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

df= pd.read_csv('amazon.csv')

df

df.info()
df.describe()

df.isnull().sum()

df.duplicated().sum()

df.drop_duplicates(inplace=True)

df.head()

df.shape

df.columns

df.tail()

df.dtypes

df.describe()

#1.What is the average rating for each product category?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'category' and 'rating' with the actual column names

# Convert the 'rating' column to numeric, handling errors
df['rating'] = pd.to_numeric(df['rating'], errors='coerce')

# Calculate average ratings
average_ratings = df.groupby('category')['rating'].mean()
print(average_ratings)

#2.What are the top rating_count products by categor?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'rating_count' with the actual column name containing review counts
# Replace 'category' with the actual column name for product category if it's different

# Check if 'category' column exists, if not, print available columns
if 'category' not in df.columns:
    print("Available columns:", df.columns)
else:
```

```
   top_products_by_category = df.groupby('category').apply(lambda x: x.nlargest(1, 'rating_count'))
   print(top_products_by_category[['product_name', 'rating_count']])

df.columns

#3.What is the distribution of discounted prices vs. actual prices?

import pandas as pd
import matplotlib.pyplot as plt

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Convert price columns to numeric (remove any special characters)
df['discounted_price'] = pd.to_numeric(df['discounted_price'].str.replace('₹', '').replace(',', ''),
errors='coerce')
df['actual_price'] = pd.to_numeric(df['actual_price'].str.replace('₹', '').replace(',', ''), errors='coerce')

# Plot histograms
plt.figure(figsize=(10, 6))
plt.hist(df['discounted_price'], bins=20, alpha=0.7, label='Discounted Price')
plt.hist(df['actual_price'], bins=20, alpha=0.7, label='Actual Price')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.title('Distribution of Discounted vs. Actual Prices')
plt.legend()
plt.show()

#4. How does the average discount percentage vary across categories?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'discount_percentage' with the actual column name

# Remove '%' and convert to numeric
df['discount_percentage'] = df['discount_percentage'].str.rstrip('%').astype('float') / 100

average_discount_by_category = df.groupby('category')['discount_percentage'].mean()
print(average_discount_by_category)

#5.What are the most popular product names?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'rating_count' with the actual column name containing review counts

top_products = df.sort_values(by='rating_count', ascending=False).head(10)
print(top_products['product_name'])

df.columns

#6. What are the most popular product keywords?

import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer

# Assuming you have loaded your dataset into a DataFrame called 'df'
```

```
# Replace 'Product' with the actual column containing product names or descriptions

# Combine product names and descriptions (if available)
df['combined_text'] = df['product_name'] + ' ' + df['about_product']

# Create a bag-of-words representation
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df['combined_text'])

# Get the most frequent keywords
keywords = vectorizer.get_feature_names_out()

print(keywords[:10])  # Print the top 10 keywords

#8. What is the correlation between discounted_price and rating?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'discounted_price' and 'rating' with the actual column names

correlation = df['discounted_price'].corr(df['rating'])
print(f"Pearson correlation coefficient: {correlation:.4f}")

#9.What are the Top 5 categories based on the highest ratings?

import pandas as pd

# Assuming you have loaded your dataset into a DataFrame called 'df'
# Replace 'Rating' with the actual column name containing ratings

average_ratings = df.groupby('category')['rating'].mean()
top_categories = average_ratings.nlargest(5)
print(top_categories)

#10. Identify any potential areas for improvement or optimization based on the data analysis

"""Based on the data analysis of the amazon.csv dataset, here are some potential areas for
improvement or optimization:

Price Optimization:
Investigate the relationship between discounted prices and ratings. If higher discounts consistently
lead to better ratings, consider adjusting pricing strategies.
Monitor price elasticity to find the optimal balance between discounts and revenue.
Category-Specific Insights:
Explore category-specific trends. Some categories may have unique patterns (e.g., electronics vs.
books).
Identify high-performing categories and allocate resources accordingly.
Review Sentiment Analysis:
Perform sentiment analysis on reviews. Understand customer sentiments associated with specific
products or categories.
Address negative sentiments promptly to improve overall customer satisfaction.
Product Descriptions and Keywords:
Enhance product descriptions with relevant keywords. Optimize content for search engines and user
understanding.
Use natural language processing (NLP) techniques to extract meaningful keywords.
Inventory Management:
```

Analyze product availability and stock levels. Avoid stockouts for popular products.
Optimize inventory turnover and reduce carrying costs."""

SPOTIFY DATASET EDA ASSINGMENT

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

df= pd.read_csv('spotify.csv')

df.head()

df.info()

df.describe()

df.isnull().sum()

df.duplicated().sum()

df.drop_duplicates(inplace=True)

df.columns
```

#1.Load the dataframe and ensure data quality by checking for missing values and duplicate rows. Handle missing values and remove duplicate rows if necessarydf.isnull().sum()

```
import pandas as pd

# Load the dataset
df = pd.read_csv('spotify.csv')

# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:")
print(missing_values)

# Check for duplicate rows
duplicate_rows = df.duplicated().sum()
print("Duplicate Rows:", duplicate_rows)
```

#2.What is the distribution of popularity among the tracks in the dataset? Visualize it using a histogram.

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")
```

```
# Create a histogram
plt.figure(figsize=(10, 6))
plt.hist(spotify_df["Popularity"], bins=20, color="skyblue", edgecolor="black")
plt.xlabel("Popularity")
plt.ylabel("Frequency")
plt.title("Popularity Distribution of Spotify Tracks")
plt.grid(axis="y")
plt.show()
```

#3.Is there any relationship between the popularity and the duration of tracks? Explore this using a scatter plot.

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Check the actual column names in your DataFrame
print(spotify_df.columns)

# Create a scatter plot, adjusting the column names if necessary
plt.figure(figsize=(10, 6))
# Replace 'Duration_ms' and 'popularity' with the correct column names from the output above
plt.scatter(spotify_df["Duration (ms)"], spotify_df["Popularity"], alpha=0.5, color="purple") # Fixed column names based on output of spotify_df.columns
plt.xlabel("Duration (ms)")
plt.ylabel("Popularity")
plt.title("Popularity vs. Duration of Spotify Tracks")
plt.grid()
plt.show()

df.columns
```

#4.Which artist has the highest number of tracks in the dataset? Display the count of tracks for each artistusing a countplot

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

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Group by artist and count tracks
artist_track_counts = spotify_df.groupby("Artist")["Track Name"].count().reset_index()

# Create a countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=artist_track_counts, x="Artist", order=artist_track_counts.sort_values(by="Track Name", ascending=False)["Artist"])
plt.xlabel("Artist")
plt.ylabel("Number of Tracks")
plt.title("Number of Tracks per Artist in Spotify Dataset")
plt.xticks(rotation=90)
```

```
plt.show()

df.columns
```

#5.What are the top 5 least popular tracks in the dataset? Provide the artist name and track name for each.

```
import pandas as pd

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Sort by popularity in ascending order
sorted_df = spotify_df.sort_values(by="Popularity")

# Get the bottom 5 tracks
least_popular_tracks = sorted_df.head(5)[["Artist", "Track Name"]]

print(least_popular_tracks)
```

#6.Among the top 5 most popular artists, which artist has the highest popularity on average? Calculate and display the average popularity for each artist.

```
import pandas as pd

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Group by artist and calculate average popularity
artist_avg_popularity = spotify_df.groupby("Artist")["Popularity"].mean()

# Display the top artist with the highest average popularity
top_artist = artist_avg_popularity.idxmax()
avg_popularity = artist_avg_popularity.max()

print(f"The artist with the highest average popularity is {top_artist} (Avg Popularity:
{avg_popularity:.2f})")
```

#7.For the top 5 most popular artists, what are their most popular tracks? List the track name for each artist

```
import pandas as pd

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Group by artist and sort by popularity
sorted_df = spotify_df.sort_values(by="Popularity", ascending=False)
top_tracks_per_artist = sorted_df.groupby("Artist").first()["Track Name"]

print(top_tracks_per_artist)
```

#8.Visualize relationships between multiple numerical variables simultaneously using a pair plot.

```
import seaborn as sns
```

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset using pandas (replace with your actual file path)
df = pd.read_csv("spotify.csv")

# Create a pair plot
sns.pairplot(df)
plt.show()
```

#9.Does the duration of tracks vary significantly across different artists?Explore this visually using a box plot or violin plot.

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

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Create a violin plot
plt.figure(figsize=(10, 6))
sns.violinplot(data=spotify_df, x="Artist", y="Duration (ms)", inner="quartiles") # Changed
'Duration_ms' to 'Duration (ms)'
plt.xlabel("Artist")
plt.ylabel("Duration (ms)")
plt.title("Track Duration Distribution by Artist")
plt.xticks(rotation=90)
plt.show()
```

#10.How does the distribution of track popularity vary for different artists?Visualize this using a swarm plot or a violin plot

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

# Load the dataset (replace with your actual file path)
spotify_df = pd.read_csv("spotify.csv")

# Create a violin plot
plt.figure(figsize=(10, 6))
sns.violinplot(data=spotify_df, x="Artist", y="Popularity", inner="quartiles")
plt.xlabel("Artist")
plt.ylabel("Popularity")
plt.title("Track Popularity Distribution by Artist")
plt.xticks(rotation=90)
plt.show()
```

complate