```
import nltk
import glob
import os
import pandas as pd
import seaborn as sns
from textblob import TextBlob
from collections import Counter
from wordcloud import WordCloud
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

main_path = '/kaggle/input/zara-dataset-men-and-women-clothing/'
```

Define paths for men's and women's clothing CSV files

```
men_csv_files = glob.glob(main_path + 'Men/Men/*.csv')
women_csv_files = glob.glob(main_path + 'Women/Women/*.csv')

# Initialize lists to store dataframes for men and women
men_dataframes = []
women_dataframes = []

# Load data for men's clothing
for csv_file in men_csv_files:
    men_dataframes.append(pd.read_csv(csv_file))

# Load data for women's clothing
for csv_file in women_csv_files:
    women_dataframes.append(pd.read_csv(csv_file))

# Concatenate dataframes for men and women
men_data = pd.concat(men_dataframes)
women_data = pd.concat(women_dataframes)
```

Display basic information about the dataset

```
#Men
num_rows, num_columns = men_data.shape
print("The dataset comprises {} rows and {} columns.".format(num_rows,
num_columns))
```

```
The dataset comprises 1478 rows and 6 columns.
```

```
#Women
num_rows, num_columns = women_data.shape
print("The dataset comprises {} rows and {} columns.".format(num_rows,
num_columns))
```

```
The dataset comprises 2707 rows and 6 columns.
```

```
len(men_data.axes[0])
```

```
1478
```

```
len(women_data.axes[0])
```

```
2707
```

Use Case1: Sentiment Analysis

```
# Perform sentiment analysis on product descriptions

men_data['description_sentiment'] = men_data['details'].apply(lambda
x: TextBlob(str(x)).sentiment.polarity)
women_data['description_sentiment'] =
women_data['Details'].apply(lambda x:
TextBlob(str(x)).sentiment.polarity)
```

```
men_data['description_sentiment']
```

```
0    -0.100000
1    -0.100000
2    -0.100000
3    -0.066667
4    -0.066667
        ...
41    0.000000
42   -0.050000
43    0.000000
44    0.000000
45    0.466667
Name: description_sentiment, Length: 1478, dtype: float64
```

```
women_data['description_sentiment']
```

```
0    -0.083333
1     0.154615
2     0.021429
3    -0.050000
4     0.010440
        ...
62   -0.050000
63   -0.018889
```

```
64    0.136667
65   -0.072222
66   -0.060714
Name: description_sentiment, Length: 2707, dtype: float64
```

Topic Modeling (LDA)

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation

# Replace NaN values in the 'details' column with empty strings
men_data['details'].fillna('', inplace=True)
women_data['Details'].fillna('', inplace=True)

# Vectorize descriptions
vectorizer = CountVectorizer(max_df=0.95, min_df=10,
stop_words='english')
men_descriptions = vectorizer.fit_transform(men_data['details'])
women_descriptions = vectorizer.fit_transform(women_data['Details'])

men_descriptions
```

```
<1478x309 sparse matrix of type '<class 'numpy.int64'>'
     with 20324 stored elements in Compressed Sparse Row format>
```

Fit LDA models

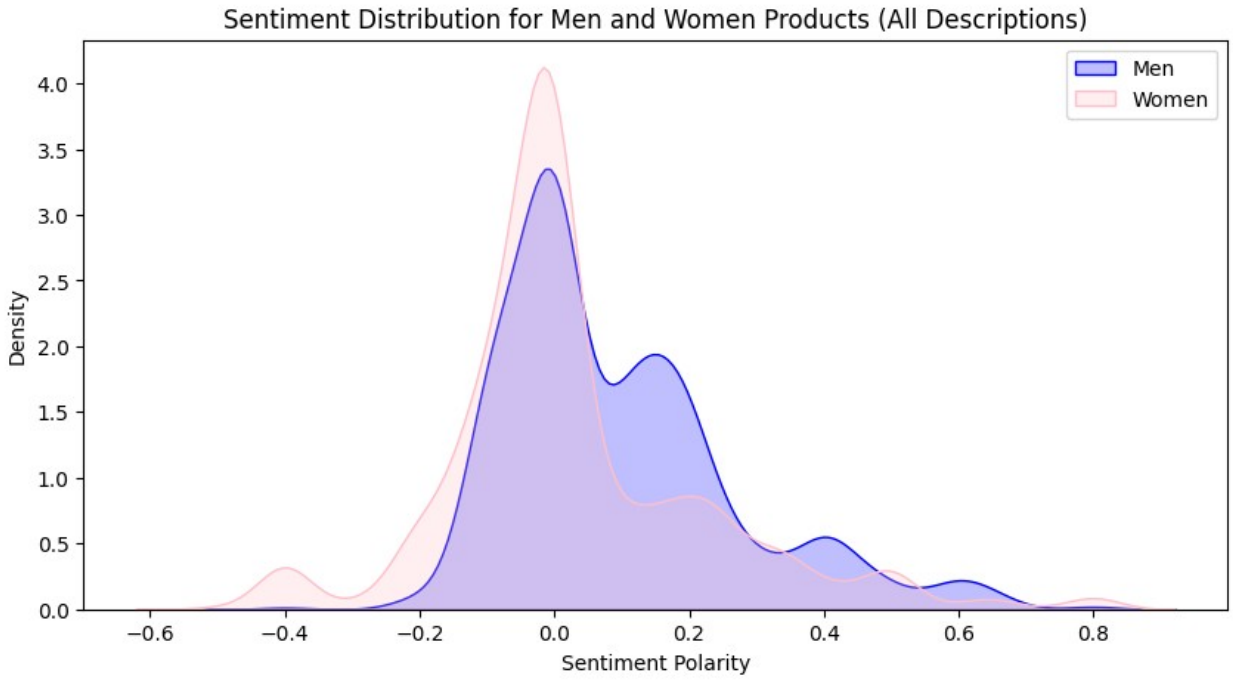
```
n_topics = 10
lda_men = LatentDirichletAllocation(n_components=n_topics,
random_state=42)
lda_women = LatentDirichletAllocation(n_components=n_topics,
random_state=42)

men_topic_weights = lda_men.fit_transform(men_descriptions)
women_topic_weights = lda_women.fit_transform(women_descriptions)

# Example

men_summer_products =
men_data[men_data['details'].str.contains('summer', case=False)]
women_summer_products =
women_data[women_data['Details'].str.contains('summer', case=False)]

# Filter men's and women's products for summer
men_summer_products =
men_data[men_data['details'].str.contains('summer', case=False,
na=False)]
```

```
women_summer_products =
women_data[women_data['Details'].str.contains('summer', case=False,
na=False)]

# Display the first few records in the subsets
print("Men's Summer Products:")
print(men_summer_products.head())

print("\nWomen's Summer Products:")
print(women_summer_products.head())
```

```
Men's Summer Products:
     Unnamed: 0                                product_name  \
117         295                      SOFT WOVEN TOTE BAG
4             4  SEOUL + SEOUL SUMMER 100ML / 3.38 oz

                                                      link  \
117  https://www.zara.com/in/en/soft-woven-tote-bag...
4    https://www.zara.com/in/en/seoul-seoul-summer-...

                                          product_images       price  \
117  [{'https://static.zara.net/photos///2023/I/1/2...  ₹ 3,990.00
4    [{'https://static.zara.net/photos///2023/V/2/2...  ₹ 1,790.00

                                                 details
description_sentiment
117  Tote bag. Soft construction in a combination o...
0.179167
4    ZARA SEOUL 532-8 SINSA DONG GANGNAM-GU EDT 100...
0.216667

Women's Summer Products:
    Unnamed: 0              Product_Name  \
14          19  HIBISCUS 90 ML / 3.04 oz

                                                   Link  \
14  https://www.zara.com/in/en/hibiscus-90-ml---3-...

                                          Product_Image       Price  \
14  [{'https://static.zara.net/photos///2023/V/2/1...  ₹ 1,290.00

                                               Details
description_sentiment
14  ZARA HIBISCUS EDP 90 ML (3.0 FL. OZ). Eau de p...
0.1
```

Calculate sentiment for all descriptions in men's and women's products

```
from textblob import TextBlob
import seaborn as sns
import matplotlib.pyplot as plt


all_men_descriptions = men_data['details'].str.cat(sep=' ')
all_women_descriptions = women_data['Details'].str.cat(sep=' ')

def calculate_sentiment(text):
    try:
        sentiment = TextBlob(text).sentiment.polarity
        return sentiment
    except:
        return 0.0

men_data['description_sentiment'] =
men_data['details'].apply(calculate_sentiment)
women_data['description_sentiment'] =
women_data['Details'].apply(calculate_sentiment)

# Sentiment distribution for all descriptions
plt.figure(figsize=(10, 5))
sns.kdeplot(men_data['description_sentiment'], color='blue',
label='Men', fill=True)
sns.kdeplot(women_data['description_sentiment'], color='pink',
label='Women', fill=True)
plt.title('Sentiment Distribution for Men and Women Products (All
Descriptions)')
plt.xlabel('Sentiment Polarity')
plt.ylabel('Density')
plt.legend(loc='upper right')
plt.show()
```




```
# Concatenate all product descriptions in men's and women's data

all_men_descriptions = men_data['details'].str.cat(sep=' ')
all_women_descriptions = women_data['Details'].str.cat(sep=' ')
```

Generate word cloud for men's descriptions

```
wordcloud_men = WordCloud(width=800, height=400,
background_color='white').generate(all_men_descriptions)
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.imshow(wordcloud_men, interpolation='bilinear')
plt.title("Word Cloud for Men Descriptions")
plt.axis('off')

# Generate word cloud for women's descriptions
wordcloud_women = WordCloud(width=800, height=400,
background_color='white').generate(all_women_descriptions)
plt.subplot(1, 2, 2)
plt.imshow(wordcloud_women, interpolation='bilinear')
plt.title("Word Cloud for Women Descriptions")
plt.axis('off')

plt.show()
```

Extract word frequencies from the word clouds

```
men_word_frequencies = wordcloud_men.words_
women_word_frequencies = wordcloud_women.words_

# Get the top 10 words for men and women
top_10_men_words = list(men_word_frequencies.keys())[:10]
top_10_women_words = list(women_word_frequencies.keys())[:10]
```

Print the top 10 words for men and women

```
print("Top 10 Words for Men:")
print(top_10_men_words)

print("\nTop 10 Words for Women:")
print(top_10_women_words)
```

```
Top 10 Words for Men:
['short sleeve', 'front', 'patch pocket', 'x x', 'long sleeve', 'shirt
made', 'Relaxed fit', 'Front pocket', 'elasticated waistband', 'button
fastening']

Top 10 Words for Women:
['long sleeve', 'front', 'round neck', 'zip fastening', 'button
fastening', 'V neck', 'detail', 'patch pocket', 'High waist', 'Midi
dress']
```

Function to calculate sentiment polarity

```
def calculate_sentiment(text):
    try:
        # Convert non-string data to an empty string
        text = str(text) if text is not None else ""
        sentiment = TextBlob(text).sentiment.polarity
```

```
        return sentiment
    except Exception as e:
        print(f"Error calculating sentiment: {e}")
        return None


men_directory =
'/kaggle/input/zara-dataset-men-and-women-clothing/Men/Men'
women_directory =
'/kaggle/input/zara-dataset-men-and-women-clothing/Women/Women'


men_csv_files = [os.path.join(men_directory, file) for file in
os.listdir(men_directory) if file.endswith('.csv')]
women_csv_files = [os.path.join(women_directory, file) for file in
os.listdir(women_directory) if file.endswith('.csv')]

# Read all CSV files for men and women into DataFrames
men_data = pd.concat([pd.read_csv(file) for file in men_csv_files])
women_data = pd.concat([pd.read_csv(file) for file in
women_csv_files])

# Calculate sentiment for men's and women's product descriptions
men_data['description_sentiment'] =
men_data['details'].apply(calculate_sentiment)
women_data['description_sentiment'] =
women_data['Details'].apply(calculate_sentiment)
```

Sentiment Distribution for Men and Women Products

```
# Plot sentiment distribution for men's and women's products
plt.figure(figsize=(10, 5))
plt.hist(men_data['description_sentiment'], bins=20, alpha=0.5,
color='blue', label='Men')
plt.hist(women_data['description_sentiment'], bins=20, alpha=0.5,
color='pink', label='Women')
plt.title('Sentiment Distribution for Men and Women Products')
plt.xlabel('Sentiment Polarity')
plt.ylabel('Frequency')
plt.legend(loc='upper right')
```

```
<matplotlib.legend.Legend at 0x799f140cece0>
```


```
pip install vaderSentiment

Requirement already satisfied: vaderSentiment in
/opt/conda/lib/python3.10/site-packages (3.3.2)
Requirement already satisfied: requests in
/opt/conda/lib/python3.10/site-packages (from vaderSentiment) (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/opt/conda/lib/python3.10/site-packages (from requests-
>vaderSentiment) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/lib/python3.10/site-packages (from requests-
>vaderSentiment) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/opt/conda/lib/python3.10/site-packages (from requests-
>vaderSentiment) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.10/site-packages (from requests-
>vaderSentiment) (2023.7.22)
Note: you may need to restart the kernel to use updated packages.

import pandas as pd
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

analyzer = SentimentIntensityAnalyzer()

# Function to calculate sentiment
def calculate_sentiment(text):
    if isinstance(text, str):
        sentiment = analyzer.polarity_scores(text)
```

```
        compound_score = sentiment['compound']
        if compound_score >= 0.05:
            return "Positive"
        elif compound_score <= -0.05:
            return "Negative"
        else:
            return "Neutral"
    else:
        return "Neutral"

all_men_data = pd.DataFrame()
all_women_data = pd.DataFrame()

for csv_file in men_csv_files:
    men_data = pd.read_csv(csv_file)
    all_men_data = pd.concat([all_men_data, men_data])

for csv_file in women_csv_files:
    women_data = pd.read_csv(csv_file)
    all_women_data = pd.concat([all_women_data, women_data])
```

Apply sentiment analysis to the concatenated descriptions

```
all_men_data['description_sentiment'] =
all_men_data['details'].fillna('').apply(calculate_sentiment)
all_women_data['description_sentiment'] =
all_women_data['Details'].fillna('').apply(calculate_sentiment)
```

Count the number of descriptions in each sentiment category

```
men_sentiment_counts =
all_men_data['description_sentiment'].value_counts()
women_sentiment_counts =
all_women_data['description_sentiment'].value_counts()
```

The sentiment distribution for men and women

```
print("Sentiment Distribution for Men's Products:")
print(men_sentiment_counts)
```

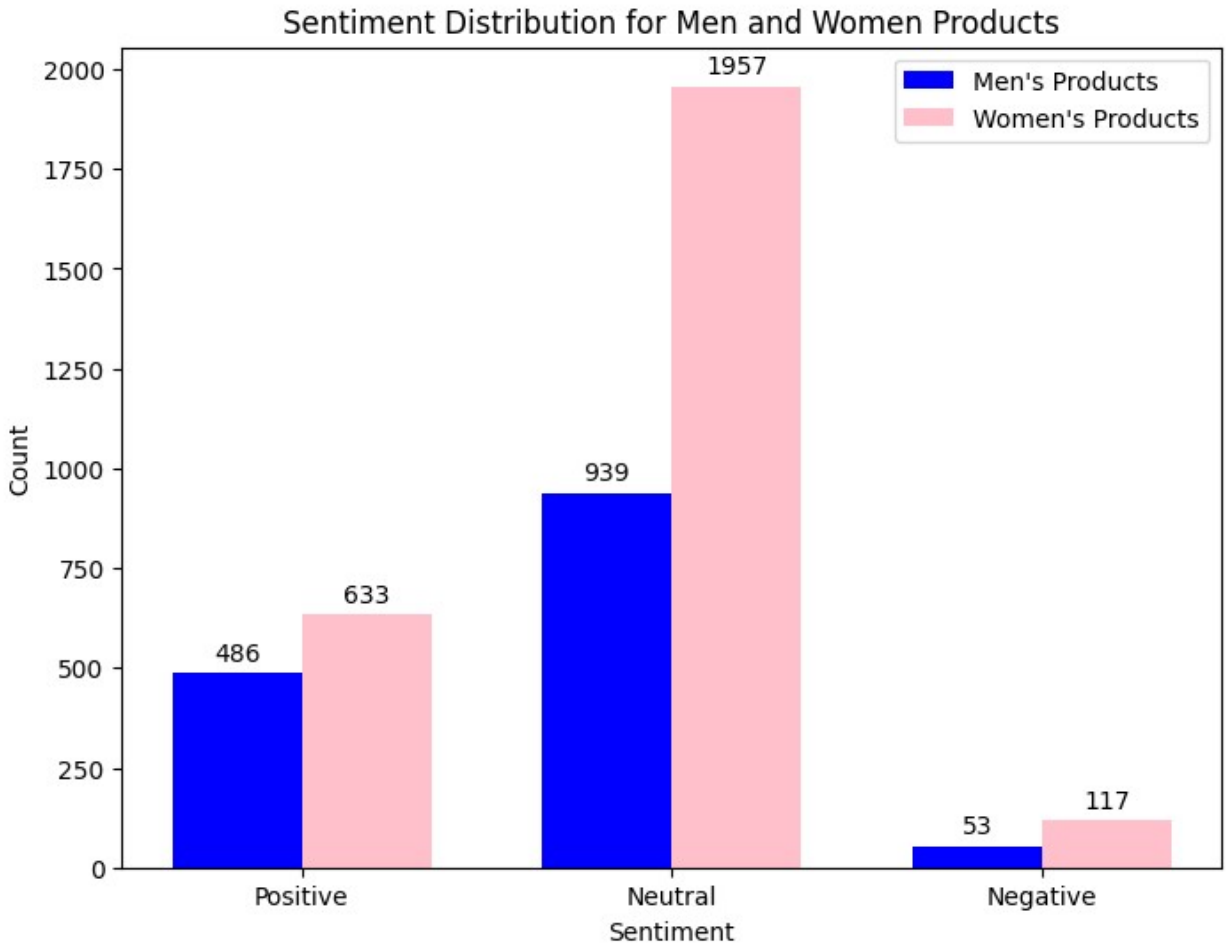
```
print("\nSentiment Distribution for Women's Products:")
print(women_sentiment_counts)
```

```
Sentiment Distribution for Men's Products:
description_sentiment
Neutral     939
Positive    486
Negative     53
Name: count, dtype: int64

Sentiment Distribution for Women's Products:
description_sentiment
Neutral     1957
Positive     633
Negative     117
Name: count, dtype: int64
```

Sentiment Distribution for Men and Women Products

```
import matplotlib.pyplot as plt

# Sentiment distribution data
sentiments = ['Positive', 'Neutral', 'Negative']
men_counts = [men_sentiment_counts.get(sentiment, 0) for sentiment in
sentiments]
women_counts = [women_sentiment_counts.get(sentiment, 0) for sentiment
in sentiments]

# Create a bar chart
fig, ax = plt.subplots(figsize=(8, 6))
width = 0.35
x = range(len(sentiments))

bar1 = ax.bar(x, men_counts, width, label="Men's Products",
color='blue')
bar2 = ax.bar([i + width for i in x], women_counts, width,
label="Women's Products", color='pink')

ax.set_xlabel('Sentiment')
ax.set_ylabel('Count')
ax.set_title('Sentiment Distribution for Men and Women Products')
ax.set_xticks([i + width / 2 for i in x])
ax.set_xticklabels(sentiments)
ax.legend()

# Add data labels above each bar
```

```
for bar in [bar1, bar2]:
    for rect in bar:
        height = rect.get_height()
        ax.annotate('{}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')

plt.show()
```



Use case 2 - Price Analysis

```
men_data
```

```
    Unnamed: 0                                product_name  \
0            0                          100% LINEN SHIRT
1            5                VISCOSE/LINEN BLEND SHIRT
2            7                       FLORAL PRINT SHIRT
```

```
62          106  LINEN BLEND CREASED-EFFECT BLAZER
63          108                      CROPPED BLAZER
64          115         SHORT TEXTURED WEAVE BLAZER
65          116     TUXEDO COLLAR BLAZER WITH BELT
66          117                    STRAIGHT BLAZER

                                                  Link  \
0   https://www.zara.com/in/en/tailored-double-bre...
1   https://www.zara.com/in/en/stripe-textured-bla...
2   https://www.zara.com/in/en/tailored-waistcoat-...
3   https://www.zara.com/in/en/straight-blazer-p09...
4   https://www.zara.com/in/en/blazer-with-rolled-...
..                                                ...
62  https://www.zara.com/in/en/linen-blend-creased...
63  https://www.zara.com/in/en/cropped-blazer-p012...
64  https://www.zara.com/in/en/short-textured-weav...
65  https://www.zara.com/in/en/tuxedo-collar-blaze...
66  https://www.zara.com/in/en/straight-blazer-p08...

                                        Product_Image       Price  \
0   [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 5,590.00
1   [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 5,990.00
2   [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 2,990.00
3   [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 4,990.00
4   [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 4,490.00
..                                                ...         ...
62                                                 []  ₹ 8,590.00
63                                                 []  ₹ 4,990.00
64                                                 []  ₹ 5,990.00
65                                                 []  ₹ 8,590.00
66                                                 []  ₹ 4,990.00

                                              Details
0   Blazer featuring a lapel collar with long slee...
1   Lapelless blazer with long sleeves and shoulde...
2   Sleeveless waistcoat with a V-neck. Front jett...
3   Blazer featuring a lapel collar with long slee...
4   Open blazer with a lapel collar and padded sho...
..                                                ...
62  Blazer with a lapel collar and long sleeves. F...
63  Cropped blazer featuring a high neck and long ...
64  Blazer with a high neck and long sleeves. Fron...
65  Blazer with a tuxedo collar and long sleeves w...
66  Blazer featuring a lapel collar and long sleev...

[67 rows x 6 columns]
```

```
print("Men's Data:")
print(men_data.head())
```

```
print("\nWomen's Data:")
print(women_data.head())
```

```
Men's Data:
   Unnamed: 0               product_name  \
0           0           100% LINEN SHIRT
1           5  VISCOSE/LINEN BLEND SHIRT
2           7         FLORAL PRINT SHIRT
3           9      VISCOSE - LINEN SHIRT
4          10           PLEATED TROUSERS

                                                link  \
0  https://www.zara.com/in/en/100-linen-shirt-p06...
1  https://www.zara.com/in/en/viscose-linen-blend...
2  https://www.zara.com/in/en/floral-print-shirt-...
3  https://www.zara.com/in/en/viscose---linen-shi...
4  https://www.zara.com/in/en/pleated-trousers-p0...

                                      product_images       price  \
0  [{'https://static.zara.net/photos///2023/I/0/2...  ₹ 3,990.00
1  [{'https://static.zara.net/photos///2023/I/0/2...  ₹ 3,290.00
2  [{'https://static.zara.net/photos///2023/I/0/2...  ₹ 3,290.00
3  [{'https://static.zara.net/photos///2023/I/0/2...  ₹ 3,290.00
4  [{'https://static.zara.net/photos///2023/I/0/2...  ₹ 4,990.00

                                             details
0  Regular-fit shirt made of lightweight linen fa...
1  Relaxed fit shirt in a linen and viscose blend...
2  Relaxed fit shirt with a camp collar, short sl...
3  Relaxed fit shirt made of a viscose and cotton...
4  Straight fit trousers. Waist with front pleate...

Women's Data:
   Unnamed: 0                    Product_Name  \
0           0  TAILORED DOUBLE-BREASTED BLAZER
1           1           STRIPE TEXTURED BLAZER
2           2               TAILORED WAISTCOAT
3           3                  STRAIGHT BLAZER
4           4    BLAZER WITH ROLLED-UP SLEEVES

                                                Link  \
0  https://www.zara.com/in/en/tailored-double-bre...
1  https://www.zara.com/in/en/stripe-textured-bla...
2  https://www.zara.com/in/en/tailored-waistcoat-...
3  https://www.zara.com/in/en/straight-blazer-p09...
4  https://www.zara.com/in/en/blazer-with-rolled-...

                                      Product_Image       Price  \
0  [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 5,590.00
1  [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 5,990.00
```

```
2  [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 2,990.00
3  [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 4,990.00
4  [{'https://static.zara.net/photos///2023/I/0/1...  ₹ 4,490.00

                                              Details
0  Blazer featuring a lapel collar with long slee...
1  Lapelless blazer with long sleeves and shoulde...
2  Sleeveless waistcoat with a V-neck. Front jett...
3  Blazer featuring a lapel collar with long slee...
4  Open blazer with a lapel collar and padded sho...
```

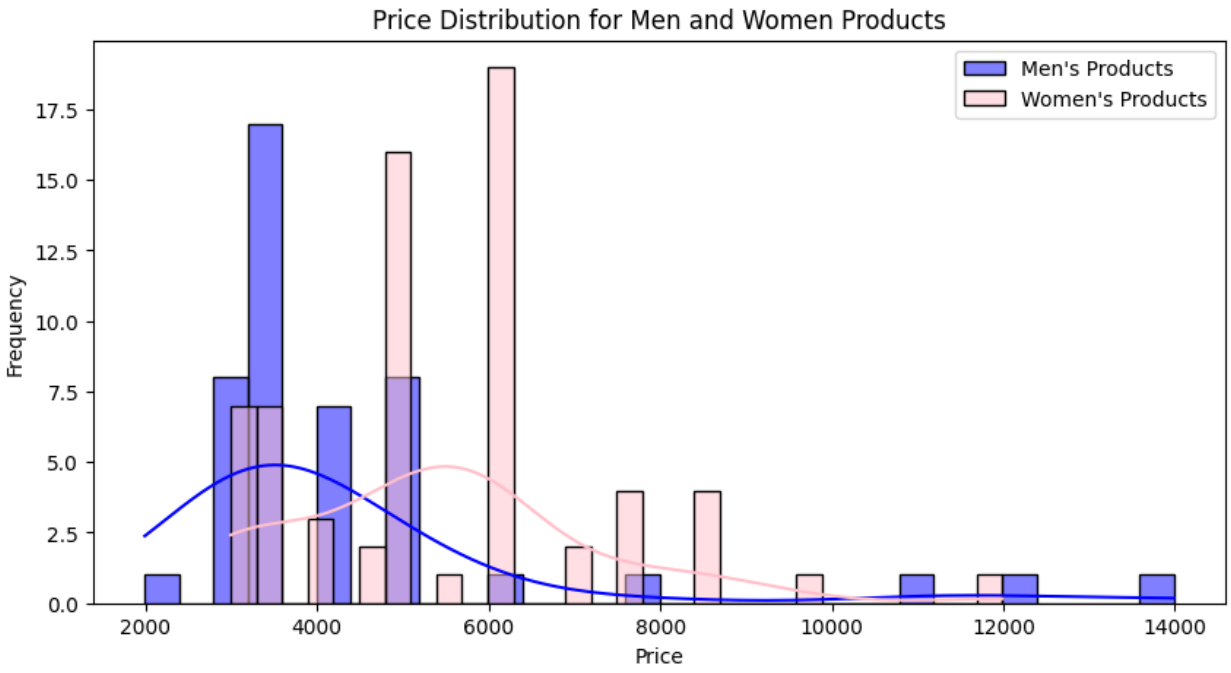
Check for missing values

```
men_missing_values = men_data.isnull().sum()
women_missing_values = women_data.isnull().sum()
```

Drop rows with missing prices

```
men_data = men_data.dropna(subset=['price'])
women_data = women_data.dropna(subset=['Price'])

# Assuming you have an exchange rate (e.g., 1 INR to Euro = 0.012 EUR)
exchange_rate_inr_to_euro = 0.012

# Check if 'price' columns are already in numeric format, if not,
convert them
if not pd.api.types.is_numeric_dtype(men_data['price']):
    men_data['price'] = men_data['price'].str.replace('₹',
'').str.replace(',', '').astype(float)

if not pd.api.types.is_numeric_dtype(women_data['Price']):
    women_data['Price'] = women_data['Price'].str.replace('₹',
'').str.replace(',', '').astype(float)

# Check if you have the exchange rate data, if not, set it to 1 (no
conversion)
if 'price' in men_data and 'price' in women_data:
    men_data['price'] *= exchange_rate_inr_to_euro
    women_data['Price'] *= exchange_rate_inr_to_euro

import matplotlib.pyplot as plt
import seaborn as sns

# Visualize price distribution for men's and women's products
plt.figure(figsize=(10, 5))
sns.histplot(men_data['price'], bins=30, color='blue', label="Men's
```

```
Products", kde=True)
sns.histplot(women_data['Price'], bins=30, color='pink',
label="Women's Products", kde=True)
plt.title('Price Distribution for Men and Women Products')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



Calculate some statistics info

```
men_price_summary = men_data['price'].describe()
women_price_summary = women_data['Price'].describe()

print("Men's Price Summary:")
print(men_price_summary)

print("\nWomen's Price Summary:")
print(women_price_summary)
```

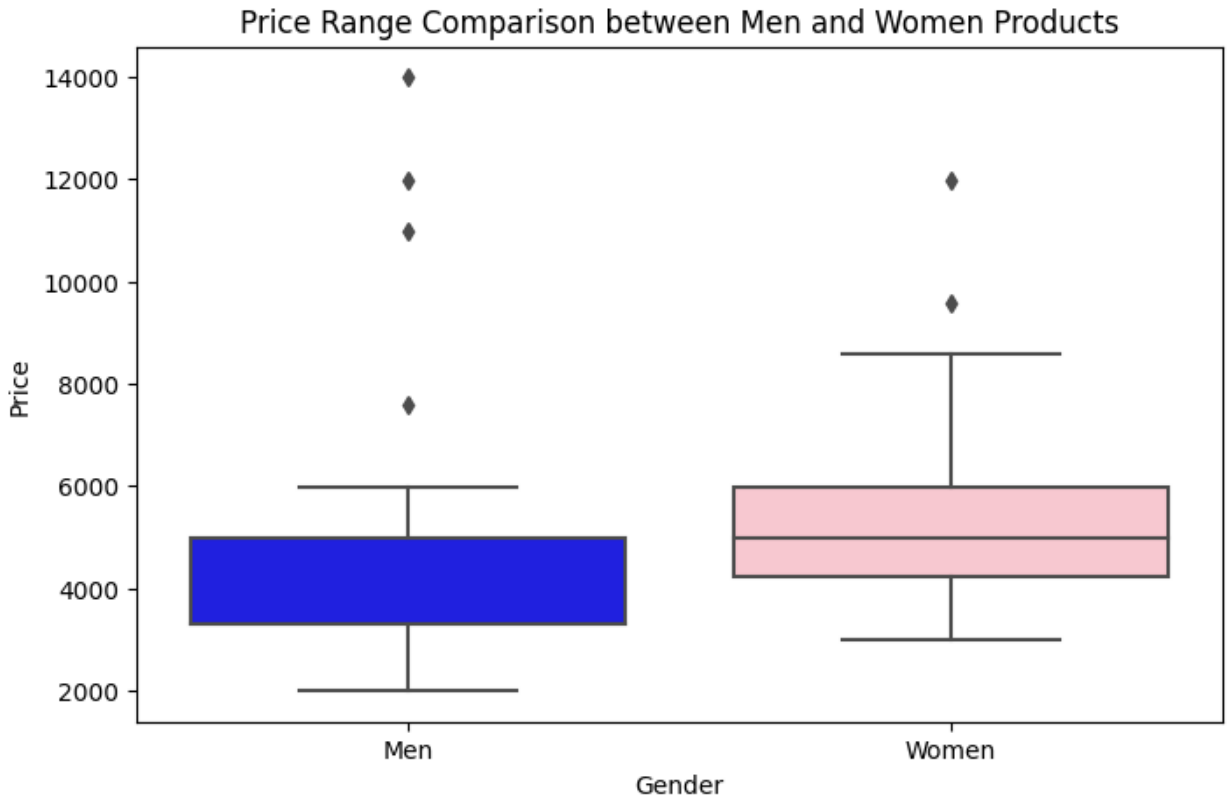
```
Men's Price Summary:
count       46.000000
mean      4348.695652
std       2370.145433
min       1990.000000
25%       3290.000000
50%       3290.000000
```

```
75%       4990.000000
max      13990.000000
Name: price, dtype: float64

Women's Price Summary:
count       67.000000
mean      5440.746269
std       1802.470790
min       2990.000000
25%       4240.000000
50%       4990.000000
75%       5990.000000
max      11990.000000
Name: Price, dtype: float64
```

```
women_data.rename(columns={'Price': 'price'}, inplace=True)
```

Compare price ranges between men's and women's products

```
plt.figure(figsize=(8, 5))
sns.boxplot(x='gender', y='price',
data=pd.concat([men_data.assign(gender='Men'),
women_data.assign(gender='Women')]), palette=['blue', 'pink'])
plt.title('Price Range Comparison between Men and Women Products')
plt.xlabel('Gender')
plt.ylabel('Price')
plt.show()
```



Use Case 3 - Recommender System

```
men_data
```

```
    Unnamed: 0                                  product_name  \
0            0                          100% LINEN SHIRT
1            5                 VISCOSE/LINEN BLEND SHIRT
2            7                        FLORAL PRINT SHIRT
3            9                    VISCOSE - LINEN SHIRT
4           10                         PLEATED TROUSERS
5           12     BERMUDA SHORTS WITH CONTRAST EMBROIDERY
6           13                       100% LINEN TROUSERS
7           20                     COTTON - LINEN SHIRT
8           27                LINEN - COTTON POLO SHIRT
9           28                     COTTON - LINEN SHIRT
10          39            LINEN - COTTON BERMUDA SHORTS
11          45                100% LINEN BERMUDA SHORTS
12          52                   COTTON - LINEN T-SHIRT
13          57                 COTTON - LINEN TROUSERS
14          63                     COTTON - LINEN SHIRT
15          68                VISCOSE/LINEN BLEND SHIRT
16          69             VISCOSE - LINEN BERMUDA SHORTS
17          75             TEXTURED OXFORD BERMUDA SHORTS
```

```
21  Trousers in a viscose and linen blend fabric. ...
22  Relaxed-fit sleeveless shirt made of a cotton ...
23  Relaxed fit Bermuda shorts made of a linen and...
24  Relaxed fit shirt made of a cotton and linen b...
25  Regular-fit trousers made of cotton and linen ...
26  Shirt made of a viscose and cotton blend fabri...
27  Linen waistcoat. V-neckline and no sleeves. We...
28  Regular-fit trousers made of a viscose and lin...
29  Slim-fit trousers with an elasticated waistban...
30  Regular-fit shirt made of a cotton and linen b...
31  Regular fit trousers made of a cotton and line...
32  Relaxed fit collared shirt made of a linen and...
33  Regular fit trousers made of a linen and lyoce...
34  Regular fit spread collar shirt featuring long...
35  Regular-fit blazer made of linen. Notched lape...
36  Straight fit trousers made of linen. Front poc...
37  Relaxed fit blazer made of a linen and cotton ...
38  Straight fit blazer made of linen. Notched lap...
39  Regular fit trousers. Waist with front pleats....
40  Regular-fit trousers. Front pockets and back w...
41  Relaxed-fit trousers with an adjustable elasti...
42  Loose-fitting knit polo shirt made of spun lin...
43  Lightweight pareo made of linen. Featuring a c...
44  Bermuda shorts made of linen fabric. Adjustabl...
45  Straight fit trousers made of linen. Waist wit...
```

```
women_data.rename(columns={'Product_Name': 'product_name'},
inplace=True)
women_data.rename(columns={'Details': 'details'}, inplace=True)

combined_data = pd.concat([men_data, women_data])

combined_data.head
```

```
<bound method NDFrame.head of     Unnamed: 0
product_name  \
0            0                    100% LINEN SHIRT
1            5          VISCOSE/LINEN BLEND SHIRT
2            7                 FLORAL PRINT SHIRT
3            9              VISCOSE - LINEN SHIRT
4           10                   PLEATED TROUSERS
..         ...                                ...
62         106  LINEN BLEND CREASED-EFFECT BLAZER
63         108                     CROPPED BLAZER
64         115        SHORT TEXTURED WEAVE BLAZER
65         116     TUXEDO COLLAR BLAZER WITH BELT
66         117                     STRAIGHT BLAZER

                                                 link  \
0   https://www.zara.com/in/en/100-linen-shirt-p06...
```

```
1   https://www.zara.com/in/en/viscose-linen-blend...
2   https://www.zara.com/in/en/floral-print-shirt-...
3   https://www.zara.com/in/en/viscose---linen-shi...
4   https://www.zara.com/in/en/pleated-trousers-p0...
..                                                ...
62                                                NaN
63                                                NaN
64                                                NaN
65                                                NaN
66                                                NaN

                                          product_images   price  \
0   [{'https://static.zara.net/photos///2023/I/0/2...  3990.0
1   [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
2   [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
3   [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
4   [{'https://static.zara.net/photos///2023/I/0/2...  4990.0
..                                                ...     ...
62                                                NaN  8590.0
63                                                NaN  4990.0
64                                                NaN  5990.0
65                                                NaN  8590.0
66                                                NaN  4990.0

                                               details  \
0   Regular-fit shirt made of lightweight linen fa...
1   Relaxed fit shirt in a linen and viscose blend...
2   Relaxed fit shirt with a camp collar, short sl...
3   Relaxed fit shirt made of a viscose and cotton...
4   Straight fit trousers. Waist with front pleate...
..                                                ...
62  Blazer with a lapel collar and long sleeves. F...
63  Cropped blazer featuring a high neck and long ...
64  Blazer with a high neck and long sleeves. Fron...
65  Blazer with a tuxedo collar and long sleeves w...
66  Blazer featuring a lapel collar and long sleev...

                                                Link Product_Image
0                                                NaN           NaN
1                                                NaN           NaN
2                                                NaN           NaN
3                                                NaN           NaN
4                                                NaN           NaN
..                                               ...           ...
62  https://www.zara.com/in/en/linen-blend-creased...            []
63  https://www.zara.com/in/en/cropped-blazer-p012...            []
64  https://www.zara.com/in/en/short-textured-weav...            []
65  https://www.zara.com/in/en/tuxedo-collar-blaze...            []
66  https://www.zara.com/in/en/straight-blazer-p08...            []
```

```
[113 rows x 8 columns]>
```

Drop duplicate product descriptions

```
combined_data.drop_duplicates(subset='product_name', keep='first',
inplace=True)
```

Handle missing values if necessary

```
combined_data.dropna(subset=['product_name'], inplace=True)
```

Reset the index

```
combined_data.reset_index(drop=True, inplace=True)
print(combined_data.head())
```

```
   Unnamed: 0                product_name  \
0           0            100% LINEN SHIRT
1           5   VISCOSE/LINEN BLEND SHIRT
2           7          FLORAL PRINT SHIRT
3           9       VISCOSE - LINEN SHIRT
4          10            PLEATED TROUSERS

                                                link  \
0  https://www.zara.com/in/en/100-linen-shirt-p06...
1  https://www.zara.com/in/en/viscose-linen-blend...
2  https://www.zara.com/in/en/floral-print-shirt-...
3  https://www.zara.com/in/en/viscose---linen-shi...
4  https://www.zara.com/in/en/pleated-trousers-p0...

                                      product_images   price  \
0  [{'https://static.zara.net/photos///2023/I/0/2...  3990.0
1  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
2  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
3  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
4  [{'https://static.zara.net/photos///2023/I/0/2...  4990.0

                                          details Link
Product_Image
0  Regular-fit shirt made of lightweight linen fa...  NaN
NaN
1  Relaxed fit shirt in a linen and viscose blend...  NaN
NaN
2  Relaxed fit shirt with a camp collar, short sl...  NaN
```

```
NaN
3  Relaxed fit shirt made of a viscose and cotton...  NaN
NaN
4  Straight fit trousers. Waist with front pleate...  NaN
NaN
```

Feature Extraction

```
from sklearn.feature_extraction.text import TfidfVectorizer

# Initialize the TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer()

# Fit and transform the descriptions
tfidf_matrix =
tfidf_vectorizer.fit_transform(combined_data['product_name'])

# Print the shape of the TF-IDF matrix
print(f"TF-IDF Matrix Shape: {tfidf_matrix.shape}")
```

```
TF-IDF Matrix Shape: (88, 89)
```

Building the Recommendation System

```
from sklearn.metrics.pairwise import cosine_similarity

# Calculate cosine similarity between products
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)

# Print the shape of the cosine similarity matrix
print(f"Cosine Similarity Matrix Shape: {cosine_sim.shape}")
```

```
Cosine Similarity Matrix Shape: (88, 88)
```

Recommendation Function

unique product names in the dataset

```
unique_product_names = combined_data['product_name'].unique()
for product_name in unique_product_names:
    print(product_name)
```

```
100% LINEN SHIRT
VISCOSE/LINEN BLEND SHIRT
FLORAL PRINT SHIRT
```

```
        # Get the top-n most similar products
        sim_scores = sim_scores[1:top_n+1]

        # Get the product indices
        product_indices = [i[0] for i in sim_scores]

        # Get the recommended product names
        recommended_products =
data['product_name'].iloc[product_indices]

        # Return the recommended products as a list
        return recommended_products.tolist()
    except IndexError:
        # Handle the case where the product name is not found
        return ["Product not found in the dataset"]

# Iterate through all unique product names and get recommendations
unique_product_names = combined_data['product_name'].unique()
for product_name in unique_product_names:
    recommended_products = recommend_products(product_name,
cosine_sim, combined_data)

    # Print the product and its recommended products
    print(f"Product: {product_name}")
    print("Recommended Products:")
    for product in recommended_products:
        print(product)
    print("\n")
```

```
Product: 100% LINEN SHIRT
Recommended Products:
100% LINEN WAISTCOAT
100% LINEN TROUSERS
100% LINEN SUIT BLAZER
COTTON - LINEN SHIRT
COTTON - LINEN T-SHIRT


Product: VISCOSE/LINEN BLEND SHIRT
Recommended Products:
VISCOSE - LINEN SHIRT
LINEN - COTTON BLEND SHIRT
VISCOSE - LINEN TROUSERS
LINEN - VISCOSE TROUSERS
LINEN BLEND WAISTCOAT


Product: FLORAL PRINT SHIRT
Recommended Products:
```

```
reverse=True)

        # Get the top-n most similar products
        sim_scores = sim_scores[1:top_n + 1]

        # Get the product indices
        product_indices = [i[0] for i in sim_scores]

        # Get the recommended product names
        recommended_products =
data['product_name'].iloc[product_indices]

        # Create a DataFrame to display recommendations
        recommendations_df = pd.DataFrame({'Recommended Products':
recommended_products})

        return recommendations_df
    except IndexError:

        return pd.DataFrame({'Recommended Products': ["Product not
found in the dataset"]})

# Example usage:
product_description = "100% LINEN SHIRT"
recommended_products_df = get_recommendations(product_description,
cosine_sim, combined_data)
recommended_products_df.reset_index(drop=True, inplace=True)
print("Recommended Products:")
print(recommended_products_df)
```

```
Recommended Products:
     Recommended Products
0    100% LINEN WAISTCOAT
1     100% LINEN TROUSERS
2  100% LINEN SUIT BLAZER
3    COTTON - LINEN SHIRT
4  COTTON - LINEN T-SHIRT
```

Function to get product recommendations as a list

```
def get_recommendations_list(product_description, cosine_sim_matrix,
data, top_n=5):
    try:
        # Find the index of the product in the dataset
        idx = data[data['product_name'] ==
product_description].index[0]
```

```
        # Get the pairwise similarity scores for the product
        sim_scores = list(enumerate(cosine_sim_matrix[idx]))

        # Sort the products based on similarity scores
        sim_scores = sorted(sim_scores, key=lambda x: x[1],
reverse=True)

        # Get the top-n most similar products
        sim_scores = sim_scores[1:top_n + 1]

        # Get the product indices
        product_indices = [i[0] for i in sim_scores]

        # Get the recommended product names as a list
        recommended_products =
data['product_name'].iloc[product_indices].tolist()

        return recommended_products
    except IndexError:

        return ["Product not found in the dataset"]

# Example usage:
product_description = "100% LINEN SHIRT"
recommended_products_list =
get_recommendations_list(product_description, cosine_sim,
combined_data)

# Combine the recommended products into a single string
recommended_products_text = " ".join(recommended_products_list)


wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(recommended_products_text)


plt.figure(figsize=(20, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Product Recommendations Word Cloud")
plt.show()
```

Use case 4: Market Research

```
market_data = pd.concat([men_data, women_data], ignore_index=True)
```

Some insight about the dataset

```
print("Number of rows and columns:", market_data.shape)
print("\nColumn names:", market_data.columns)
print("\nData types:\n", market_data.dtypes)
```

```
Number of rows and columns: (113, 8)

Column names: Index(['Unnamed: 0', 'product_name', 'link', '
product_images', 'price',
       'details', 'Link', 'Product_Image'],
      dtype='object')

Data types:
 Unnamed: 0           int64
product_name         object
link                 object
 product_images      object
price               float64
details              object
```

```
Link                object
Product_Image       object
dtype: object
```

```
print("First few rows of the dataset:")
print(market_data.head())
```

```
First few rows of the dataset:
   Unnamed: 0              product_name  \
0           0           100% LINEN SHIRT
1           5  VISCOSE/LINEN BLEND SHIRT
2           7         FLORAL PRINT SHIRT
3           9      VISCOSE - LINEN SHIRT
4          10           PLEATED TROUSERS

                                                link  \
0  https://www.zara.com/in/en/100-linen-shirt-p06...
1  https://www.zara.com/in/en/viscose-linen-blend...
2  https://www.zara.com/in/en/floral-print-shirt-...
3  https://www.zara.com/in/en/viscose---linen-shi...
4  https://www.zara.com/in/en/pleated-trousers-p0...

                                      product_images   price  \
0  [{'https://static.zara.net/photos///2023/I/0/2...  3990.0
1  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
2  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
3  [{'https://static.zara.net/photos///2023/I/0/2...  3290.0
4  [{'https://static.zara.net/photos///2023/I/0/2...  4990.0

                                           details Link
Product_Image
0  Regular-fit shirt made of lightweight linen fa...  NaN
NaN
1  Relaxed fit shirt in a linen and viscose blend...  NaN
NaN
2  Relaxed fit shirt with a camp collar, short sl...  NaN
NaN
3  Relaxed fit shirt made of a viscose and cotton...  NaN
NaN
4  Straight fit trousers. Waist with front pleate...  NaN
NaN
```

Check for missing values in each column

```
missing_values = market_data.isnull().sum()
print("\nMissing values in each column:")
print(missing_values)
```

```
Missing values in each column:
Unnamed: 0          0
product_name        0
link               67
 product_images    67
price               0
details             0
Link               46
Product_Image      46
dtype: int64
```

Summary statistics

```
summary_statistics = market_data['price'].describe()
print("\nSummary statistics for 'price' column:")
print(summary_statistics)
```

```
Summary statistics for 'price' column:
count      113.000000
mean      4996.194690
std       2112.348185
min       1990.000000
25%       3290.000000
50%       4990.000000
75%       5990.000000
max      13990.000000
Name: price, dtype: float64
```

Unique values and their counts for categorical columns

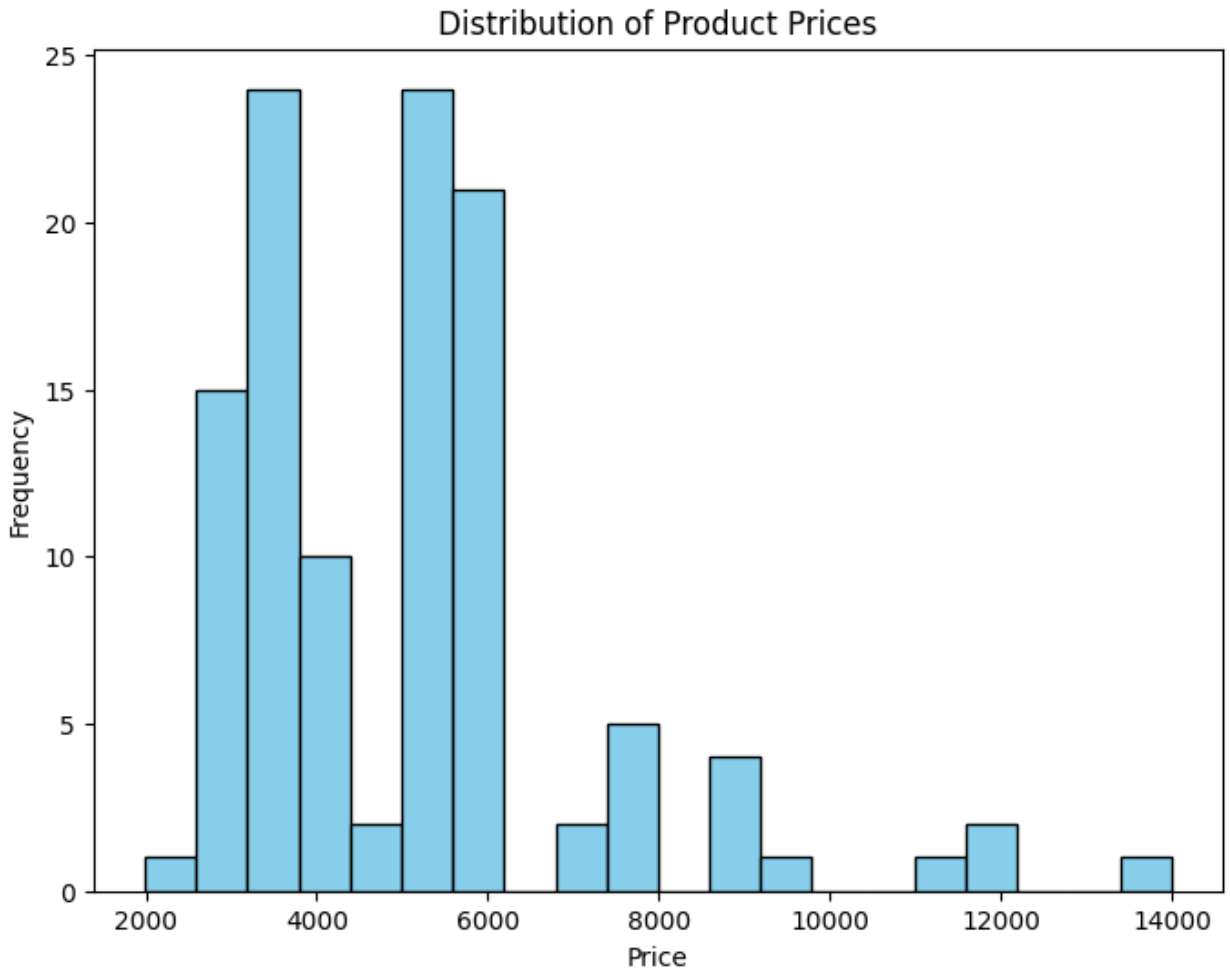
```
unique_values_counts = market_data['product_name'].value_counts()
print("\nUnique values and their counts for 'product_name' column:")
print(unique_values_counts)
```

```
Unique values and their counts for 'product_name' column:
product_name
LINEN BLEND WAISTCOAT                          4
100% LINEN SHIRT                               3
COTTON - LINEN SHIRT                           3
TEXTURED DOUBLE-BREASTED BLAZER                3
TAILORED WAISTCOAT                             3
```

```
                                                ..
TEXTURED KNIT POLO SHIRT                         1
STRIPED LINEN-VISCOSE TROUSERS                   1
FLORAL PRINT SUIT BLAZER - LIMITED EDITION       1
LINEN - LYOCELL TROUSERS                         1
TUXEDO COLLAR BLAZER WITH BELT                   1
Name: count, Length: 88, dtype: int64
```

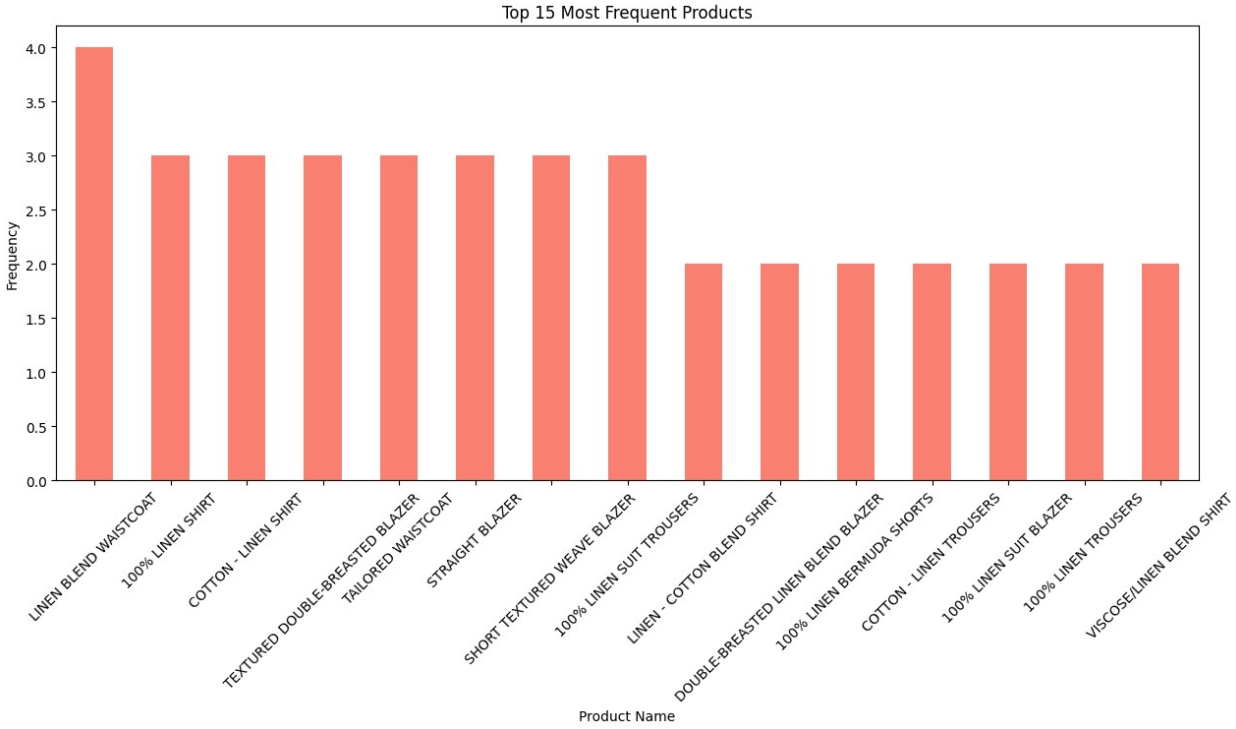
Visualization of 'price' distribution using a histogram

```
plt.figure(figsize=(8, 6))
plt.hist(market_data['price'], bins=20, color='skyblue',
edgecolor='black')
plt.title("Distribution of Product Prices")
plt.xlabel("Price")
plt.ylabel("Frequency")
plt.show()
```



Visualization of top 10 most frequent 'product_name' values using a bar chart

```
top_10_products = unique_values_counts.head(15)
plt.figure(figsize=(15, 6))
top_10_products.plot(kind='bar', color='salmon')
plt.title("Top 15 Most Frequent Products")
plt.xlabel("Product Name")
plt.ylabel("Frequency")
plt.xticks(rotation=45)
plt.show()
```


Top 15 Most Frequent Products
Frequency
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
LINEN BLEND WAISTCOAT
100% LINEN SHIRT
COTTON - LINEN SHIRT
TEXTURED DOUBLE-BREASTED BLAZER
TAILORED WAISTCOAT
STRAIGHT BLAZER
SHORT TEXTURED WEAVE BLAZER
100% LINEN SUIT TROUSERS
LINEN - COTTON BLEND SHIRT
DOUBLE-BREASTED LINEN BLEND BLAZER
100% LINEN BERMUDA SHORTS
COTTON - LINEN TROUSERS
100% LINEN SUIT BLAZER
100% LINEN TROUSERS
VISCOSE/LINEN BLEND SHIRT
Product Name