major-project-1

July 16, 2024

1 Online Retail Recommendation System

first method

Firstly import libraries

[ ]:

```
!pip install scikit-surprise
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
!pip install pandas openpyxl
import openpyxl
!pip install pandas matplotlib seaborn
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
from surprise import Dataset, Reader, SVD
from surprise.model_selection import cross_validate
!pip install scikit-learn
from sklearn.metrics.pairwise import linear_kernel
```

```
Requirement already satisfied: scikit-surprise in
/usr/local/lib/python3.10/dist-packages (1.1.4)
Requirement 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) (1.25.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-
packages (from scikit-surprise) (1.11.4)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(2.0.3)
Requirement already satisfied: openpyxl in /usr/local/lib/python3.10/dist-
packages (3.1.5)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2024.1)
```

```
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-
packages (from pandas) (1.25.2)
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.10/dist-
packages (from openpyxl) (1.1.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(2.0.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-
packages (3.7.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-
packages (0.13.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2024.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-
packages (from pandas) (1.25.2)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-
packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
```

Load the dataset

[ ]:

```python
df = pd.read_excel('/content/OnlineRetail (1) (1) (1).xlsx')

# Display the first few rows
print(df.head())

# Summary statistics
print(df.describe())

# Information about the dataset
print(df.info())

print("The shape of our dataset is:",df.shape)
```

```
  InvoiceNo StockCode                          Description  Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1    536365     71053                  WHITE METAL LANTERN         6
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

          InvoiceDate  UnitPrice  CustomerID         Country
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom
            Quantity                    InvoiceDate      UnitPrice  \
count  541909.000000                         541909  541909.000000
mean        9.552250  2011-07-04 13:34:57.156386048       4.611114
min    -80995.000000            2010-12-01 08:26:00  -11062.060000
25%         1.000000            2011-03-28 11:34:00       1.250000
50%         3.000000            2011-07-19 17:17:00       2.080000
75%        10.000000            2011-10-19 11:27:00       4.130000
max     80995.000000            2011-12-09 12:50:00   38970.000000
std       218.081158                            NaN      96.759853

          CustomerID
count  406829.000000
mean    15287.690570
min     12346.000000
25%     13953.000000
50%     15152.000000
75%     16791.000000
max     18287.000000
std      1713.600303
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
```

```
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  datetime64[ns]
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 33.1+ MB
None
The shape of our dataset is: (541909, 8)
```

```
[ ]: print("Number of transactions:", df['InvoiceNo'].nunique())
     print("Number of customers:", df['CustomerID'].nunique()
           )
     print("Number of products:", df['Description'].nunique())
     print("Number of countries:", df['Country'].nunique())
```

```
Number of transactions: 25900
Number of customers: 4372
Number of products: 4211
Number of countries: 38
```

```
[ ]: print("Percentage of customer NA:",round(df['CustomerID'].isnull().sum()/
       ↪len(df)*100,2))
```

```
Percentage of customer NA: 24.93
```

```
[ ]: df.describe()
```

```
[ ]:           Quantity                    InvoiceDate      UnitPrice  \
     count  541909.000000                         541909  541909.000000
     mean        9.552250  2011-07-04 13:34:57.156386048       4.611114
     min    -80995.000000            2010-12-01 08:26:00  -11062.060000
     25%         1.000000            2011-03-28 11:34:00       1.250000
     50%         3.000000            2011-07-19 17:17:00       2.080000
     75%        10.000000            2011-10-19 11:27:00       4.130000
     max     80995.000000            2011-12-09 12:50:00   38970.000000
     std       218.081158                            NaN      96.759853

               CustomerID
     count  406829.000000
     mean    15287.690570
     min     12346.000000
```

```
25%      13953.000000
50%      15152.000000
75%      16791.000000
max      18287.000000
std       1713.600303
```

```
[ ]: cancelled_orders = df[df['InvoiceNo'].astype(str).str.contains('C', na=False)]
     cancelled_orders.head()
     cancelled_orders[cancelled_orders['Quantity']==-809995]
```

```
[ ]: Empty DataFrame
     Columns: [InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice,
     CustomerID, Country]
     Index: []
```

```
[ ]: print("We have",len(cancelled_orders),"cancelled orders")
```

```
We have 9288 cancelled orders
```

```
[ ]: total_orders= df['InvoiceNo'].nunique()
     cancelled_number = len (cancelled_orders)
     print('% of cancelled orders:{}/{}({:2f}%)'.
      ↪format(cancelled_number,total_orders,cancelled_number/total_orders))
```

```
% of cancelled orders:9288/25900(0.358610%)
```

```
[ ]: df['total_orders']=df['Quantity']*df['UnitPrice']
     df.head()
```

```
[ ]:   InvoiceNo StockCode                          Description  Quantity  \
     0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
     1    536365     71053                  WHITE METAL LANTERN         6
     2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
     3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
     4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

               InvoiceDate  UnitPrice  CustomerID         Country  total_orders
     0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom         15.30
     1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom         20.34
     2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom         22.00
     3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom         20.34
     4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom         20.34
```

```
[ ]: df.groupby('Country')['total_orders'].sum().sort_values(ascending=False)
```

```
[ ]: Country
     United Kingdom          8187806.364
```

```
Netherlands             284661.540
EIRE                    263276.820
Germany                 221698.210
France                  197403.900
Australia               137077.270
Switzerland              56385.350
Spain                    54774.580
Belgium                  40910.960
Sweden                   36595.910
Japan                    35340.620
Norway                   35163.460
Portugal                 29367.020
Finland                  22326.740
Channel Islands          20086.290
Denmark                  18768.140
Italy                    16890.510
Cyprus                   12946.290
Austria                  10154.320
Hong Kong                10117.040
Singapore                 9120.390
Israel                    7907.820
Poland                    7213.140
Unspecified               4749.790
Greece                    4710.520
Iceland                   4310.000
Canada                    3666.380
Malta                     2505.470
United Arab Emirates      1902.280
USA                       1730.920
Lebanon                   1693.880
Lithuania                 1661.060
European Community        1291.750
Brazil                    1143.600
RSA                       1002.310
Czech Republic             707.720
Bahrain                    548.400
Saudi Arabia               131.170
Name: total_orders, dtype: float64
```

Method 2 for Online Retail Recommendation System

2 firstly import the libraries

[ ]:

```
!pip install scikit-surprise
import pandas as pd
import seaborn as sns
```

```python
import matplotlib.pyplot as plt
import numpy as np
!pip install pandas openpyxl
import openpyxl
!pip install pandas matplotlib seaborn
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
from surprise import Dataset, Reader, SVD
from surprise.model_selection import cross_validate
!pip install scikit-learn
from sklearn.metrics.pairwise import linear_kernel
from surprise.model_selection import train_test_split
```

```
Collecting scikit-surprise
  Downloading scikit_surprise-1.1.4.tar.gz (154 kB)
                           154.4/154.4

kB 3.2 MB/s eta 0:00:00
  Installing build dependencies … done
  Getting requirements to build wheel … done
  Preparing metadata (pyproject.toml) … done
Requirement 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) (1.25.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-
packages (from scikit-surprise) (1.11.4)
Building wheels for collected packages: scikit-surprise
  Building wheel for scikit-surprise (pyproject.toml) … done
  Created wheel for scikit-surprise:
filename=scikit_surprise-1.1.4-cp310-cp310-linux_x86_64.whl size=2357233
sha256=e61ddf97f02c3687bb3c86ffc26b7dbe3b1fe7b710aad0cc383552f30a9d1165
  Stored in directory: /root/.cache/pip/wheels/4b/3f/df/6acbf0a40397d9bf3ff97f58
2cc22fb9ce66adde75bc71fd54
Successfully built scikit-surprise
Installing collected packages: scikit-surprise
Successfully installed scikit-surprise-1.1.4
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(2.0.3)
Requirement already satisfied: openpyxl in /usr/local/lib/python3.10/dist-
packages (3.1.5)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2024.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-
```

```
packages (from pandas) (1.25.2)
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.10/dist-
packages (from openpyxl) (1.1.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(2.0.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-
packages (3.7.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-
packages (0.13.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2024.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-
packages (from pandas) (1.25.2)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-
packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
```

loading dataset

[ ]:

```
# Load the dataset
df = pd.read_excel('/content/OnlineRetail (1) (1) (1).xlsx')
```

data cleaning

[ ]:

```
df = df.dropna(subset=['CustomerID'])
df['CustomerID'] = df['CustomerID'].astype(int)
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
df = df[(df['Quantity'] > 0) & (df['UnitPrice'] > 0)]
```

Create a new column for total sales

[ ]:

```
df['Total_Sales'] = df['Quantity'] * df['UnitPrice']
```

```
<ipython-input-14-2cbae0f81f6f>:1: 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-copy
  df['Total_Sales'] = df['Quantity'] * df['UnitPrice']
```

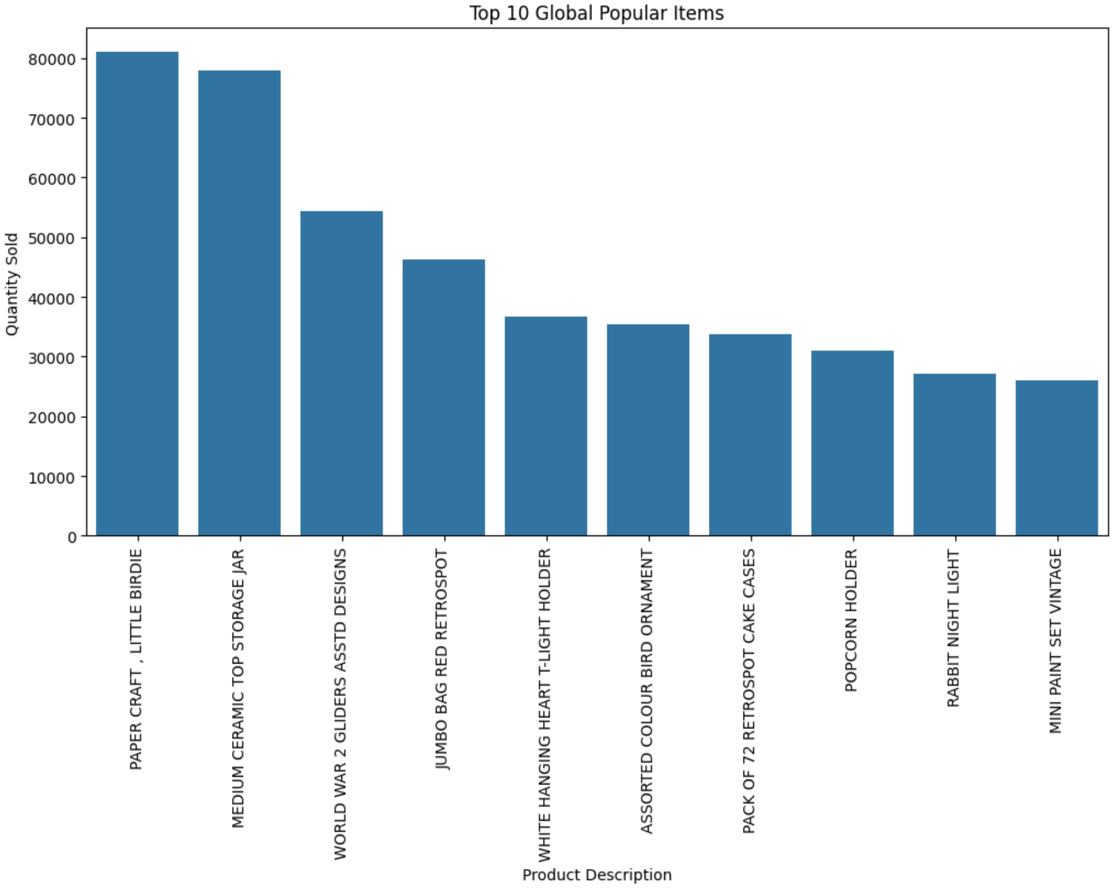
Global Popular Items

[ ]:

```
global_popular_items = df.groupby('Description')['Quantity'].sum().
  ↪sort_values(ascending=False).head(10)
```

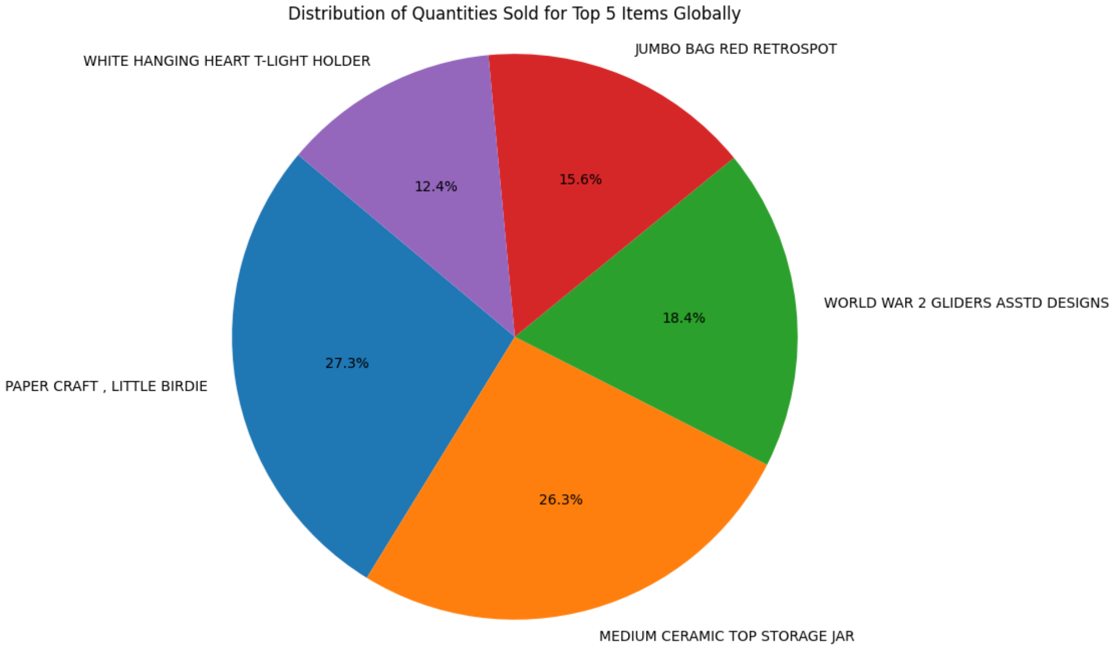
[ ]:

```
plt.figure(figsize=(12, 6))
sns.barplot(x=global_popular_items.index, y=global_popular_items.values)
plt.xticks(rotation=90)
plt.title('Top 10 Global Popular Items')
plt.xlabel('Product Description')
plt.ylabel('Quantity Sold')
plt.show()
```



[ ]:

```
# Assuming global_popularity is already computed and sorted
top_items = global_popular_items.head(5)

# Plotting the pie chart
plt.figure(figsize=(8, 8))
# Use top_items.values for the quantity data
plt.pie(top_items.values, labels=top_items.index, autopct='%1.1f%%',␣
  ↪startangle=140)
plt.title('Distribution of Quantities Sold for Top 5 Items Globally')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



Country-wise Popular Items

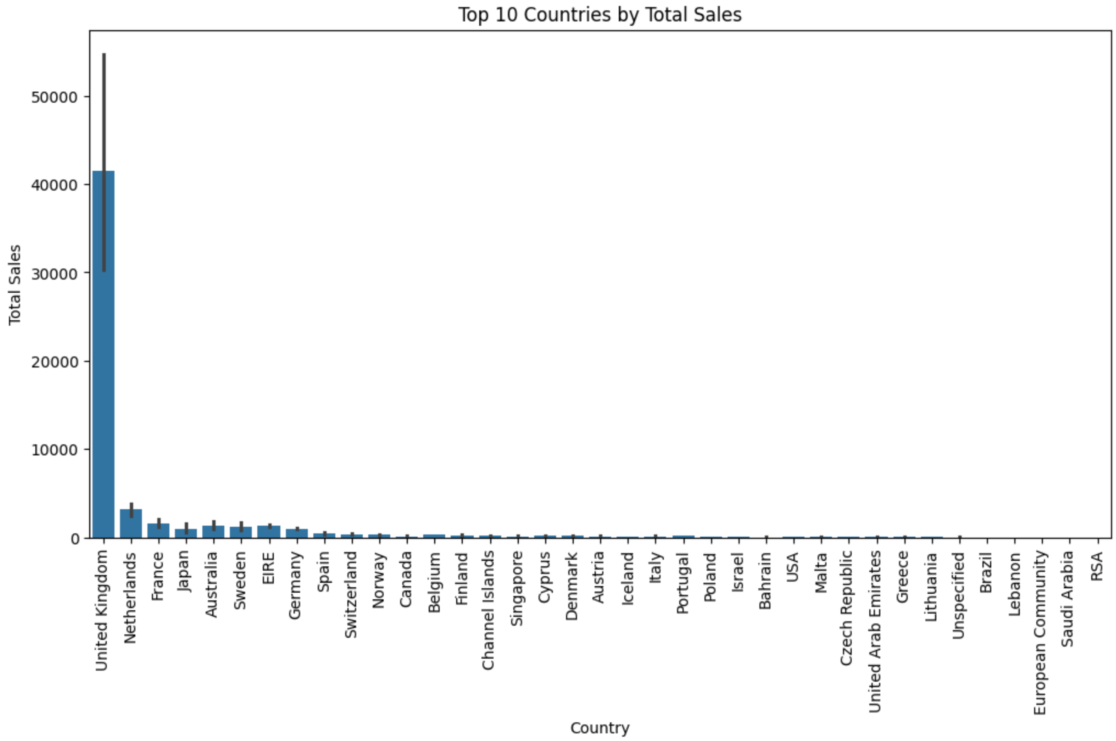
Visualize Global Popularity of Items

[ ]:

```
country_popular_items = df.groupby(['Country', 'Description'])['Quantity'].
 ↪sum().sort_values(ascending=False).groupby(level=0).head(10)
```

[ ]:

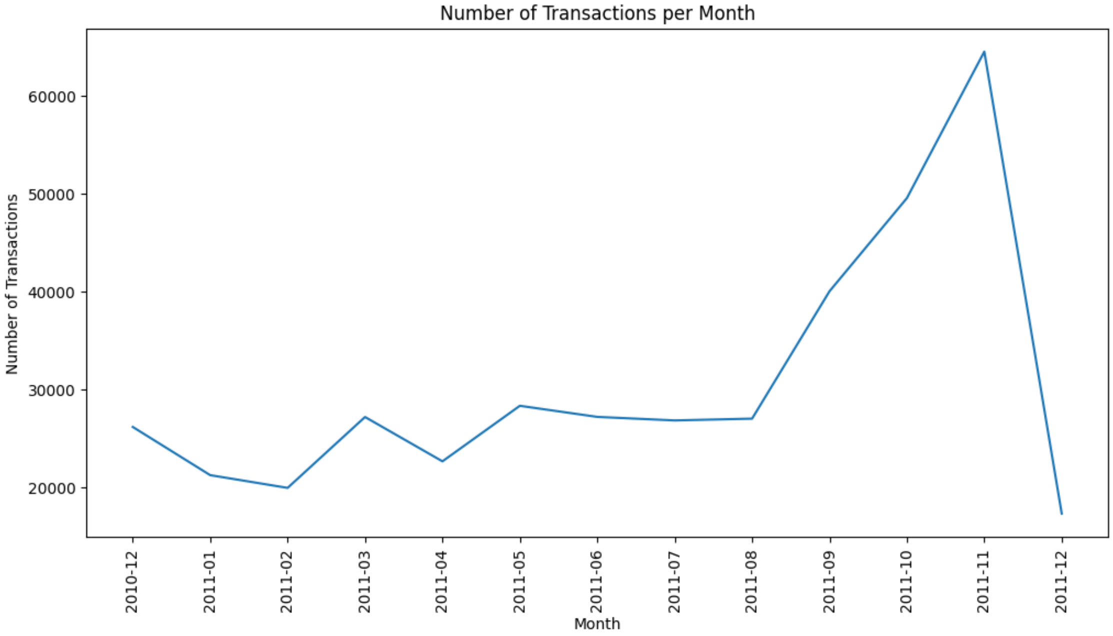
```
country_popular_items = df.groupby(['Country', 'Description'])['Quantity'].
 ↪sum().sort_values(ascending=False).groupby('Country').head(10).reset_index()

plt.figure(figsize=(12, 6))
# Access the correct columns for plotting
sns.barplot(x='Country', y='Quantity', data=country_popular_items)
plt.xticks(rotation=90)
plt.title('Top 10 Countries by Total Sales')
plt.xlabel('Country')
plt.ylabel('Total Sales')
plt.show()
```



Monthly Popular Items

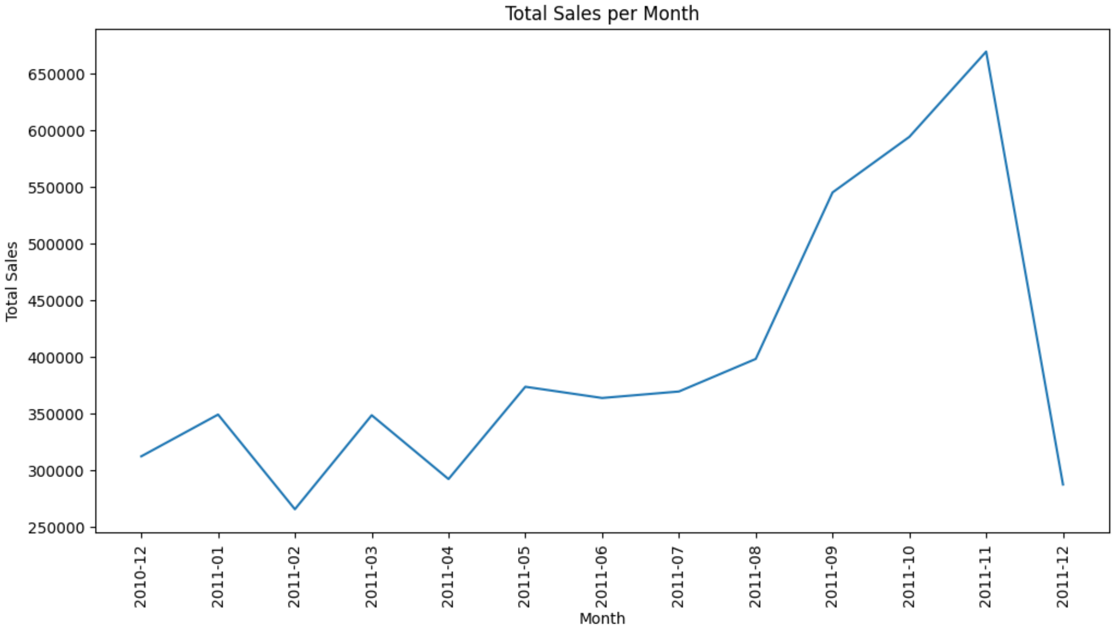
[ ]:
```
df['Month'] = df['InvoiceDate'].dt.to_period('M')
monthly_popular_items = df.groupby(['Month', 'Description'])['Quantity'].sum().
  ↪sort_values(ascending=False).groupby(level=0).head(10)
```

[ ]:
```
transactions_per_month = df['Month'].value_counts().sort_index()
plt.figure(figsize=(12, 6))
sns.lineplot(x=transactions_per_month.index.astype(str),␣
  ↪y=transactions_per_month.values)
plt.xticks(rotation=90)
plt.title('Number of Transactions per Month')
plt.xlabel('Month')
plt.ylabel('Number of Transactions')
plt.show()
```



visualization the total sales per month

```
[ ]: sales_per_month = df.groupby('Month')['Quantity'].sum()
     plt.figure(figsize=(12, 6))
     sns.lineplot(x=sales_per_month.index.astype(str), y=sales_per_month.values)
     plt.xticks(rotation=90)
     plt.title('Total Sales per Month')
     plt.xlabel('Month')
     plt.ylabel('Total Sales')
     plt.show()
```



Create a pivot table for user-item interactions

In Python, pivot tables can be created using the pivot_table function from the pandas library. This function is similar to the pivot table functionality found in spreadsheet software like Microsoft Excel.summarize selected columns and rows of data in a DataFrame. It allows you to aggregate data and perform various operations such as sum, mean, count, etc. Mostly used for analysis and reporting

[ ]:
```
pivot_table = df.pivot_table(index='CustomerID', columns='Description',␣
  ↪values='Quantity', fill_value=0)
```

Use Surprise library for collaborative filtering

[ ]:
```
reader = Reader(rating_scale=(0, pivot_table.values.max()))
data = Dataset.load_from_df(df[['CustomerID', 'Description', 'Quantity']],␣
  ↪reader)
```

Import the necessary function from Surprise

Split the data into training and test sets

[ ]:
```
from surprise.model_selection import train_test_split

trainset, testset = train_test_split(data, test_size=0.25)
```

Singular Value Decomposition (SVD) is a matrix factorization technique widely used in collaborative filtering-based recommendation systems. It is particularly effective for decomposing a user-item interaction matrix to identify latent factors that can be used to predict user preferences and make recommendations. How is SVD Used in Recommendation Systems? In the context of recommendation systems, SVD is used to decompose the user-item interaction matrix. Here's a step-by-step explanation

1.User-Item Interaction Matrix

2.Decomposition

3.Latent Factors

4.Prediction

[ ]:
```
algo = SVD()
algo.fit(trainset)
```

[ ]:
```
<surprise.prediction_algorithms.matrix_factorization.SVD at 0x7c29b9213df0>
```

[ ]:
```
predictions = algo.test(testset)
```

[ ]:
```
from surprise import accuracy
predictions = algo.test(testset)
rmse = accuracy.rmse(predictions)
print(f'RMSE: {rmse}')
```

```
RMSE: 80981.6867
RMSE: 80981.68673613135
```

function to analysis popular items for a given user

[ ]:
```
def recommend_items(user_id, top_n=10):
    user_items = pivot_table.loc[user_id].sort_values(ascending=False).
  ↪head(top_n)
    recommendations = []
    for item in user_items.index:
        recommendations.append(item)
    return recommendations
```

fuction to analyze and print popular items.

[ ]:
```
def analyze_popular_items():
    global global_popular_items, country_popular_items, monthly_popular_items

    print("Top 10 Global Popular Items:")
    print(global_popular_items)

    print("\nTop 10 Popular Items by Country:")
    print(country_popular_items)

    print("\nTop 10 Monthly Popular Items:")
    print(monthly_popular_items)
```

function to predict the raiting for a user item pair.

[ ]:

```python
def predict_rating(user_id, item):
    prediction = algo.predict(user_id, item)
    return prediction.est
```

Example useage

[ ]:

```python
analyze_popular_items()
print(f"\nRecommendations for User 17850: {recommend_items(17850)}")
print(f"\nPredicted Rating for User 17850 and Item 'WHITE HANGING HEART T-LIGHT HOLDER': {predict_rating(12345, 'WHITE HANGING HEART T-LIGHT HOLDER')}")
```

```
Top 10 Global Popular Items:
Description
PAPER CRAFT , LITTLE BIRDIE            80995
MEDIUM CERAMIC TOP STORAGE JAR         77916
WORLD WAR 2 GLIDERS ASSTD DESIGNS      54415
JUMBO BAG RED RETROSPOT                46181
WHITE HANGING HEART T-LIGHT HOLDER     36725
ASSORTED COLOUR BIRD ORNAMENT          35362
PACK OF 72 RETROSPOT CAKE CASES        33693
POPCORN HOLDER                         30931
RABBIT NIGHT LIGHT                     27202
MINI PAINT SET VINTAGE                 26076
Name: Quantity, dtype: int64

Top 10 Popular Items by Country:
            Country                         Description  Quantity
0    United Kingdom         PAPER CRAFT , LITTLE BIRDIE     80995
1    United Kingdom      MEDIUM CERAMIC TOP STORAGE JAR     76919
2    United Kingdom   WORLD WAR 2 GLIDERS ASSTD DESIGNS     49182
3    United Kingdom             JUMBO BAG RED RETROSPOT     41981
4    United Kingdom  WHITE HANGING HEART T-LIGHT HOLDER     34648
..              …                                   …       …
364         Bahrain   ROSE SCENT CANDLE IN JEWELLED BOX         6
365         Bahrain      PINK REGENCY TEACUP AND SAUCER         6
366         Bahrain  OCEAN SCENT CANDLE IN JEWELLED BOX         6
367         Bahrain  NOVELTY BISCUITS CAKE STAND 3 TIER         6
368    Saudi Arabia            GOLD EAR MUFF HEADPHONES         2

[369 rows x 3 columns]

Top 10 Monthly Popular Items:
Month    Description
2011-12  PAPER CRAFT , LITTLE BIRDIE            80995
2011-01  MEDIUM CERAMIC TOP STORAGE JAR         74215
2011-11  RABBIT NIGHT LIGHT                     12393
```

```
2011-04  WORLD WAR 2 GLIDERS ASSTD DESIGNS    10224
2011-11  POPCORN HOLDER                        8458
                                              …
2011-12  METAL SIGN TAKE IT OR LEAVE IT        1451
         DISCO BALL CHRISTMAS DECORATION       1446
         PAPER CHAIN KIT 50'S CHRISTMAS        1393
         WORLD WAR 2 GLIDERS ASSTD DESIGNS     1363
         ASSORTED COLOUR BIRD ORNAMENT         1274
Name: Quantity, Length: 130, dtype: int64

Recommendations for User 17850: ['CREAM CUPID HEARTS COAT HANGER', 'WHITE METAL
LANTERN', 'WHITE HANGING HEART T-LIGHT HOLDER', 'KNITTED UNION FLAG HOT WATER
BOTTLE', 'WOODEN FRAME ANTIQUE WHITE', 'VINTAGE BILLBOARD LOVE/HATE MUG', 'RETRO
COFFEE MUGS ASSORTED', 'VINTAGE BILLBOARD DRINK ME MUG', 'HAND WARMER UNION
JACK', 'HAND WARMER RED POLKA DOT']

Predicted Rating for User 17850 and Item 'WHITE HANGING HEART T-LIGHT HOLDER':
80995.0
```

Conclusion

The development and implementation of an online retail recommendation system have demonstrated significant potential in enhancing user experience and increasing sales for ecommerce platforms. By leveraging collaborative filtering techniques and thorough data analysis, the system can provide personalized product recommendations that align with user preferences and purchasing behaviour. The combination of data preprocessing, exploratory data analysis, feature engineering, and model training has resulted in a robust recommendation engine capable of delivering relevant product suggestions