Import Libaries

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from surprise import Dataset, Reader, SVD
from surprise.model_selection import train_test_split

!pip install scikit-surprise

Collecting scikit-surprise
  Downloading scikit_surprise-1.1.4.tar.gz (154 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 154.4/154.4 kB 1.6 MB/s eta
0:00:00
ents to build wheel ... etadata (pyproject.toml) ... ent already
satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packages
(from scikit-surprise) (1.4.2)
Requirement already satisfied: numpy>=1.19.5 in
/usr/local/lib/python3.10/dist-packages (from scikit-surprise)
(1.26.4)
Requirement already satisfied: scipy>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-surprise)
(1.13.1)
Building wheels for collected packages: scikit-surprise
  Building wheel for scikit-surprise (pyproject.toml) ...
e=scikit_surprise-1.1.4-cp310-cp310-linux_x86_64.whl size=2357278
sha256=6d977fd50aa8d365044ce98fb3c3dc1f1209fd0bb0daafdcdac6588abae3468
6
  Stored in directory:
/root/.cache/pip/wheels/4b/3f/df/6acbf0a40397d9bf3ff97f582cc22fb9ce66a
dde75bc71fd54
Successfully built scikit-surprise
Installing collected packages: scikit-surprise
Successfully installed scikit-surprise-1.1.4
```

IMPORTING DATA

```
df= pd.read_csv('/content/fashion_products.csv')
```

EDA

```
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 1000,\n  \"fields\":
[\n    {\n      \"column\": \"User ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 28,\n        \"min\": 1,\n
\"max\": 100,\n        \"num_unique_values\": 100,\n
\"samples\": [\n          55,\n          4,\n          29\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Product ID\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
288,\n        \"min\": 1,\n        \"max\": 1000,\n
\"num_unique_values\": 1000,\n        \"samples\": [\n          522,\n
738,\n          741\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Product Name\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"Shoes\",\n          \"Sweater\",\n          \"T-
shirt\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Brand\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 5,\n        \"samples\": [\n
\"H&M\",\n          \"Nike\",\n          \"Zara\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Category\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
3,\n        \"samples\": [\n          \"Men's Fashion\",\n
\"Women's Fashion\",\n          \"Kids' Fashion\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Price\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 26,\n
\"min\": 10,\n        \"max\": 100,\n        \"num_unique_values\":
91,\n        \"samples\": [\n          59,\n          21,\n
42\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Rating\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1.1531854436466726,\n        \"min\": 1.000967235428064,\n
\"max\": 4.987964320970842,\n        \"num_unique_values\": 1000,\n
\"samples\": [\n          2.9390797234183057,\n
1.0338431929963692,\n          3.3868368182647823\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Color\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 6,\n
\"samples\": [\n          \"Black\",\n          \"Yellow\",\n
\"Red\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Size\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          \"L\",\n
\"M\",\n          \"XL\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```
df.describe()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"User ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 338.2012621861894,\n
\"min\": 1.0,\n        \"max\": 1000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          50.419,\n
50.0,\n          1000.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Product ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 360.1000917722167,\n
\"min\": 1.0,\n        \"max\": 1000.0,\n
\"num_unique_values\": 6,\n        \"samples\": [\n          1000.0,\n
500.5,\n          750.25\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Price\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 336.5907173351229,\n        \"min\":
10.0,\n        \"max\": 1000.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          55.785,\n          57.0,\n          1000.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Rating\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
352.5914385671099,\n        \"min\": 1.000967235428064,\n
\"max\": 1000.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          2.9931351057620845,\n
2.984002878443823,\n          1000.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

```
df.info()
```

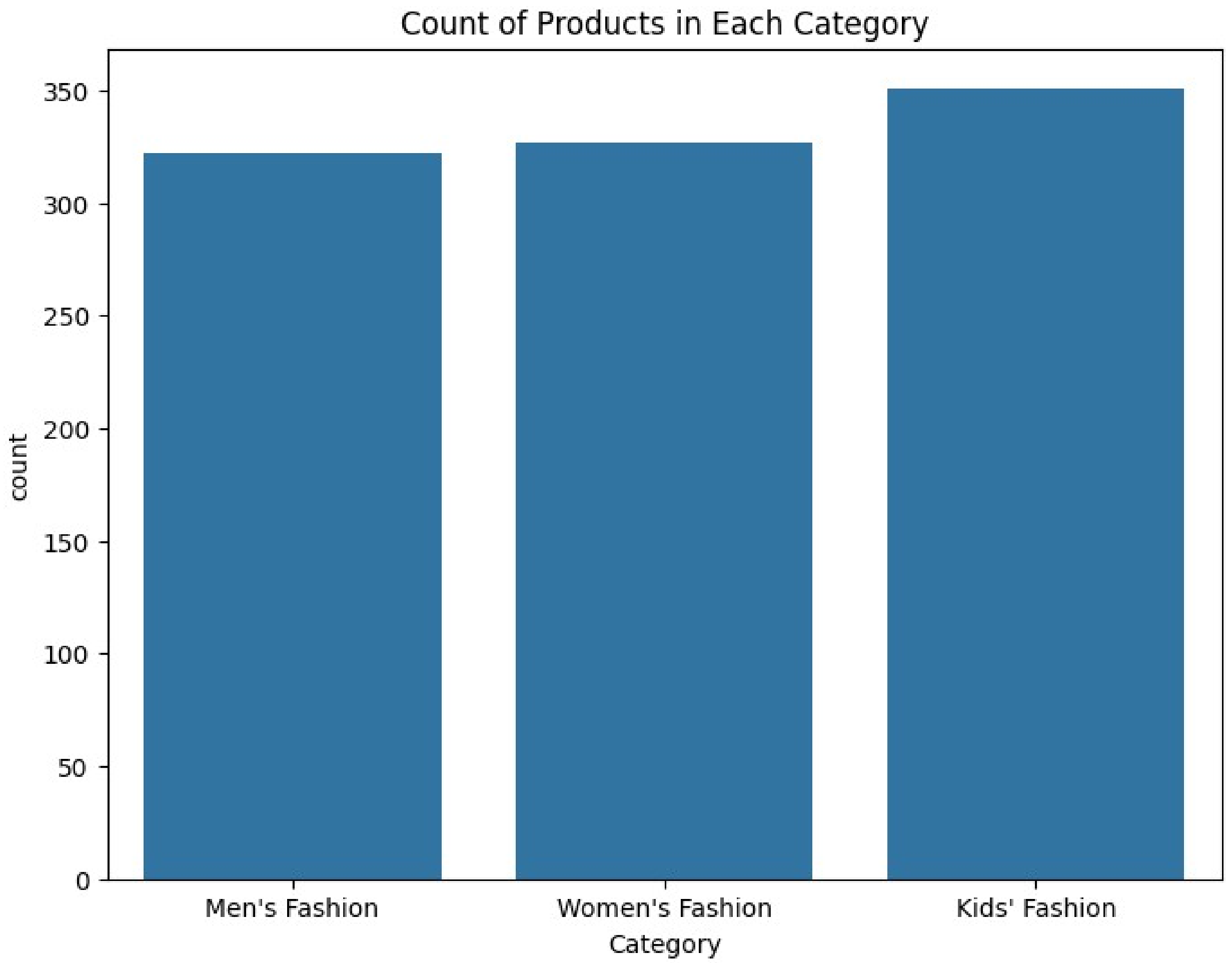
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   User ID       1000 non-null   int64  
 1   Product ID    1000 non-null   int64  
 2   Product Name  1000 non-null   object 
 3   Brand         1000 non-null   object 
 4   Category      1000 non-null   object 
 5   Price         1000 non-null   int64  
 6   Rating        1000 non-null   float64
 7   Color         1000 non-null   object 
 8   Size          1000 non-null   object 
dtypes: float64(1), int64(3), object(5)
memory usage: 70.4+ KB
```

```
df.shape
```

```
(1000, 9)
```

Distribution of Categorical Variables

```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Category')
plt.title('Count of Products in Each Category')
plt.show()
```



Men’s Fashion: This category has the highest count, with just over 300 products.

Women’s Fashion: This category has slightly fewer products compared to Men’s Fashion, with the count just below 300.

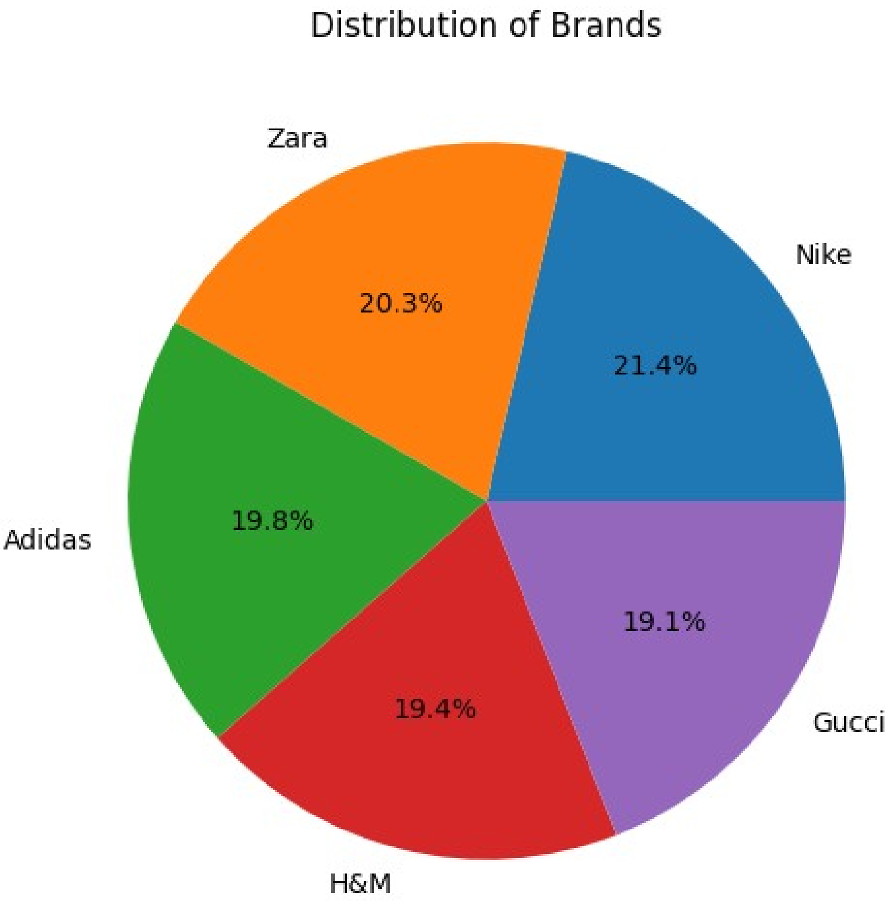
Kids’ Fashion: Similar to Men’s Fashion, this category also has just over 300 products.

Overall, Men’s and Kids’ Fashion categories have a similar number of products, while Women’s Fashion has a slightly lower count. This could be useful for inventory management or market analysis in the fashion retail sector.

Distribution of Categorical Variables

```
plt.figure(figsize=(8, 6))
df['Brand'].value_counts().plot(kind='pie', autopct='%1.1f%%')
```

```
plt.title('Distribution of Brands')
plt.ylabel('')
plt.show()
```

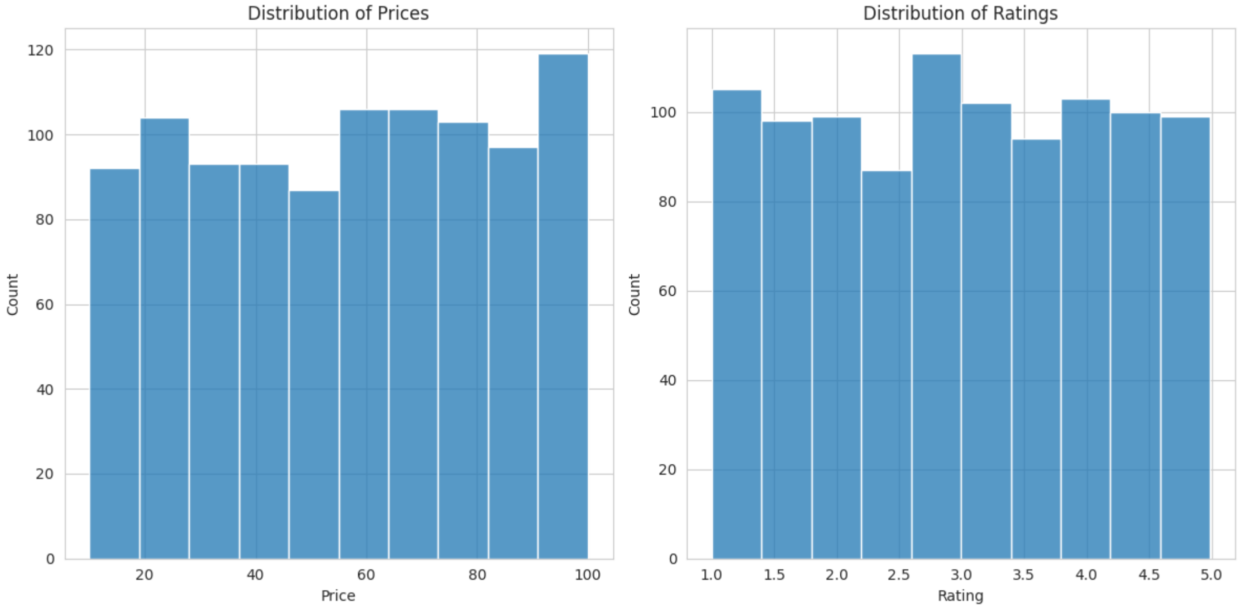


1. **Nike** holds the largest market share at 21.4%. This indicates Nike’s strong presence and popularity in the fashion market.
2. **Zara** follows closely with a 20.3% share, showing its significant influence and competitive edge.
3. **Adidas** has a 19.8% share, highlighting its robust position in the market.
4. **H&M** accounts for 19.4%, reflecting its substantial market presence.
5. **Gucci** has the smallest share at 19.1%, but still maintains a notable portion of the market.

```
sns.set_style('whitegrid')
fig, axes = plt.subplots(1, 2, figsize=(12, 6))
```

```
# Plot distribution of prices
sns.histplot(data=df, x='Price', bins=10, ax=axes[0])
axes[0].set_title('Distribution of Prices')

# Plot distribution of ratings
sns.histplot(data=df, x='Rating', bins=10, ax=axes[1])
axes[1].set_title('Distribution of Ratings')
plt.tight_layout()
plt.show()
```



A. **Distribution of Prices**

Right-Skewed Distribution: The price distribution is right-skewed, meaning there are more products at lower price points. This suggests that lower-priced items are more common in the dataset.

High Frequency at Lower Prices: The highest counts are concentrated at the lower end of the price range, indicating that most products are priced below a certain threshold.

B. **Distribution of Ratings**

Uniform Distribution with a Peak: The ratings distribution is relatively uniform, with a slight increase in frequency at the highest rating value (5.0). This could indicate that many products receive high ratings, possibly reflecting customer satisfaction or rating inflation.

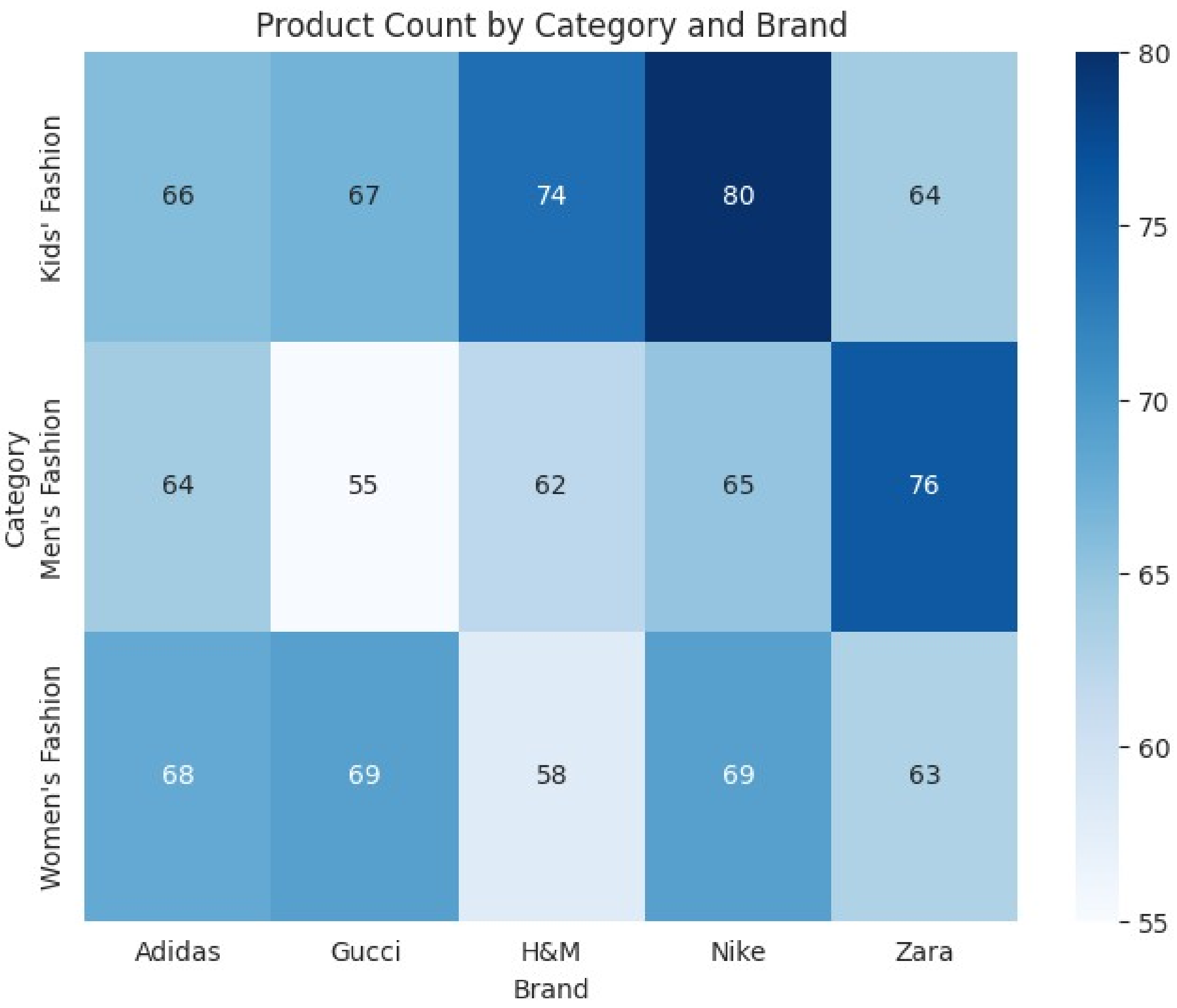
Consistent Counts Across Ratings: The counts are fairly consistent across different rating values, suggesting a balanced spread of ratings with a preference for higher ratings.

These insights can help understand consumer preferences and pricing strategies. For example, the concentration of lower-priced items might indicate a market strategy targeting budget-conscious consumers, while the high ratings could reflect positive customer experiences.

Multivariate Analysis

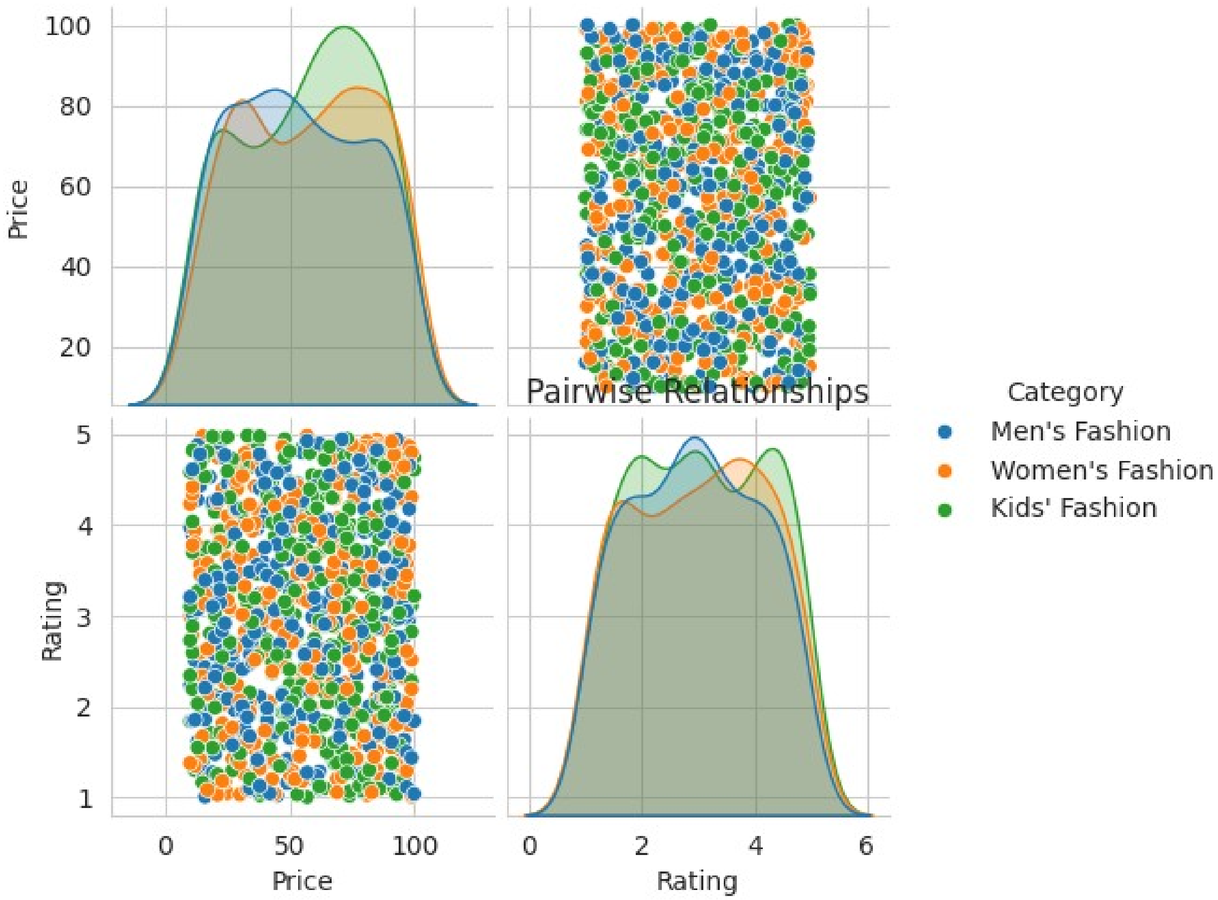
```
plt.figure(figsize=(8, 6))
category_brand_counts = df.groupby(['Category',
'Brand']).size().unstack()
sns.heatmap(data=category_brand_counts, cmap='Blues', annot=True,
fmt='g')
plt.title('Product Count by Category and Brand')
plt.show()

plt.figure(figsize=(8, 6))
sns.pairplot(df[['Price', 'Rating', 'Category']], hue='Category')
plt.title('Pairwise Relationships')
plt.show()
```



Product Count by Category and Brand

```
<Figure size 800x600 with 0 Axes>
```



I.Heatmap Insights

1.**Nike:**

Men's Fashion: Highest count at 80 products.

Kids' Fashion: Also high with 74 products.

Women's Fashion: Moderate count.

2.**Adidas**:

Consistent Counts: Similar product counts across all categories, indicating a balanced inventory.

3.**Gucci**:

Women's Fashion: Highest count at 69 products.

Lower Counts: In Kids' and Men's categories.

4.**H&M**:

Lower Counts: Across all categories compared to other brands, suggesting a smaller inventory.

5.**Zara**:

Women’s Fashion: Higher count at 63 products.

Kids’ Fashion: Slightly higher than Women’s Fashion.

Insights:

Nike and Adidas have strong presences in multiple categories, with Nike leading in Men’s and Kids’ Fashion. Gucci focuses more on Women’s Fashion, while H&M has a smaller overall inventory. Zara shows a balanced approach with a slight emphasis on Women’s and Kids’ Fashion.

These insights can help understand the inventory strategies of these brands and their focus areas in different fashion categories.

II.Pairplot

Price Distribution:

Men’s Fashion: Prices peak around 20 units, indicating a concentration of products at this price point.

Women’s Fashion: Prices are more widely distributed but also show a peak around 20 units.

Kids’ Fashion: Prices are centered around 10 units, suggesting lower price points for this category.

III.Scatter Plots:

Dense Clustering at Lower Prices: Across all categories, there is a high density of products at lower price points, indicating that most items are priced lower.

Higher-Priced Items: Less common but present up to 100 units for Men’s and Women’s Fashion, while Kids’ Fashion has fewer high-priced items.

Ratings Consistency: Ratings are fairly consistent across different price ranges for all categories. This suggests that price does not significantly impact the ratings given to items.

Insights:

Men’s and Women’s Fashion: Both categories have a significant number of products priced around 20 units, but Women’s Fashion shows a wider range of prices.

Kids’ Fashion: Generally lower-priced, with most items around 10 units.

Consumer Behavior: The consistent ratings across price ranges might indicate that consumers are equally satisfied with products regardless of their price.

These insights can help understand pricing strategies and consumer preferences across different fashion categories.

Hybrid Recommendation System

```
# Create the Surprise dataset
reader = Reader(rating_scale=(1, 5))
```

```
data = Dataset.load_from_df(df[['User ID', 'Brand', 'Rating']],
reader)

# Split the data into training and testing sets
trainset, testset = train_test_split(data, test_size=0.2)

# Collaborative Filtering using SVD
algo_cf = SVD()
algo_cf.fit(trainset)

# Brand-Based Recommendation System
def brand_recommendation(user_id, num_recommendations):
    # Collaborative Filtering predictions
    cf_predictions = algo_cf.test(testset)

    # Filter predictions based on user ID and brand
    user_brand_predictions = [pred for pred in cf_predictions if
pred.uid == user_id and pred.iid in df[df['User ID'] == user_id]
['Brand'].unique()]

    # Sort the predictions by rating in descending order
    user_brand_predictions.sort(key=lambda x: x.est, reverse=True)

    # Get the top N recommendations
    top_recommendations = user_brand_predictions[:num_recommendations]

    return top_recommendations

# Example usage: Get brand recommendations for User ID 97, top 3
recommendations
user_id = 6
num_recommendations = 3
recommendations = brand_recommendation(user_id, num_recommendations)

# Print the recommendations
print(f"Brand Recommendations for User ID {user_id}:")
for recommendation in recommendations:
    brand = recommendation.iid
    rating = recommendation.est
    print(f"Brand: {brand}, Rating: {rating}")
```

```
Brand Recommendations for User ID 6:
Brand: H&M, Rating: 2.706136277360802
Brand: Adidas, Rating: 2.6698459871160543
Brand: Adidas, Rating: 2.6698459871160543
```

Steps in the Code:

Create the Dataset:

The Surprise library is used to create a dataset from a DataFrame containing user IDs, brand names, and ratings. The Reader object specifies the rating scale (1 to 5).

Split the Data:

The dataset is split into training and testing sets with an 80-20 ratio. Collaborative Filtering using SVD: The SVD algorithm from the Surprise library is used for collaborative filtering. The model is trained on the training set.

Brand-Based Recommendation System:

A function brand_recommendation is defined to generate brand recommendations for a given user. The function makes predictions using the trained SVD model. It filters predictions to include only those brands that the user has rated. The predictions are sorted by estimated rating in descending order. The top N recommendations are returned.

Example Usage:

The function is called to get the top 3 brand recommendations for User ID 6. The recommendations are printed, showing the brand and the estimated rating.

Summary:

This code implements a hybrid recommender system that combines collaborative filtering (using SVD) with brand-based filtering. It predicts ratings for brands that a user has interacted with and recommends the top-rated brands.