std	94401.243477	NaN	1.141202	0.366791	0.313842	29.106834	5.459710	
min	0.000000	NaN	1.000000	0.000000	0.000000	0.000000	0.000000	
25%	81753.500000	NaN	4.000000	1.000000	0.666667	0.000000	0.000000	
50%	163507.000000	NaN	5.000000	1.000000	1.000000	0.000000	0.000000	
75%	245260.500000	NaN	5.000000	1.000000	1.000000	3.000000	0.000000	
max	327014.000000	NaN	5.000000	1.000000	1.000000	5464.000000	573.000000	

```
# Check total missing values in each column
missing_values = df.isnull().sum()
missing_percent = (missing_values / len(df)) * 100

# Display columns with missing values
missing_data = pd.DataFrame({
    'Missing Values': missing_values,
    'Percentage': missing_percent
}).sort_values(by='Percentage', ascending=False)

print("Missing Values Summary:\n")
print(missing_data)

# Drop rows with missing values in CRITICAL columns (essential for analysis)
critical_columns = ['rating', 'review_text', 'product_name']
df = df.dropna(subset=critical_columns)

# Fill missing values in LESS critical columns with 'Unknown' or appropriate placeholder
columns_to_fill = ['skin_tone', 'eye_color', 'hair_color', 'skin_type', 'brand_name']
for col in columns_to_fill:
    df[col] = df[col].fillna('Unknown')

# Optionally fill price column with median if you're using price
if df['price_usd'].isnull().sum() > 0:
    df['price_usd'] = df['price_usd'].fillna(df['price_usd'].median())

# Re-check for missing values after handling
print("\nMissing Values After Cleaning:\n")
```

```
print(df.isnull().sum())
```

```
Missing Values Summary:

                                  Values  Percentage
h                                 184149   56.312096
r                                  89855   27.477333
h                                  81925   25.052368
e                                  81272   24.852683
i                                  70128   21.444888
skin_tone                          61000   18.653579
skin_type                          41514   12.694831
review_text                          600    0.183478
product_id                             1    0.000306
brand_name                             1    0.000306
product_name                           1    0.000306
price_usd                              1    0.000306
rating                                 0    0.000000
Unnamed: 0                             0    0.000000
author_id                              0    0.000000
total_pos_feedback_count               0    0.000000
submission_time                        0    0.000000
total_feedback_count                   0    0.000000
total_neg_feedback_count               0    0.000000
<ipython-input-5-32f1df6844d9>:21: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df[col] = df[col].fillna('Unknown')

Missing Values After Cleaning:

Unnamed: 0                      0
author_id                       0
rating                          0
is_recommended              70128
helpfulness                183668
total_feedback_count            0
total_neg_feedback_count        0
total_pos_feedback_count        0
submission_time                 0
review_text                     0
review_title                89255
skin_tone                       0
eye_color                       0
skin_type                       0
hair_color                      0
product_id                      0
product_name                    0
brand_name                      0
price_usd                       0
dtype: int64
```

Run this cell to mount your Google Drive.
Learn more

```
# Remove duplicate rows if any
df = df.drop_duplicates()

# Confirm no duplicates remain
df.duplicated().sum()
```

```
np.int64(0)
```

```
import re

def clean_text(text):
    text = str(text).lower()
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove non-alphabetic characters
    text = re.sub(r'\s+', ' ', text).strip() # Remove extra whitespace
    return text

df['review_text'] = df['review_text'].apply(clean_text)
```

```
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 326415 entries, 0 to 327013
Data columns (total 19 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 [illegible]                   [illegible]415 non-null  int64
 [illegible]                   [illegible]415 non-null  object
 [illegible]                   [illegible]415 non-null  int64
 [illegible]                   [illegible]287 non-null  float64
 [illegible]                   [illegible]747 non-null  float64
 5   total_feedback_count      326415 non-null  int64
 6   total_neg_feedback_count  326415 non-null  int64
 7   total_pos_feedback_count  326415 non-null  int64
 8   submission_time           326415 non-null  object
 9   review_text               326415 non-null  object
 10  review_title              237160 non-null  object
 11  skin_tone                 326415 non-null  object
 12  eye_color                 326415 non-null  object
 13  skin_type                 326415 non-null  object
 14  hair_color                326415 non-null  object
 15  product_id                326415 non-null  object
 16  product_name              326415 non-null  object
 17  brand_name                326415 non-null  object
 18  price_usd                 326415 non-null  float64
dtypes: float64(3), int64(5), object(11)
memory usage: 49.8+ MB
```

Run this cell to mount your Google Drive.
Learn more

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

Exploratory Data Analysis (EDA)

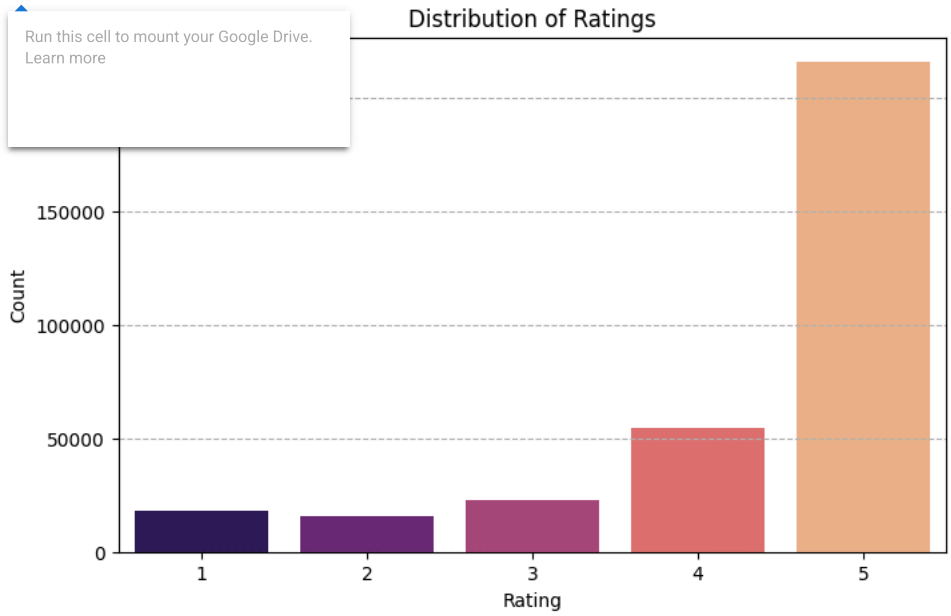
Distribution of Ratings

```
# Plot distribution of ratings
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='rating', palette='magma')
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.grid(axis='y', linestyle='--')
plt.show()

# Average rating overall
print("Average Rating:", round(df['rating'].mean(), 2))
```

```
<ipython-input-9-e054f17d017c>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend
```

```
sns.countplot(data=df, x='rating', palette='magma')
```

Run this cell to mount your Google Drive.
Learn more

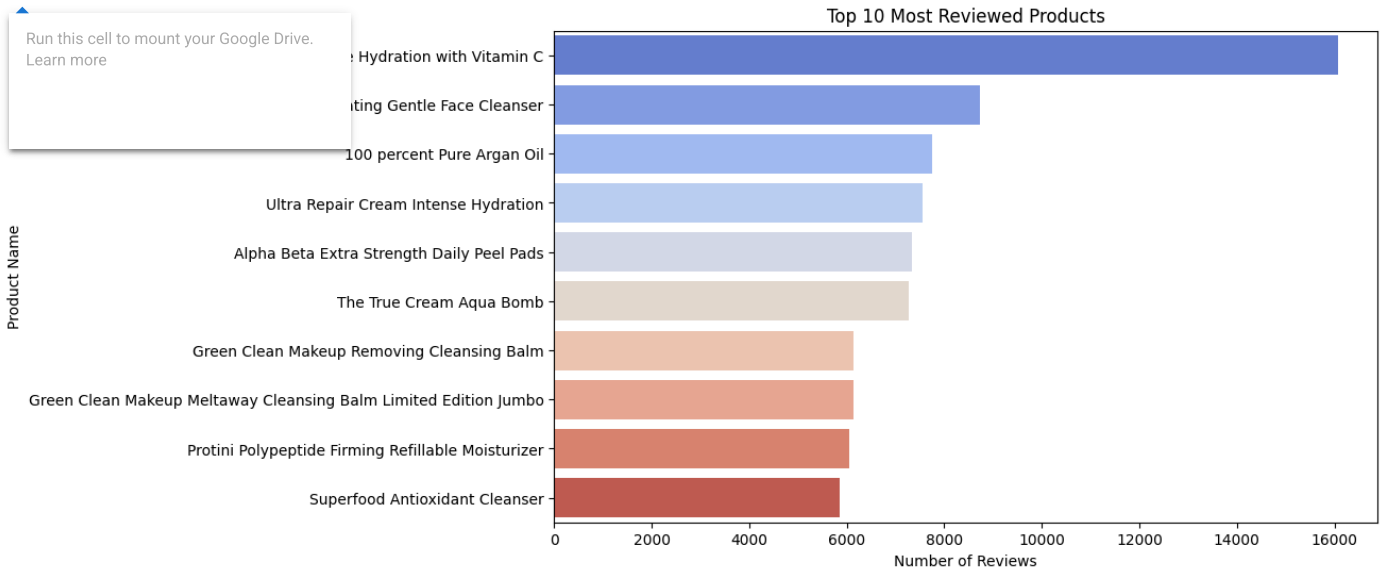


Most Reviewed Products

```
top_products = df['product_name'].value_counts().head(10)

plt.figure(figsize=(10, 6))
sns.barplot(x=top_products.values, y=top_products.index, palette='coolwarm')
plt.title('Top 10 Most Reviewed Products')
plt.xlabel('Number of Reviews')
plt.ylabel('Product Name')
plt.show()
```

```
<ipython-input-10-fcef18c5a3a1>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend

  sns.barplot(x=top_products.values, y=top_products.index, palette='coolwarm')
```

Run this cell to mount your Google Drive.
Learn more

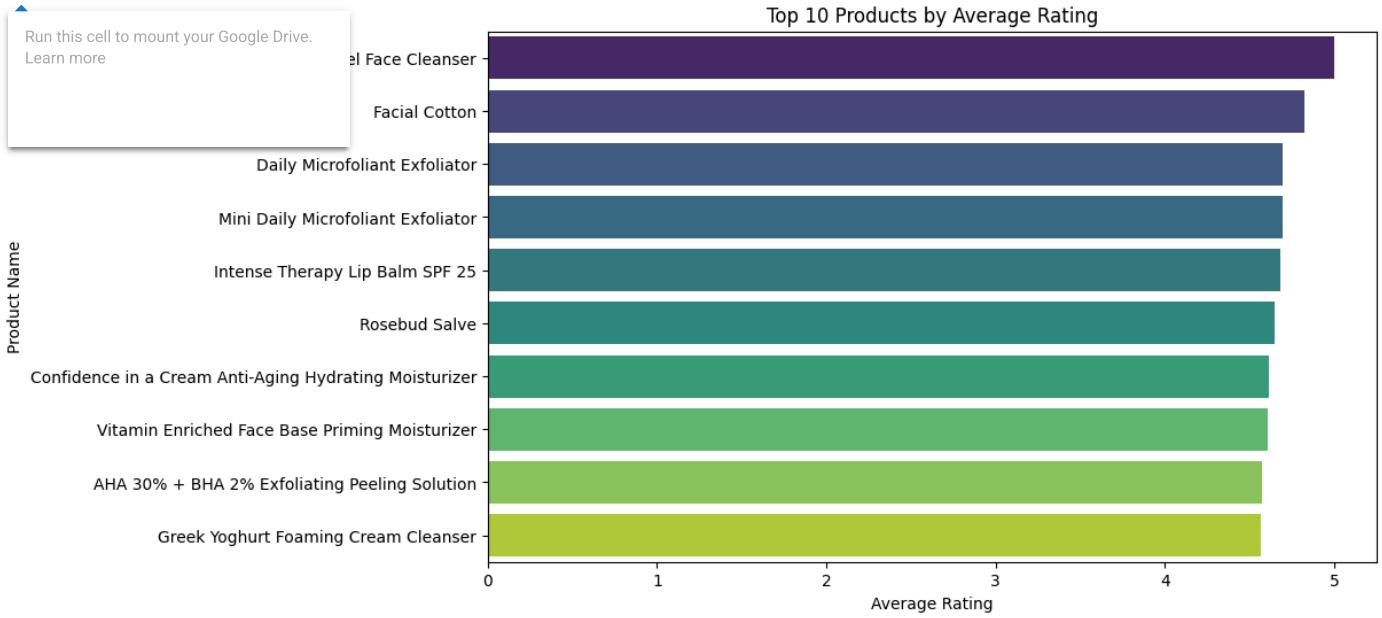


Average Ratings per Product

```
avg_rating_per_product = df.groupby('product_name')['rating'].mean().sort_values(ascending=False).head(10)

plt.figure(figsize=(10, 6))
sns.barplot(x=avg_rating_per_product.values, y=avg_rating_per_product.index, palette='viridis')
plt.title('Top 10 Products by Average Rating')
plt.xlabel('Average Rating')
plt.ylabel('Product Name')
plt.show()
```

```
<ipython-input-11-e724dd8e68d1>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend

  sns.barplot(x=avg_rating_per_product.values, y=avg_rating_per_product.index, palette='viridis')
```

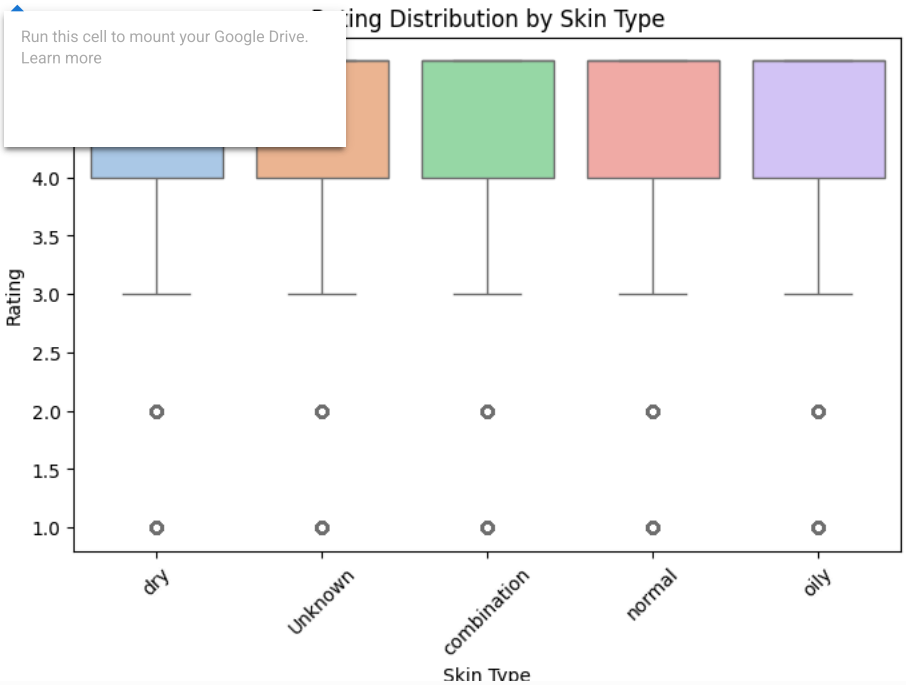
Run this cell to mount your Google Drive.
Learn more



Customer Behavior Patterns: Skin Type & Ratings

```
plt.figure(figsize=(8, 5))
sns.boxplot(data=df, x='skin_type', y='rating', palette='pastel')
plt.title('Rating Distribution by Skin Type')
plt.xlabel('Skin Type')
plt.ylabel('Rating')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-12-763038eb0da1>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.boxplot(data=df, x='skin_type', y='rating', palette='pastel')
```

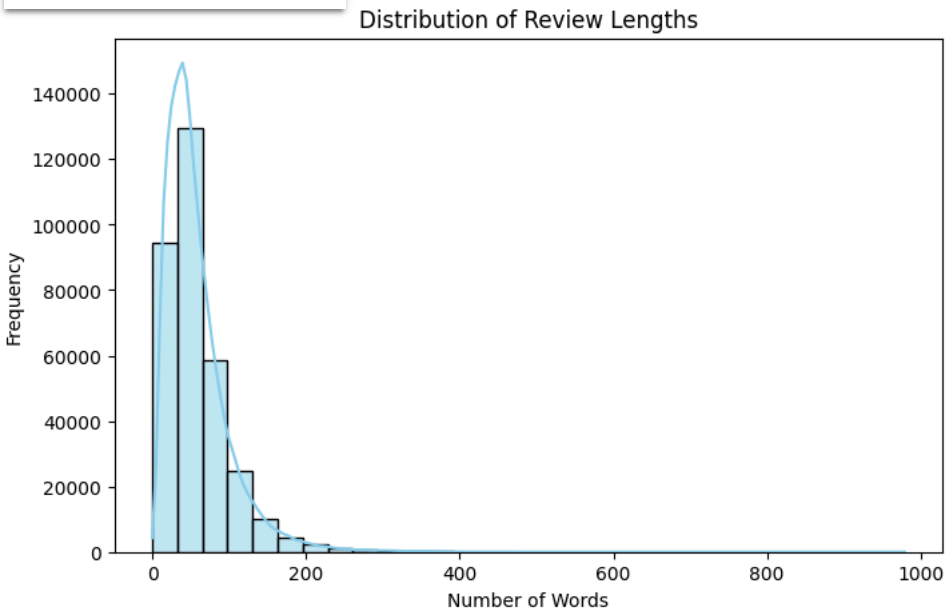


Descriptive Stats for Review Length

```
# Add new column for review text length
df['review_length'] = df['review_text'].apply(lambda x: len(str(x).split()))

# Basic stats
print("\nReview Length Statistics:")
print(df['review_length'].describe())

# Histogram
plt.figure(figsize=(8, 5))
sns.histplot(df['review_length'], bins=30, kde=True, color='skyblue')
plt.title('Distribution of Review Lengths')
plt.xlabel('Number of Words')
plt.ylabel('Frequency')
plt.show()
```

```
Review Length Statistics:
count    326415.000000
mean         58.094910
std          42.598846
min           0.000000
2
5
7
m
N
```

Run this cell to mount your Google Drive.
Learn more

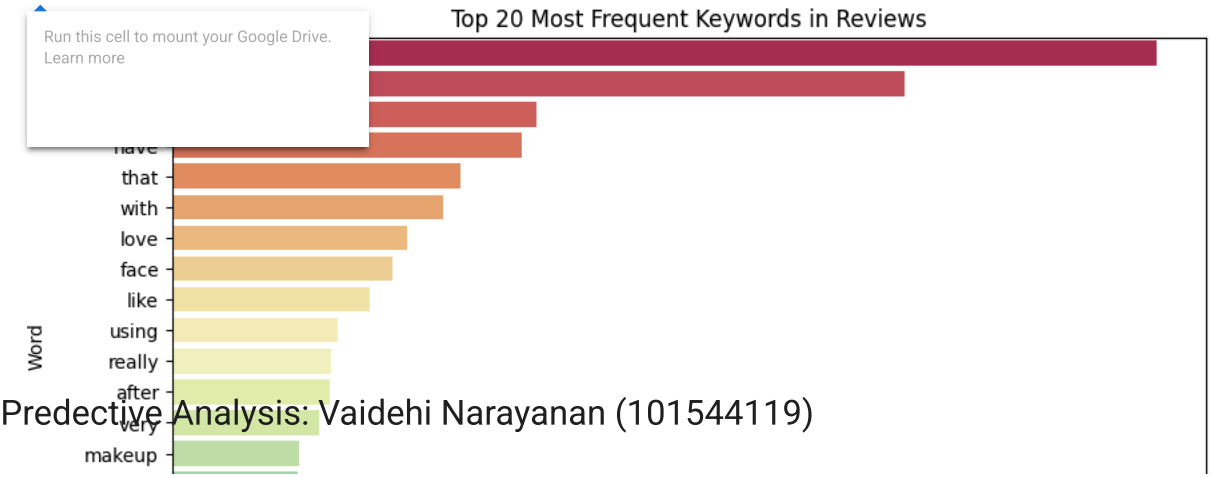


Frequent Keywords in Reviews (Simple Word Count)

```
from collections import Counter
import re

# Combine all reviews
all_text = ' '.join(df['review_text'].dropna().astype(str).tolist())
all_text = re.sub(r'[^a-zA-Z\s]', '', all_text).lower()  # remove punctuation & lowercase
word_list = all_text.split()

# Count most common words (excluding short ones)
word_counts = Counter([word for word in word_list if len(word) > 3])
common_words = word_counts.most_common(20)

# Plot
words, counts = zip(*common_words)
plt.figure(figsize=(10, 6))
sns.barplot(x=list(counts), y=list(words), palette='Spectral')
plt.title('Top 20 Most Frequent Keywords in Reviews')
plt.xlabel('Frequency')
plt.ylabel('Word')
plt.show()
```

```
<ipython-input-14-c95ac45266e8>:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend

  sns.barplot(x=list(counts), y=list(words), palette='Spectral')
```

Run this cell to mount your Google Drive.
Learn more



Predective Analysis: Vaidehi Narayanan (101544119)

```
# Convert date to datetime
df['submission_time'] = pd.to_datetime(df['submission_time'], errors='coerce')

# RFM variables
latest_date = df['submission_time'].max()
rfm = df.groupby('author_id').agg({
    'submission_time': lambda x: (latest_date - x.max()).days,
    'author_id': 'count',
    'price_usd': 'mean'
}).rename(columns={
    'submission_time': 'Recency',
    'author_id': 'Frequency',
```