ONLINE RETAIL RECOMMENDATION SYSTEM

By

Kusan Chakraborty

Department Name: Data Science Intern

Date: 15-08-2025

# Abstract

This project implements an Online Retail Recommendation System using the Online Retail dataset. The system applies three complementary recommendation strategies — Popularity-based ranking, Item-Item Collaborative Filtering, and Frequently-Bought-Together analysis — to suggest relevant products to customers. The methodology is designed to be memory-efficient and scalable to the full dataset. Results demonstrate a reasonable Hit Rate@10, and the approach can be integrated into an e-commerce platform for real-time personalized suggestions.

# Introduction & Objective

The Online Retail dataset contains transactional data for a UK-based online store. The objective of this project is to develop a recommendation system capable of suggesting products to customers based on their past purchase behavior and general purchasing trends.

Specific goals:

1. Preprocess and clean the dataset for modeling.

2. Implement multiple recommendation strategies to balance accuracy and diversity.

3. Evaluate the system on historical data using a time-based split.

4. Ensure scalability so the system can operate on the full dataset.

# Methodology

Dataset: Online Retail (Excel format).

Key columns used: InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country.

Preprocessing steps:

- Drop rows with missing CustomerID.

- Remove cancelled transactions (InvoiceNo starts with 'C').

- Keep only positive Quantity and UnitPrice.

- Parse InvoiceDate and compute TotalPrice.

Models implemented:

1) Popularity-based: ranks items by number of unique buyers, quantity sold, and revenue.

2) Item-Item Collaborative Filtering: computes cosine similarity between items based on co-occurrence in user baskets.

3) Frequently-Bought-Together: identifies items that often appear together in the same invoice.

Train/Validation Split: last 30 days as validation (fallback to 80/20 time split).

# Results & Evaluation

Loading dataset from OnlineRetail (1).xlsx ...

Data summary:

rows: 397884

unique\_customers: 4338

unique\_items: 3665

countries: 37

date\_min: 2010-12-01 08:26:00

date\_max: 2011-12-09 12:50:00

Saved popularity CSVs.

Train rows: 332310

Valid rows: 65574

Building user-item matrix for CF (may take time on full dataset)...

Hit@10 (validation last-invoice): 25.09%

Saved sample user recommendations CSV.

Saved FBT CSV.

===== KEY INSIGHTS =====

Top 5 Most Popular Products (Global):

1. REGENCY CAKESTAND 3 TIER - 881 buyers

2. WHITE HANGING HEART T-LIGHT HOLDER - 856 buyers

3. PARTY BUNTING - 708 buyers

4. ASSORTED COLOUR BIRD ORNAMENT - 678 buyers

5. SET OF 3 CAKE TINS PANTRY DESIGN - 640 buyers

Top 3 Countries by Unique Buyers:

1. United Kingdom - 6273 buyers

2. Germany - 351 buyers

3. France - 312 buyers

Busiest Month: 11/2011 with 1664 unique buyers

Top 3 Frequently Bought Together Pairs (Example Items):

1. ROSES REGENCY TEACUP AND SAUCER, GREEN REGENCY TEACUP AND SAUCER

2. RED HANGING HEART T-LIGHT HOLDER, WOODEN PICTURE FRAME WHITE FINISH

3. SPOTTY BUNTING, WHITE HANGING HEART T-LIGHT HOLDER

All outputs written. Files:

- retail\_outputs\_popularity\_global.csv

- retail\_outputs\_popularity\_by\_country.csv

- retail\_outputs\_popularity\_by\_month.csv

- retail\_outputs\_sample\_user\_recommendations.csv

- retail\_outputs\_fbt\_recommendations.csv

- retail\_outputs\_chart\_global.png

- retail\_outputs\_chart\_country.png

- retail\_outputs\_chart\_month.png

- retail\_outputs\_chart\_sample\_recs.png

- retail\_outputs\_chart\_fbt.png

**Retail\_Outputs\_FBT\_Recommendation (CSV)**  
Item(StockCode), FBT(StockCode), FBT(Description)

22423,"[22699, 22697, 22698, 47566, 23245, 84879, '85123A', 22720, 23173, 21212, 22960, 22961, '85099B', 23170, 22727, 22666, 'POST', 21843, 23174, 22629]","['ROSES REGENCY TEACUP AND SAUCER', 'GREEN REGENCY TEACUP AND SAUCER', 'PINK REGENCY TEACUP AND SAUCER', 'PARTY BUNTING', 'SET OF 3 REGENCY CAKE TINS', 'ASSORTED COLOUR BIRD ORNAMENT', 'WHITE HANGING HEART T-LIGHT HOLDER', 'SET OF 3 CAKE TINS PANTRY DESIGN', 'REGENCY TEAPOT ROSES', 'PACK OF 72 RETROSPOT CAKE CASES', 'JAM MAKING SET WITH JARS', 'JAM MAKING SET PRINTED', 'JUMBO BAG RED RETROSPOT', 'REGENCY TEA PLATE ROSES', 'ALARM CLOCK BAKELIKE RED', 'RECIPE BOX PANTRY YELLOW DESIGN', 'POSTAGE', 'RED RETROSPOT CAKE STAND', 'REGENCY SUGAR BOWL GREEN', 'SPACEBOY LUNCH BOX']"

85123A,"[21733, 82482, 22804, 47566, 22470, 22469, 22457, 84879, '82494L', '85099B', 22423, 20725, 21754, 22383, 22720, 22384, 20728, 22382, 20727, 21755]","['RED HANGING HEART T-LIGHT HOLDER', 'WOODEN PICTURE FRAME WHITE FINISH', 'CANDLEHOLDER PINK HANGING HEART', 'PARTY BUNTING', 'HEART OF WICKER LARGE', 'HEART OF WICKER SMALL', 'NATURAL SLATE HEART CHALKBOARD', 'ASSORTED COLOUR BIRD ORNAMENT', 'WOODEN FRAME ANTIQUE WHITE', 'JUMBO BAG RED RETROSPOT', 'REGENCY CAKESTAND 3 TIER', 'LUNCH BAG RED RETROSPOT', 'HOME BUILDING BLOCK WORD', 'LUNCH BAG SUKI  DESIGN', 'SET OF 3 CAKE TINS PANTRY DESIGN', 'LUNCH BAG PINK POLKADOT', 'LUNCH BAG CARS BLUE', 'LUNCH BAG SPACEBOY DESIGN', 'LUNCH BAG  BLACK SKULL.', 'LOVE BUILDING BLOCK WORD']"

47566,"[23298, '85123A', 22423, 84879, '85099B', '47566B', 20725, 22720, 22383, 82482, 22090, 23206, '47590B', 23203, 22457, 23209, 22993, 22178, 21212, 22384]","['SPOTTY BUNTING', 'WHITE HANGING HEART T-LIGHT HOLDER', 'REGENCY CAKESTAND 3 TIER', 'ASSORTED COLOUR BIRD ORNAMENT', 'JUMBO BAG RED RETROSPOT', 'TEA TIME PARTY BUNTING', 'LUNCH BAG RED RETROSPOT', 'SET OF 3 CAKE TINS PANTRY DESIGN', 'LUNCH BAG SUKI  DESIGN', 'WOODEN PICTURE FRAME WHITE FINISH', 'PAPER BUNTING RETROSPOT', 'LUNCH BAG APPLE DESIGN', 'PINK HAPPY BIRTHDAY BUNTING', 'JUMBO BAG DOILEY PATTERNS', 'NATURAL SLATE HEART CHALKBOARD', 'LUNCH BAG DOILEY PATTERN', 'SET OF 4 PANTRY JELLY MOULDS', 'VICTORIAN GLASS HANGING T-LIGHT', 'PACK OF 72 RETROSPOT CAKE CASES', 'LUNCH BAG PINK POLKADOT']"

84879,"['85123A', 21136, 22423, 47566, 22178, 23298, 21754, 22457, 22720, 82482, 20725, 22470, 22727, '85099B', 84946, 22699, 22469, 23209, 21212, 84755]","['WHITE HANGING HEART T-LIGHT HOLDER', 'PAINTED METAL PEARS ASSORTED', 'REGENCY CAKESTAND 3 TIER', 'PARTY BUNTING', 'VICTORIAN GLASS HANGING T-LIGHT', 'SPOTTY BUNTING', 'HOME BUILDING BLOCK WORD', 'NATURAL SLATE HEART CHALKBOARD', 'SET OF 3 CAKE TINS PANTRY DESIGN', 'WOODEN PICTURE FRAME WHITE FINISH', 'LUNCH BAG RED RETROSPOT', 'HEART OF WICKER LARGE', 'ALARM CLOCK BAKELIKE RED', 'JUMBO BAG RED RETROSPOT', 'ANTIQUE SILVER TEA GLASS ETCHED', 'ROSES REGENCY TEACUP AND SAUCER', 'HEART OF WICKER SMALL', 'LUNCH BAG DOILEY PATTERN', 'PACK OF 72 RETROSPOT CAKE CASES', 'COLOUR GLASS T-LIGHT HOLDER HANGING']"

22720,"[22722, 22666, 22960, 22423, 23243, 22961, '85123A', 22993, 47566, 23245, 22624, 21212, 22907, '85099B', 20725, 84879, 20914, 22969, 23298, 22966]","['SET OF 6 SPICE TINS PANTRY DESIGN', 'RECIPE BOX PANTRY YELLOW DESIGN', 'JAM MAKING SET WITH JARS', 'REGENCY CAKESTAND 3 TIER', 'SET OF TEA COFFEE SUGAR TINS PANTRY', 'JAM MAKING SET PRINTED', 'WHITE HANGING HEART T-LIGHT HOLDER', 'SET OF 4 PANTRY JELLY MOULDS', 'PARTY BUNTING', 'SET OF 3 REGENCY CAKE TINS', 'IVORY KITCHEN SCALES', 'PACK OF 72 RETROSPOT CAKE CASES', 'PACK OF 20 NAPKINS PANTRY DESIGN', 'JUMBO BAG RED RETROSPOT', 'LUNCH BAG RED RETROSPOT', 'ASSORTED COLOUR BIRD ORNAMENT', 'SET/5 RED RETROSPOT LID GLASS BOWLS', 'HOMEMADE JAM SCENTED CANDLES', 'SPOTTY BUNTING', 'GINGERBREAD MAN COOKIE CUTTER']"

**Retail\_Outputs\_Popularity\_By\_Country (CSV)**

Country,Description,buyers,quantity,revenue

Australia,SPOTTY BUNTING,5,192,830.4000000000001

Australia,PAPER BUNTING RETROSPOT,4,220,585.0

Australia,BAKING SET SPACEBOY DESIGN,4,186,803.1

Australia,RECYCLING BAG RETROSPOT,4,130,248.0

Australia,BAKING SET 9 PIECE RETROSPOT,4,112,487.2

Australia,RED TOADSTOOL LED NIGHT LIGHT,3,1344,1987.1999999999998

Australia,LUNCH BAG SPACEBOY DESIGN,3,450,662.5

Australia,SET OF 12 FAIRY CAKE BAKING CASES,3,426,311.34000000000003

Australia,SET OF 12 MINI LOAF BAKING CASES,3,426,311.34000000000003

Australia,SET OF 6 TEA TIME BAKING CASES,3,418,441.86

Austria,POSTAGE,9,37,1456.0

Austria,RETROSPOT TEA SET CERAMIC 11 PC,4,45,205.95000000000002

Austria,PLASTERS IN TIN CIRCUS PARADE,3,48,79.19999999999999

Austria,DOLLY GIRL LUNCH BOX,3,36,70.19999999999999

Austria,PLASTERS IN TIN VINTAGE PAISLEY,3,36,59.39999999999999

Austria,KIDS RAIN MAC BLUE,2,48,40.8

Austria,KIDS RAIN MAC PINK,2,48,40.8

Austria,PACK OF 60 PINK PAISLEY CAKE CASES,2,48,26.400000000000002

Austria,ROUND SNACK BOXES SET OF4 WOODLAND,2,36,106.20000000000002

Austria,PLASTERS IN TIN WOODLAND ANIMALS,2,36,59.39999999999999

Bahrain,NOVELTY BISCUITS CAKE STAND 3 TIER,2,6,59.699999999999996

Bahrain,ICE CREAM SUNDAE LIP GLOSS,1,96,120.0

…

United Arab Emirates,STRAWBERRY CERAMIC TRINKET BOX,1,24,30.0

United Kingdom,WHITE HANGING HEART T-LIGHT HOLDER,821,34648,94858.6

United Kingdom,REGENCY CAKESTAND 3 TIER,767,9641,110990.2

United Kingdom,PARTY BUNTING,659,13992,63109.380000000005

United Kingdom,ASSORTED COLOUR BIRD ORNAMENT,642,32727,52395.99

United Kingdom,PAPER CHAIN KIT 50'S CHRISTMAS,594,14792,40451.08

United Kingdom,NATURAL SLATE HEART CHALKBOARD,573,8114,23223.420000000002

United Kingdom,SET OF 3 CAKE TINS PANTRY DESIGN,569,5253,25256.350000000002

United Kingdom,JUMBO BAG RED RETROSPOT,562,41981,77371.57

United Kingdom,HEART OF WICKER SMALL,555,16361,27561.539999999997

United Kingdom,PACK OF 72 RETROSPOT CAKE CASES,531,22465,11002.25

Unspecified,ASSORTED COLOUR BIRD ORNAMENT,2,22,37.18

Unspecified,JAM MAKING SET WITH JARS,2,18,70.5

Unspecified,FRIDGE MAGNETS LA VIE EN ROSE,2,18,15.299999999999999

Unspecified,PACK OF 6 BIRDY GIFT TAGS,2,13,16.25

Unspecified,3 STRIPEY MICE FELTCRAFT,2,7,13.649999999999999

Unspecified,SET OF 10 LED DOLLY LIGHTS,2,6,37.5

Unspecified,SET 7 BABUSHKA NESTING BOXES,2,5,42.5

Unspecified,SET OF 2 WOODEN MARKET CRATES,2,4,51.0

Unspecified,RED KITCHEN SCALES,2,3,25.5

Unspecified,ROUND CAKE TIN VINTAGE RED,2,3,23.85

**Retail\_Outputs\_Popularity\_By\_Month (CSV)**

Year,Month,Description,buyers,quantity,revenue

2010,12,WHITE HANGING HEART T-LIGHT HOLDER,155,3611,9602.05

2010,12,REGENCY CAKESTAND 3 TIER,119,1570,17756.7

2010,12,PAPER CHAIN KIT 50'S CHRISTMAS,117,1697,4614.150000000001

2010,12,SCOTTIE DOG HOT WATER BOTTLE,115,827,3908.8500000000004

2010,12,HAND WARMER BABUSHKA DESIGN,114,3367,4172.7

2010,12,CHOCOLATE HOT WATER BOTTLE,99,963,4430.85

2010,12,JAM MAKING SET PRINTED,95,1459,2077.1499999999996

…

2011,11,RABBIT NIGHT LIGHT,324,12393,23268.73

2011,11,PAPER CHAIN KIT 50'S CHRISTMAS,276,5919,16139.01

2011,11,HOT WATER BOTTLE KEEP CALM,216,2102,9500.1

2011,11,PAPER CHAIN KIT VINTAGE CHRISTMAS,195,3181,8654.11

2011,11,JUMBO BAG 50'S CHRISTMAS,192,2972,5984.56

2011,11,WHITE HANGING HEART T-LIGHT HOLDER,191,4861,13855.83

2011,11,ASSORTED COLOUR BIRD ORNAMENT,177,5190,8252.7

2011,11,LUNCH BAG PAISLEY PARK,177,2275,3611.75

2011,11,JUMBO BAG PAISLEY PARK,175,2543,5033.14

2011,11,WOODEN STAR CHRISTMAS SCANDINAVIAN,174,4981,1473.35

2011,12,RABBIT NIGHT LIGHT,103,3691,6889.76

2011,12,PAPER CHAIN KIT 50'S CHRISTMAS,92,1393,3856.4700000000003

2011,12,HOT WATER BOTTLE KEEP CALM,71,468,2182.2000000000003

2011,12,PAPER CHAIN KIT VINTAGE CHRISTMAS,70,739,2028.5700000000002

2011,12,HAND WARMER OWL DESIGN,64,773,1581.39

2011,12,BICYCLE PUNCTURE REPAIR KIT,61,1145,2203.44

2011,12,JUMBO BAG 50'S CHRISTMAS,56,998,1893.84

2011,12,CHOCOLATE HOT WATER BOTTLE,52,468,2131.8

2011,12,HAND WARMER RED LOVE HEART,51,466,978.6000000000001

2011,12,HAND WARMER BIRD DESIGN,49,727,1460.7900000000002

**Retail\_Outputs\_Popularity\_By\_Global (CSV)**

Description,buyers,quantity,revenue

REGENCY CAKESTAND 3 TIER,881,12402,142592.94999999998

WHITE HANGING HEART T-LIGHT HOLDER,856,36725,100448.15000000001

PARTY BUNTING,708,15291,68844.33

ASSORTED COLOUR BIRD ORNAMENT,678,35362,56580.34

SET OF 3 CAKE TINS PANTRY DESIGN,640,7020,33347.8

JUMBO BAG RED RETROSPOT,635,46181,85220.78

PACK OF 72 RETROSPOT CAKE CASES,635,33693,16394.53

PAPER CHAIN KIT 50'S CHRISTMAS,613,15617,42660.83

NATURAL SLATE HEART CHALKBOARD,587,8480,24245.52

BAKING SET 9 PIECE RETROSPOT,581,4797,23007.55

HEART OF WICKER SMALL,573,16775,28228.64

JAM MAKING SET WITH JARS,573,8151,32662.97

SPOTTY BUNTING,571,7639,35539.25

JAM MAKING SET PRINTED,537,15055,21170.75

LUNCH BAG RED RETROSPOT,532,17697,28048.45

LUNCH BAG SPACEBOY DESIGN,490,9912,15881.199999999999

RETROSPOT TEA SET CERAMIC 11 PC,488,4248,20470.780000000002

HEART OF WICKER LARGE,485,9591,28520.95

REX CASH+CARRY JUMBO SHOPPER,480,1893,1798.35

LUNCH BAG CARS BLUE,479,10820,17265.68

**Retail\_Outputs\_Sample\_User\_Recommendations(CSV)**

CustomerID,Recommendations(StockCode),Recommendations(Description)

14911.0,"['79151B', '79161A', 17174, '84569C', 23321, 21931, 20964, 21667, '84743C', 84227, 20827, 21930, '84387A', '71495A', 23343, 22659, 20663, 37461, 90119, 21989]","['SILICON CUBE 25W, BLUE', 'ST GEORGE SET OF 10 PARTY LIGHTS', 'ASSTD RASTA KEY-CHAINS', 'PACK 4 FLOWER/BUTTERFLY PATCHES', 'SMALL WHITE HEART OF WICKER', 'JUMBO STORAGE BAG SUKI', 'POLYESTER FILLER PAD 60x40cm', 'GLASS CAKE COVER AND PLATE', 'ORANGE FELT VASE + FLOWERS', 'HEN HOUSE W CHICK IN NEST', 'GOLD APERITIF GLASS', 'JUMBO STORAGE BAG SKULLS', 'BIRD ON BRANCH CANVAS SCREEN', 'CD WALL TIDY BLUE OFFICE', 'JUMBO BAG VINTAGE CHRISTMAS', 'LUNCH BOX I LOVE LONDON', 'QUEEN OF THE SKIES HOLIDAY PURSE', 'FUNKY MONKEY MUG', 'METALIC LEAVES BAG CHARMS', 'PACK OF 20 SKULL PAPER NAPKINS']"

12748.0,"[22966, '90141E', '90059A', 90168, 90169, 90118, '90059D', '90141D', 90103, 90076, '90177E', '90177A', '90182C', '90202A', 20678, 22326, 23204, 23322, 21667, '84402B']","['GINGERBREAD MAN COOKIE CUTTER', 'ORANGE PENDANT TRIPLE SHELL NECKLAC', 'DIAMANTE HAIR GRIP PACK/2 CRYSTAL', '2 DAISIES HAIR COMB', 'DAISY HAIR COMB', 'PINK DAISY BAG CHARM', 'DIAMANTE HAIR GRIP PACK/2 PERIDOT', 'ROSE PENDANT TRIPLE SHELL NECKLACE', 'PURPLE FRANGIPANI NECKLACE', 'MONTANA DIAMOND CLUSTER EARRINGS', 'DROP DIAMANTE EARRINGS GREEN', 'CLASSIC DIAMANTE EARRINGS JET', 'BLACK 3 BEAD DROP EARRINGS', 'PURPLE ENAMEL FLOWER HAIR TIE', 'LARGE BLACK DIAMANTE HAIRSLIDE', 'ROUND SNACK BOXES SET OF4 WOODLAND', 'CHARLOTTE BAG APPLES DESIGN', 'LARGE WHITE HEART OF WICKER', 'GLASS CAKE COVER AND PLATE', 'PURPLE DRESS JEWELLERY STAND']"

17841.0,"['90059D', '90059A', '90141D', '90141E', '90177E', '90177A', 90076, 90168, 90103, 90118, '90182C', 90169, 20678, 84879, '90202A', 20827, '84387A', '71495A', 22086, 21166]","['DIAMANTE HAIR GRIP PACK/2 PERIDOT', 'DIAMANTE HAIR GRIP PACK/2 CRYSTAL', 'ROSE PENDANT TRIPLE SHELL NECKLACE', 'ORANGE PENDANT TRIPLE SHELL NECKLAC', 'DROP DIAMANTE EARRINGS GREEN', 'CLASSIC DIAMANTE EARRINGS JET', 'MONTANA DIAMOND CLUSTER EARRINGS', '2 DAISIES HAIR COMB', 'PURPLE FRANGIPANI NECKLACE', 'PINK DAISY BAG CHARM', 'BLACK 3 BEAD DROP EARRINGS', 'DAISY HAIR COMB', 'LARGE BLACK DIAMANTE HAIRSLIDE', 'ASSORTED COLOUR BIRD ORNAMENT', 'PURPLE ENAMEL FLOWER HAIR TIE', 'GOLD APERITIF GLASS', 'BIRD ON BRANCH CANVAS SCREEN', 'CD WALL TIDY BLUE OFFICE', ""PAPER CHAIN KIT 50'S CHRISTMAS"", 'COOK WITH WINE METAL SIGN']"

14096.0,"[23209, 20727, 23298, 23203, 23206, 23208, 22722, 22386, 21755, 84879, 21930, 23321, 20964, 21175, 21667, '84743C', 22326, 23204, 23243, '90141D']","['LUNCH BAG DOILEY PATTERN', 'LUNCH BAG  BLACK SKULL.', 'SPOTTY BUNTING', 'JUMBO BAG DOILEY PATTERNS', 'LUNCH BAG APPLE DESIGN', 'LUNCH BAG VINTAGE LEAF DESIGN', 'SET OF 6 SPICE TINS PANTRY DESIGN', 'JUMBO BAG PINK POLKADOT', 'LOVE BUILDING BLOCK WORD', 'ASSORTED COLOUR BIRD ORNAMENT', 'JUMBO STORAGE BAG SKULLS', 'SMALL WHITE HEART OF WICKER', 'POLYESTER FILLER PAD 60x40cm', 'GIN + TONIC DIET METAL SIGN', 'GLASS CAKE COVER AND PLATE', 'ORANGE FELT VASE + FLOWERS', 'ROUND SNACK BOXES SET OF4 WOODLAND', 'CHARLOTTE BAG APPLES DESIGN', 'SET OF TEA COFFEE SUGAR TINS PANTRY', 'ROSE PENDANT TRIPLE SHELL NECKLACE']"

…

14646.0,"[23208, 23202, 22722, 23308, 22469, 21243, 22470, 22045, 82482, 21733, 22457, 21915, 22197, 21755, 22139, 22965, 22748, 22379, 23300, 22271]","['LUNCH BAG VINTAGE LEAF DESIGN', 'JUMBO BAG VINTAGE LEAF', 'SET OF 6 SPICE TINS PANTRY DESIGN', 'SET OF 60 VINTAGE LEAF CAKE CASES', 'HEART OF WICKER SMALL', 'PINK  POLKADOT PLATE', 'HEART OF WICKER LARGE', 'SPACEBOY GIFT WRAP', 'WOODEN PICTURE FRAME WHITE FINISH', 'RED HANGING HEART T-LIGHT HOLDER', 'NATURAL SLATE HEART CHALKBOARD', 'RED  HARMONICA IN BOX', 'SMALL POPCORN HOLDER', 'LOVE BUILDING BLOCK WORD', 'RETROSPOT TEA SET CERAMIC 11 PC', '3 TRADITIONAL COOKIE CUTTERS  SET', ""POPPY'S PLAYHOUSE KITCHEN"", 'RECYCLING BAG RETROSPOT', 'GARDENERS KNEELING PAD CUP OF TEA', 'FELTCRAFT DOLL ROSIE']"

13089.0,"[22383, 23208, 22993, 23202, 23322, 21931, 23321, 23300, 22551, 23301, 20712, 23243, 21929, 85152, 21930, 22804, 23200, 22328, 22352, 22355]","['LUNCH BAG SUKI  DESIGN', 'LUNCH BAG VINTAGE LEAF DESIGN', 'SET OF 4 PANTRY JELLY MOULDS', 'JUMBO BAG VINTAGE LEAF', 'LARGE WHITE HEART OF WICKER', 'JUMBO STORAGE BAG SUKI', 'SMALL WHITE HEART OF WICKER', 'GARDENERS KNEELING PAD CUP OF TEA', 'PLASTERS IN TIN SPACEBOY', 'GARDENERS KNEELING PAD KEEP CALM', 'JUMBO BAG WOODLAND ANIMALS', 'SET OF TEA COFFEE SUGAR TINS PANTRY', 'JUMBO BAG PINK VINTAGE PAISLEY', 'HAND OVER THE CHOCOLATE   SIGN', 'JUMBO STORAGE BAG SKULLS', 'CANDLEHOLDER PINK HANGING HEART', 'JUMBO BAG PEARS', 'ROUND SNACK BOXES SET OF 4 FRUITS', 'LUNCH BOX WITH CUTLERY RETROSPOT', 'CHARLOTTE BAG SUKI DESIGN']"

A graph of a number of data

AI-generated content may be incorrect.

A graph of a number of people

AI-generated content may be incorrect.

A graph of a bar graph

AI-generated content may be incorrect.

A line graph with a line pointing upwards

AI-generated content may be incorrect.

A graph of a number of people

AI-generated content may be incorrect.

# Conclusion

This project demonstrates the design and implementation of a hybrid recommendation system for an online retail store. By combining popularity ranking, collaborative filtering, and co-occurrence analysis, the system provides a balance of accuracy and diversity in recommendations. While this report used a subset of the dataset for demonstration purposes, the pipeline is fully scalable to the complete dataset and can be deployed in a real-world e-commerce environment.

Future improvements could include:

- Adding customer segmentation to personalize strategies further.

- Using advanced matrix factorization or deep learning approaches.

- Incorporating real-time data streaming for live updates.

# References

1. Online Retail Dataset (UCI Machine Learning Repository).

2. Aggarwal, C. C. (2016). Recommender Systems: The Textbook. Springer.

3. Ricci, F., Rokach, L., Shapira, B. (2015). Recommender Systems Handbook. Springer.

4. Official Python documentation: https://docs.python.org/3/

5. Pandas documentation: https://pandas.pydata.org/

# Appendix: Key Code Snippets

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Online Retail Recommendation System — Final Auto-run Script

Features:

- Popularity-first analysis (Global / Country / Month)

- Item-Item Collaborative Filtering (CF)

- Frequently-Bought-Together (FBT)

- Presentation-ready Seaborn charts (readable labels, wrapped/truncated)

- Combined dashboard saved with date in filename

- Interactive plotting (works in VS Code and Colab)

- Outputs: CSVs + PNGs

Author: Kusan Chakraborty

"""

import os

import json

import textwrap

from datetime import datetime

from collections import defaultdict, Counter

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# ------------------- CONFIG -------------------

DATA\_FILE = "OnlineRetail(1).xlsx"  # place this file in same folder

OUTPUT\_PREFIX = "retail\_outputs"

TOPN = 20  # change top-N globally here

TOP\_ITEMS\_CF = 5000  # cap for CF to control memory

TRUNCATE\_LEN = 28  # max characters before truncation/wrapping

DASHBOARD\_TOP\_COUNTRIES = 3  # countries to show on dashboard country subplot

# ----------------------------------------------

# Enable interactive mode so plots show live in VS Code and remain until closed

plt.ion()

def ensure\_dir(path):

    d = os.path.dirname(path)

    if d and not os.path.exists(d):

        os.makedirs(d, exist\_ok=True)

# ------------------- Unified label helpers -------------------

def shorten\_and\_wrap(text, max\_len=TRUNCATE\_LEN, max\_lines=3):

    """Return text wrapped into lines of width max\_len.

       Limits to max\_lines and adds '...' if truncated."""

    if pd.isna(text):

        return ""

    s = str(text)

    if len(s) <= max\_len:

        return s

    # wrap into lines

    lines = textwrap.wrap(s, width=max\_len)

    if len(lines) > max\_lines:

        lines = lines[:max\_lines]

        if len(lines[-1]) > max\_len - 3:

            lines[-1] = lines[-1][:max\_len - 3] + "..."

        else:

            lines[-1] = lines[-1] + "..."

    return "\n".join(lines)

def prepare\_label\_col(df, col="Description", max\_len=TRUNCATE\_LEN):

    """Add \_LabelRaw and \_Label columns to df for plotting labels."""

    df = df.copy()

    if col not in df.columns:

        df["\_LabelRaw"] = ""

        df["\_Label"] = ""

        return df

    df["\_LabelRaw"] = df[col].fillna("").astype(str)

    df["\_Label"] = df["\_LabelRaw"].apply(lambda x: shorten\_and\_wrap(x, max\_len=max\_len))

    return df

# ------------------- Data Load & Clean -------------------

def load\_and\_clean(path):

    if not os.path.exists(path):

        raise FileNotFoundError(f"Dataset file not found: {path}. Put it in the same folder as this script.")

    print(f"Loading dataset from {path} ...")

    df = pd.read\_excel(path, engine="openpyxl",

                       usecols=["InvoiceNo", "StockCode", "Description", "Quantity", "InvoiceDate", "UnitPrice", "CustomerID", "Country"])

    df = df.dropna(subset=["CustomerID"]).copy()

    df["InvoiceNo"] = df["InvoiceNo"].astype(str)

    df = df[~df["InvoiceNo"].str.startswith("C")]  # remove cancellations

    df = df[(df["Quantity"] > 0) & (df["UnitPrice"] > 0)]

    df["InvoiceDate"] = pd.to\_datetime(df["InvoiceDate"], errors="coerce")

    df = df.dropna(subset=["InvoiceDate"])

    df["TotalPrice"] = df["Quantity"] \* df["UnitPrice"]

    df["Year"] = df["InvoiceDate"].dt.year

    df["Month"] = df["InvoiceDate"].dt.month

    return df

# ------------------- Descriptions & Popularity -------------------

def describe\_data(df):

    return {

        "rows": len(df),

        "unique\_customers": int(df["CustomerID"].nunique()),

        "unique\_items": int(df["StockCode"].nunique()),

        "countries": int(df["Country"].nunique()),

        "date\_min": str(df["InvoiceDate"].min()),

        "date\_max": str(df["InvoiceDate"].max()),

    }

def \_popularity\_core(g):

    pop = (g.groupby("Description")

           .agg(buyers=("CustomerID", "nunique"),

                quantity=("Quantity", "sum"),

                revenue=("TotalPrice", "sum"))

           .sort\_values(["buyers", "quantity", "revenue"], ascending=False))

    return pop

def popularity\_global(df, topn=TOPN):

    return \_popularity\_core(df).head(topn).reset\_index()

def popularity\_by\_country(df, topn=10):

    rows = []

    for country, g in df.groupby("Country"):

        pop = \_popularity\_core(g).head(topn).reset\_index()

        pop.insert(0, "Country", country)

        rows.append(pop)

    return pd.concat(rows, ignore\_index=True) if rows else pd.DataFrame()

def popularity\_by\_month(df, topn=10):

    rows = []

    for (y, m), g in df.groupby(["Year", "Month"]):

        pop = \_popularity\_core(g).head(topn).reset\_index()

        pop.insert(0, "Year", y); pop.insert(1, "Month", m)

        rows.append(pop)

    return pd.concat(rows, ignore\_index=True) if rows else pd.DataFrame()

# ------------------- Item-Item CF -------------------

class ItemCF:

    def \_\_init\_\_(self, user\_item\_df, popularity\_fallback):

        self.user\_item = user\_item\_df

        X = self.user\_item.to\_numpy(dtype=np.float32)

        col\_norms = np.sqrt((X\*\*2).sum(axis=0, keepdims=True)) + 1e-9

        self.Xn = X / col\_norms

        self.items = user\_item\_df.columns.tolist()

        self.idx\_to\_item = {i: self.items[i] for i in range(len(self.items))}

        self.item\_to\_idx = {v: k for k, v in self.idx\_to\_item.items()}

        self.\_sims = None

        self.popularity\_fallback = popularity\_fallback

    def \_cosine\_sims(self, chunk=800):

        if self.\_sims is not None:

            return self.\_sims

        Xn = self.Xn; n = Xn.shape[1]

        sims = np.zeros((n, n), dtype=np.float32)

        for start in range(0, n, chunk):

            end = min(start + chunk, n)

            sims[start:end, :] = Xn[:, start:end].T @ Xn

        np.fill\_diagonal(sims, 0.0)

        self.\_sims = sims

        return sims

    def recommend(self, user\_id, topn=10, k\_neighbors=50):

        if user\_id not in self.user\_item.index:

            return self.popularity\_fallback[:topn]

        u = self.user\_item.loc[user\_id].to\_numpy(dtype=np.float32)

        bought\_idx = np.where(u > 0)[0]

        if len(bought\_idx) == 0:

            return self.popularity\_fallback[:topn]

        sims = self.\_cosine\_sims()

        scores = np.zeros(sims.shape[0], dtype=np.float32)

        for bi in bought\_idx:

            row = sims[bi]

            if k\_neighbors and k\_neighbors < len(row):

                topk = np.argpartition(-row, k\_neighbors)[:k\_neighbors]

                scores[topk] += row[topk]

            else:

                scores += row

        scores[bought\_idx] = -np.inf

        top\_idx = np.argpartition(-scores, topn)[:topn]

        top\_idx = top\_idx[np.argsort(-scores[top\_idx])]

        return [self.idx\_to\_item[i] for i in top\_idx]

def build\_user\_item(df\_train, top\_items=TOP\_ITEMS\_CF):

    top\_ids = (df\_train.groupby("StockCode")["CustomerID"].nunique()

               .sort\_values(ascending=False).head(top\_items).index)

    t = df\_train[df\_train["StockCode"].isin(top\_ids)].copy()

    ui = (t.assign(interaction=1)

          .drop\_duplicates(subset=["CustomerID", "StockCode"])

          .pivot(index="CustomerID", columns="StockCode", values="interaction")

          .fillna(0).astype(np.float32))

    return ui

# ------------------- FBT -------------------

def build\_fbt(df\_train, whitelist=None):

    if whitelist is not None:

        df\_train = df\_train[df\_train["StockCode"].isin(whitelist)].copy()

    co = defaultdict(Counter)

    for \_, g in df\_train.groupby("InvoiceNo"):

        items = list(set(g["StockCode"].tolist()))

        for i in range(len(items)):

            for j in range(i + 1, len(items)):

                a, b = items[i], items[j]

                co[a][b] += 1

                co[b][a] += 1

    return co

def fbt\_for\_item(item\_id, co\_counts, popularity\_index, topn=10):

    if item\_id in co\_counts and len(co\_counts[item\_id]) > 0:

        return [pid for pid, \_ in co\_counts[item\_id].most\_common(topn)]

    return [i for i in popularity\_index if i != item\_id][:topn]

# ------------------- Evaluation -------------------

def build\_user\_last\_basket(df\_part):

    if df\_part.empty:

        return pd.DataFrame(columns=["CustomerID", "InvoiceNo", "StockCode"])

    last = df\_part.sort\_values("InvoiceDate").groupby("CustomerID").tail(1)

    inv\_ids = last["InvoiceNo"].unique().tolist()

    inv\_items = (df\_part[df\_part["InvoiceNo"].isin(inv\_ids)][["InvoiceNo", "CustomerID", "StockCode"]]

                 .drop\_duplicates().groupby(["CustomerID", "InvoiceNo"])["StockCode"].apply(list).reset\_index())

    return inv\_items

def hit\_rate\_at\_k(eval\_data, recommender, items\_index, k=10, max\_users=2000):

    if eval\_data.empty:

        return float("nan")

    hits = 0; total = 0

    for \_, row in eval\_data.head(max\_users).iterrows():

        truth = set([i for i in row["StockCode"] if i in items\_index])

        if not truth: continue

        preds = recommender(row["CustomerID"], k)

        hits += int(len(truth.intersection(set(preds))) > 0)

        total += 1

    return hits / total if total > 0 else float("nan")

# ------------------- Plotting helpers -------------------

def save\_and\_show(fig, filename):

    ensure\_dir(filename)

    fig.savefig(filename, bbox\_inches="tight")

    plt.show()

    plt.pause(0.1)

def plot\_global(pop\_global, out\_png):

    pop = prepare\_label\_col(pop\_global)

    sns.set\_palette("Set2")

    fig, ax = plt.subplots(figsize=(18, 12))

    sns.barplot(data=pop.sort\_values("buyers", ascending=True), x="buyers", y="\_Label", ax=ax)

    ax.set\_title(f"Top {len(pop\_global)} Products Globally (by unique buyers)")

    ax.set\_xlabel("Unique Buyers"); ax.set\_ylabel("Product")

    save\_and\_show(fig, out\_png)

def plot\_country(pop\_country, out\_png, top\_per\_country=5):

    if pop\_country.empty:

        return

    pop = pop\_country.copy()

    top\_countries = pop["Country"].value\_counts().head(6).index.tolist()

    pop = pop[pop["Country"].isin(top\_countries)]

    pop = pop.groupby("Country").head(top\_per\_country)

    pop = prepare\_label\_col(pop, col="Description", max\_len=15)

    sns.set\_palette("Set2")

    fig\_height = max(8, 0.5 \* len(pop))

    fig, ax = plt.subplots(figsize=(14, fig\_height))

    sns.barplot(

        data=pop.sort\_values("buyers", ascending=True),

        x="buyers", y="\_Label", hue="Country",

        dodge=False, ax=ax

    )

    ax.set\_title(f"Top {top\_per\_country} Products by Country (Top {len(top\_countries)} Countries)")

    ax.set\_xlabel("Unique Buyers")

    ax.set\_ylabel("Product")

    plt.tight\_layout()

    save\_and\_show(fig, out\_png)

def plot\_monthly(df, out\_png):

    sns.set\_palette("Set2")

    month\_summary = df.groupby(["Year", "Month"])["CustomerID"].nunique().reset\_index(name="unique\_buyers")

    fig, ax = plt.subplots(figsize=(16, 8))

    sns.lineplot(data=month\_summary, x="Month", y="unique\_buyers", hue="Year", marker="o", ax=ax)

    ax.set\_title("Unique Buyers per Month (by Year)")

    ax.set\_xlabel("Month"); ax.set\_ylabel("Unique Buyers")

    save\_and\_show(fig, out\_png)

def plot\_sample\_recs(df\_samples, out\_png):

    if df\_samples.empty:

        return

    row = df\_samples.iloc[0]

    descs = row["Recommendations(Description)"]

    if isinstance(descs, str):

        try:

            descs = json.loads(descs.replace("'", '"'))

        except Exception:

            descs = [d.strip() for d in descs.strip("[]").split(",") if d.strip()]

    descs = [shorten\_and\_wrap(d, TRUNCATE\_LEN) for d in descs]

    y = list(reversed(descs))

    x = list(range(1, len(y) + 1))

    sns.set\_palette("Set2")

    fig, ax = plt.subplots(figsize=(16, 8))

    sns.barplot(x=x, y=y, orient="h", ax=ax)

    ax.set\_title(f"Sample Recommendations for Customer {row['CustomerID']} (Top {len(x)})")

    ax.set\_xlabel("Rank"); ax.set\_ylabel("Recommended Items")

    save\_and\_show(fig, out\_png)

def plot\_fbt(df\_fbt, out\_png):

    if df\_fbt.empty:

        return

    row = df\_fbt.iloc[0]

    items = row["FBT(Description)"]

    if isinstance(items, str):

        try:

            items = json.loads(items.replace("'", '"'))

        except Exception:

            items = [d.strip() for d in items.strip("[]").split(",") if d.strip()]

    items = [shorten\_and\_wrap(d, TRUNCATE\_LEN) for d in items]

    y = list(reversed(items))

    x = list(range(1, len(y) + 1))

    sns.set\_palette("Set2")

    fig, ax = plt.subplots(figsize=(16, 8))

    sns.barplot(x=x, y=y, orient="h", ax=ax)

    ax.set\_title("Frequently Bought Together (example for a top item)")

    ax.set\_xlabel("Rank"); ax.set\_ylabel("Items")

    save\_and\_show(fig, out\_png)

def print\_key\_insights(pop\_global, pop\_country, df, df\_fbt):

    print("\n===== KEY INSIGHTS =====")

    print("\nTop 5 Most Popular Products (Global):")

    top5 = pop\_global.head(5)

    for idx, row in top5.iterrows():

        print(f"{idx + 1}. {row['Description']} - {row['buyers']} buyers")

    print("\nTop 3 Countries by Unique Buyers:")

    buyers\_by\_country = pop\_country.groupby("Country")["buyers"].sum().nlargest(3)

    for i, (country, buyers) in enumerate(buyers\_by\_country.items(), start=1):

        print(f"{i}. {country} - {buyers} buyers")

    monthly\_buyers = df.groupby(["Year", "Month"])["CustomerID"].nunique().reset\_index()

    busiest = monthly\_buyers.loc[monthly\_buyers["CustomerID"].idxmax()]

    print(f"\nBusiest Month: {int(busiest['Month'])}/{int(busiest['Year'])} "

          f"with {busiest['CustomerID']} unique buyers")

    if not df\_fbt.empty:

        print("\nTop 3 Frequently Bought Together Pairs (Example Items):")

        for i, row in df\_fbt.head(3).iterrows():

            descs = row["FBT(Description)"]

            if isinstance(descs, str):

                try:

                    descs = json.loads(descs.replace("'", '"'))

                except:

                    descs = [d.strip() for d in descs.strip("[]").split(",") if d.strip()]

            print(f"{i + 1}. {', '.join(descs[:2])}")

# ------------------- Main -------------------

def main():

    df = load\_and\_clean(DATA\_FILE)

    ds = describe\_data(df)

    print("\nData summary:\n")

    for key, value in ds.items():

        print(f"{key}: {value}")

    pop\_global = popularity\_global(df, TOPN)

    pop\_country = popularity\_by\_country(df, topn=max(10, TOPN // 2))

    pop\_month = popularity\_by\_month(df, topn=max(10, TOPN // 2))

    pop\_global.to\_csv(f"{OUTPUT\_PREFIX}\_popularity\_global.csv", index=False)

    pop\_country.to\_csv(f"{OUTPUT\_PREFIX}\_popularity\_by\_country.csv", index=False)

    pop\_month.to\_csv(f"{OUTPUT\_PREFIX}\_popularity\_by\_month.csv", index=False)

    print("\nSaved popularity CSVs.\n")

    plot\_global(pop\_global, f"{OUTPUT\_PREFIX}\_chart\_global.png")

    plot\_country(pop\_country, f"{OUTPUT\_PREFIX}\_chart\_country.png")

    plot\_monthly(df, f"{OUTPUT\_PREFIX}\_chart\_month.png")

    train = df[df["InvoiceDate"] <= (df["InvoiceDate"].max() - pd.Timedelta(days=30))].copy()

    if len(train) < 0.6 \* len(df):

        s = df.sort\_values("InvoiceDate"); split = int(0.8 \* len(s))

        train = s.iloc[:split].copy()

    valid = df.drop(train.index)

    print("Train rows:", len(train), "\nValid rows:", len(valid))

    global\_pop\_items = train.groupby("StockCode")["CustomerID"].nunique().sort\_values(ascending=False).index.tolist()

    print("\nBuilding user-item matrix for CF (may take time on full dataset)...")

    ui = build\_user\_item(train, top\_items=TOP\_ITEMS\_CF)

    items\_index = ui.columns.tolist()

    model = ItemCF(ui, popularity\_fallback=global\_pop\_items)

    # Build FBT

    co = build\_fbt(train, whitelist=items\_index)

    # Evaluation

    eval\_data = build\_user\_last\_basket(valid)

    def \_rec(uid, k): return model.recommend(uid, topn=k)

    hr10 = hit\_rate\_at\_k(eval\_data, \_rec, items\_index, k=10, max\_users=2000)

    print(f"\nHit@10 (validation last-invoice): {hr10 \* 100:.2f}%\n" if hr10 == hr10 else "Hit@10 (validation last-invoice): N/A")

    # Sample recommendations CSV

    name\_map = train.drop\_duplicates("StockCode")[["StockCode", "Description"]].set\_index("StockCode")["Description"].to\_dict()

    top\_users = train.groupby("CustomerID")["StockCode"].nunique().sort\_values(ascending=False).head(10).index.tolist()

    samples = []

    for u in top\_users:

        recs = model.recommend(u, topn=TOPN)

        samples.append({

            "CustomerID": u,

            "Recommendations(StockCode)": recs,

            "Recommendations(Description)": [name\_map.get(x, str(x)) for x in recs]

        })

    df\_samples = pd.DataFrame(samples)

    df\_samples.to\_csv(f"{OUTPUT\_PREFIX}\_sample\_user\_recommendations.csv", index=False)

    print("Saved sample user recommendations CSV.")

    # FBT CSV

    fbt\_rows = []

    for it in global\_pop\_items[:5]:

        recs = fbt\_for\_item(it, co, global\_pop\_items, topn=TOPN)

        fbt\_rows.append({

            "Item(StockCode)": it,

            "FBT(StockCode)": recs,

            "FBT(Description)": [name\_map.get(x, str(x)) for x in recs]

        })

    df\_fbt = pd.DataFrame(fbt\_rows)

    df\_fbt.to\_csv(f"{OUTPUT\_PREFIX}\_fbt\_recommendations.csv", index=False)

    print("Saved FBT CSV.")

    # Plot sample recs and FBT

    plot\_sample\_recs(df\_samples, f"{OUTPUT\_PREFIX}\_chart\_sample\_recs.png")

    plot\_fbt(df\_fbt, f"{OUTPUT\_PREFIX}\_chart\_fbt.png")

    # After saving FBT CSV

    print\_key\_insights(pop\_global, pop\_country, df, df\_fbt)

    print("\nAll outputs written. Files:")

    out\_files = [

        f"{OUTPUT\_PREFIX}\_popularity\_global.csv",

        f"{OUTPUT\_PREFIX}\_popularity\_by\_country.csv",

        f"{OUTPUT\_PREFIX}\_popularity\_by\_month.csv",

        f"{OUTPUT\_PREFIX}\_sample\_user\_recommendations.csv",

        f"{OUTPUT\_PREFIX}\_fbt\_recommendations.csv",

        f"{OUTPUT\_PREFIX}\_chart\_global.png",

        f"{OUTPUT\_PREFIX}\_chart\_country.png",

        f"{OUTPUT\_PREFIX}\_chart\_month.png",

        f"{OUTPUT\_PREFIX}\_chart\_sample\_recs.png",

        f"{OUTPUT\_PREFIX}\_chart\_fbt.png",

    ]

    for f in out\_files:

        print(" -", f)

if \_\_name\_\_ == "\_\_main\_\_":

    main()