Mid-Term Project Report

Name – Srushti Thakre

Email ID – st2229@njit.edu

Course – CS634 Data Mining

Instructor – Dr. Yasser Abduallah

Abstract:

This project performs market-basket analysis on five small, deterministic datasets (Amazon, BestBuy, KMart, Nike, Walmart). A **brute-force** frequent-itemset miner enumerates L1/L2/L3… itemsets and generates association rules (A → B). In addition, Python libraries (mlxtend) are used to run **Apriori** and **FP-Growth** on the same transactions and thresholds. Outputs report support, confidence, and simple timings. The code runs via a command-line interface (CLI) and a Jupyter Notebook; screenshots show representative outputs.

1. Environment & Setup

- **Python:** 3.12 (any 3.10+ is fine). Use a local virtual environment (venv- recommended) during development.
- **Install:**
  pip install -r requirements.txt
- **Run CLI (brute force):**
  python src/cli.py
  The program prompts for the dataset and thresholds (support % and confidence %).
- **Run Notebook:** open and execute notebooks/midtermproject.ipynb top-to-bottom.
- **Dependencies (requirements.txt):**
  mlxtend, pandas, numpy (exact versions per pip freeze).

2. Github Repository

- This project is uploaded here: https://github.com/srushti-510/thakre_srushti_midtermproject
- Collaborator access given to – ya54@njit.edu ; hl534@njit.edu

3. Project Layout

```
thakre_srushti_midtermproject/
├── data/                                  # five CSV datasets (one basket per line)
│   ├── amazon.csv
│   ├── bestbuy.csv
│   ├── kmart.csv
│   ├── nike.csv
│   └── walmart.csv
├── notebooks/
│   └── midtermproject.ipynb               # project walkthrough + observations
├── report/                                # project report
│   └── thakre_srushti_midtermproject.pdf
├── src/
│   ├── cli.py                             # interactive runner (menu + minsup/minconf)
│   ├── bruteforce.py                      # frequent itemsets via enumeration (L1, L2, L3…)
│   ├── rules.py                           # association rule generation (A → B) & printing
│   ├── io_utils.py                        # CSV → list-of-sets transaction loader
│   └── apriori_fp.py                      # Apriori & FP-Growth wrappers (mlxtend)
├── README.md                              # quick start, layout, usage
├── requirements.txt                       # Python dependencies
└── .gitignore                             # ignore venv/.idea/outputs, etc.
```

4. Datasets

Four of the five CSV datasets were derived directly from Midterm_Project_Items_Datasets_Examples.pdf provided by the instructor (Amazon, BestBuy, KMart, Nike). Each file is a small, deterministic list of transactions where each line is one basket of comma-separated items (no header).

To keep names consistent, minor cleanup (trimming whitespace, normalizing item casing) was applied during CSV creation.

Walmart CSV was created separately to mirror realistic grocery baskets (e.g., bread, milk, eggs, etc.). Transactions were written deterministically (no randomness) so results are reproducible across runs.

Below is the Amazon dataset for reference:

	C1	C2	C3	C4	C5
1	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Android Programmin…	<unset>
2	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	<unset>	<unset>
3	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Android Programmin…	Head First Java 2nd Edi
4	Android Programming: The …	Head First Java 2nd Editi…	Beginning Programmi…	<unset>	<unset>
5	Android Programming: The …	Beginning Programming wit…	Java 8 Pocket Guide	<unset>	<unset>
6	A Beginner's Guide	Android Programming: The …	Head First Java 2nd…	<unset>	<unset>
7	A Beginner's Guide	Head First Java 2nd Editi…	Beginning Programmi…	<unset>	<unset>
8	Java: The Complete Refere…	Java For Dummies	Android Programming…	<unset>	<unset>
9	Java For Dummies	Android Programming: The …	Head First Java 2nd…	Beginning Programm…	<unset>
10	Beginning Programming wit…	Java 8 Pocket Guide	C++ Programming in …	<unset>	<unset>
11	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Android Programmin…	<unset>
12	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	HTML and CSS: Desi…	<unset>
13	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Java 8 Pocket Guide	HTML and CSS: Design an
14	Java For Dummies	Android Programming: The …	Head First Java 2nd…	<unset>	<unset>
15	Java For Dummies	Android Programming: The …	<unset>	<unset>	<unset>
16	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Android Programmin…	<unset>
17	A Beginner's Guide	Java: The Complete Refere…	Java For Dummies	Android Programmin…	<unset>
18	Head First Java 2nd Editi…	Beginning Programming wit…	Java 8 Pocket Guide	<unset>	<unset>
19	Android Programming: The …	Head First Java 2nd Editi…	<unset>	<unset>	<unset>

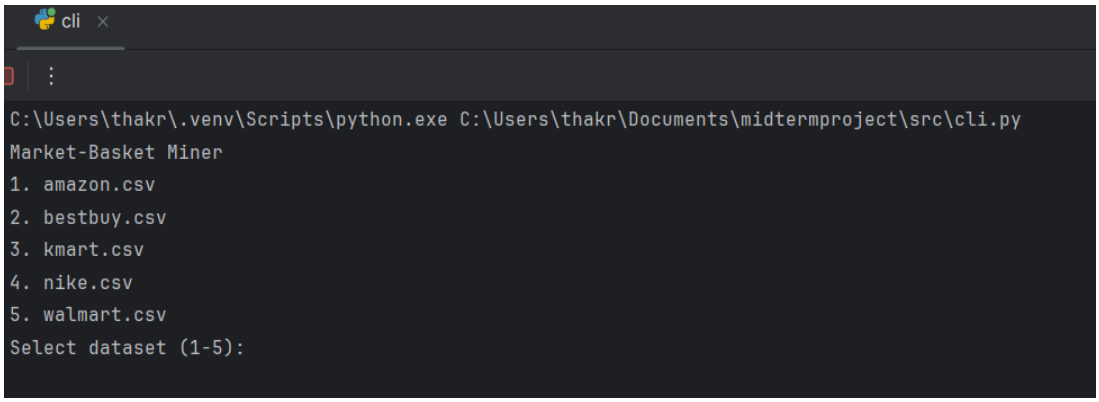
```
1  A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch
2  A Beginner's Guide,Java: The Complete Reference,Java For Dummies
3  A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch,Head First Ja
4  Android Programming: The Big Nerd Ranch,Head First Java 2nd Edition,Beginning Programming with Java
5  Android Programming: The Big Nerd Ranch,Beginning Programming with Java,Java 8 Pocket Guide
6  A Beginner's Guide,Android Programming: The Big Nerd Ranch,Head First Java 2nd Edition
7  A Beginner's Guide,Head First Java 2nd Edition,Beginning Programming with Java
8  Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch
9  Java For Dummies,Android Programming: The Big Nerd Ranch,Head First Java 2nd Edition,Beginning Programming with Java
10 Beginning Programming with Java,Java 8 Pocket Guide,C++ Programming in Easy Steps
11 A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch
12 A Beginner's Guide,Java: The Complete Reference,Java For Dummies,HTML and CSS: Design and Build Websites
13 A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Java 8 Pocket Guide,HTML and CSS: Design and Build We
14 Java For Dummies,Android Programming: The Big Nerd Ranch,Head First Java 2nd Edition
15 Java For Dummies,Android Programming: The Big Nerd Ranch
16 A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch
17 A Beginner's Guide,Java: The Complete Reference,Java For Dummies,Android Programming: The Big Nerd Ranch
18 Head First Java 2nd Edition,Beginning Programming with Java,Java 8 Pocket Guide
19 Android Programming: The Big Nerd Ranch,Head First Java 2nd Edition
20 A Beginner's Guide,Java: The Complete Reference,Java For Dummies
```

5. How to Run

A) Command line (brute force):

"python src/cli.py"

1. Menu shows the 5 datasets.



2. Choose any one of these 5, say for example: 1. amazon.csv.
   Now, enter min support = 20 and min confidence = 50:

```
cli ×

C:\Users\thakr\.venv\Scripts\python.exe C:\Users\thakr\Documents\midtermproject\src\cli.py
Market-Basket Miner
1. amazon.csv
2. bestbuy.csv
3. kmart.csv
4. nike.csv
5. walmart.csv
Select dataset (1-5): 1
Minimum Support (e.g., 0.2 or 20%): 20
Minimum Confidence (e.g., 0.2 or 20%): 50
```

3. Output prints:

   - L1/L2/L3… frequent itemsets with support & counts.

```
Store: amazon | file=C:\Users\thakr\Documents\midtermproject\data\amazon.csv | transactions=20 | minsup=20.00% | minconf=50.00%

L1 (frequent 1-itemsets):
['A Beginner's Guide'] | count=11 | support=55.00%
['Android Programming: The Big Nerd Ranch'] | count=13 | support=65.00%
['Beginning Programming with Java'] | count=6 | support=30.00%
['Head First Java 2nd Edition'] | count=8 | support=40.00%
['Java 8 Pocket Guide'] | count=4 | support=20.00%
['Java For Dummies'] | count=13 | support=65.00%
['Java: The Complete Reference'] | count=10 | support=50.00%

L2 (frequent 2-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] | count=6 | support=30.00%
['A Beginner's Guide', 'Java For Dummies'] | count=9 | support=45.00%
['A Beginner's Guide', 'Java: The Complete Reference'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Head First Java 2nd Edition'] | count=6 | support=30.00%
['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] | count=6 | support=30.00%
['Beginning Programming with Java', 'Head First Java 2nd Edition'] | count=4 | support=20.00%
['Java For Dummies', 'Java: The Complete Reference'] | count=10 | support=50.00%

L3 (frequent 3-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] | count=5 | support=25.00%
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] | count=5 | support=25.00%
['A Beginner's Guide', 'Java For Dummies', 'Java: The Complete Reference'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] | count=6 | support=30.00%

L4 (frequent 4-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] | count=5 | support=25.00%
project > src > cli.py                                                                                    132:27  CRLF  UTF-8
```

- Final association rules (A → B) with support & confidence.

```
Final Association Rules (A -> B):
Rule 1: ['Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 50.00% (count=10)  Confidence: 100.00%
Rule 2: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 30.00% (count=6)  Confidence: 100.00%
Rule 3: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 100.00%
Rule 4: ['A Beginner's Guide', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 100.00%
Rule 5: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 100.00%
Rule 6: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 100.00%
Rule 7: ['Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 8: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 9: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 10: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 11: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 12: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 13: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 14: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 83.33%
```

```
Rule 15: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 16: ['A Beginner's Guide'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 17: ['A Beginner's Guide'] -> ['Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 18: ['A Beginner's Guide'] -> ['Java For Dummies', 'Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 19: ['Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 50.00% (count=10)  Confidence: 76.92%
Rule 20: ['Head First Java 2nd Edition'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 75.00%
Rule 21: ['Java For Dummies'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 22: ['Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 23: ['Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 24: ['Java For Dummies'] -> ['A Beginner's Guide', 'Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 25: ['Beginning Programming with Java'] -> ['Head First Java 2nd Edition']
  Support: 20.00% (count=4)  Confidence: 66.67%
```

```
Rule 25: ['Beginning Programming with Java'] -> ['Head First Java 2nd Edition']
  Support: 20.00% (count=4)  Confidence: 66.67%
Rule 26: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 30.00% (count=6)  Confidence: 66.67%
Rule 27: ['Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 28: ['Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch', 'Java For Dummies']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 29: ['Java For Dummies', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 30: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 31: ['A Beginner's Guide', 'Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 32: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 33: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch', 'Java For Dummies']
```

```
Rule 34: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['A Beginner's Guide', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 35: ['A Beginner's Guide', 'Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 36: ['A Beginner's Guide', 'Java For Dummies', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 37: ['A Beginner's Guide'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 54.55%
Rule 38: ['Head First Java 2nd Edition'] -> ['Beginning Programming with Java']
  Support: 20.00% (count=4)  Confidence: 50.00%
Rule 39: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 50.00%
Rule 40: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 50.00%
Rule 41: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 50.00%

Process finished with exit code 0
```

B) Jupyter Notebook (brute force + Apriori + FP-Growth)

Open and run: notebooks/midtermproject.ipynb top-to-bottom:

- Environment & imports are shown first (reproducibility).

Environment

Kernel: project `.venv`. Print Python version and interpreter path for reproducibility.

[13]:
```
import sys, platform
print("Python:", platform.python_version())
print("Interpreter:", sys.executable)
```

```
Python: 3.12.6
Interpreter: C:\Users\thakr\Documents\midtermproject\.venv\Scripts\python.exe
```

Imports and project path

Add `../src` to `sys.path`, import helpers and mining functions.

[14]:
```
from pathlib import Path
import sys

# notebook is in /notebooks -> add ../src to path
root = Path.cwd().parent
sys.path.append(str(root / "src"))

from io_utils import load_transactions
from bruteforce import mine_frequent_itemsets
from rules import generate_rules

# Library wrappers (Apriori & FP-Growth)
import importlib, apriori_fp
importlib.reload(apriori_fp)
from apriori_fp import mine_with_apriori, mine_with_fpgrowth, print_rules
```

- Helper function runs the **brute-force** miner and prints L1/L2/L3/L4 plus rules and timings.

Brute-force runner

`run_scenario(csv_path, minsup, minconf)`

Loads the CSV, mines frequent itemsets (brute force), prints L1/L2/L3 with count/support, then prints final rules (A → B) and timings.
Returns the transactions so we can reuse them for Apriori/FP-Growth.

[15]:
```
from time import perf_counter

def run_scenario(csv_path, minsup, minconf):
    """
    Load CSV → mine frequent itemsets (brute force) → generate rules → print.
    """
    txns = load_transactions(csv_path)
    n_tx = len(txns)

    t0 = perf_counter()
    freq = mine_frequent_itemsets(txns, minsup)
    t1 = perf_counter()
    rules = generate_rules(freq, minconf)
    t2 = perf_counter()

    print(f"\nDataset: {csv_path.name} | transactions={n_tx} | "
          f"minsup={minsup:.0%} | minconf={minconf:.0%}")
    # Print frequent itemsets by size
    by_k = {}
    for it, sup in freq.items():
        by_k.setdefault(len(it), []).append((sorted(it), sup))
    if not by_k:
        print("No frequent itemsets at this support.")
    else:
        for k in sorted(by_k):
            print(f"\nL{k} (frequent {k}-itemsets):")
            for items, sup in sorted(by_k[k]):
                count = int(round(sup * n_tx))
                print(f"{items} | count={count} | support={sup:.2%}")

    # Print rules
    print("\nFinal Association Rules (A -> B):")
    if not rules:
        print("No rules at these thresholds.")
    else:
        for i, (A,B,s,conf) in enumerate(rules, 1):
            count = int(round(s * n_tx))
            print(f"Rule {i}: {sorted(A)} -> {sorted(B)}")
            print(f"  Support: {s:.2%} (count={count})  Confidence: {conf:.2%}")

    print(f"\nTiming: mining={t1 - t0:.4f}s  rules={t2 - t1:.4f}s")
    return txns
```

- **Runs demonstrated:**
    - Amazon **20% / 50%** (moderate thresholds):

Run 1 - Amazon (minsup=20%, minconf=50%)

Moderate thresholds. Expect multiple L2-L4 itemsets and several rules.

[16]:
```
csv = root / "data/amazon.csv"
minsup, minconf = 0.20, 0.50
txns_amazon_2050 = run_scenario(csv, minsup, minconf)
```

```
Dataset: amazon.csv | transactions=20 | minsup=20% | minconf=50%

L1 (frequent 1-itemsets):
['A Beginner's Guide'] | count=11 | support=55.00%
['Android Programming: The Big Nerd Ranch'] | count=13 | support=65.00%
['Beginning Programming with Java'] | count=6 | support=30.00%
['Head First Java 2nd Edition'] | count=8 | support=40.00%
['Java 8 Pocket Guide'] | count=4 | support=20.00%
['Java For Dummies'] | count=13 | support=65.00%
['Java: The Complete Reference'] | count=10 | support=50.00%

L2 (frequent 2-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] | count=6 | support=30.00%
['A Beginner's Guide', 'Java For Dummies'] | count=9 | support=45.00%
['A Beginner's Guide', 'Java: The Complete Reference'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Head First Java 2nd Edition'] | count=6 | support=30.00%
['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] | count=6 | support=30.00%
['Beginning Programming with Java', 'Head First Java 2nd Edition'] | count=4 | support=20.00%
['Java For Dummies', 'Java: The Complete Reference'] | count=10 | support=50.00%

L3 (frequent 3-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] | count=5 | support=25.00%
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] | count=5 | support=25.00%
['A Beginner's Guide', 'Java For Dummies', 'Java: The Complete Reference'] | count=9 | support=45.00%
['Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] | count=6 | support=30.00%

L4 (frequent 4-itemsets):
['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] | count=5 | support=25.00%
```

```
Final Association Rules (A -> B):
Rule 1: ['Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 50.00% (count=10)  Confidence: 100.00%
Rule 2: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 30.00% (count=6)  Confidence: 100.00%
Rule 3: ['A Beginner's Guide', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 100.00%
Rule 4: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 100.00%
Rule 5: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 100.00%
Rule 6: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 100.00%
Rule 7: ['Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 8: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 9: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 90.00%
Rule 10: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 11: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 12: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 13: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 14: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 15: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 83.33%
Rule 16: ['A Beginner's Guide'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 17: ['A Beginner's Guide'] -> ['Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 18: ['A Beginner's Guide'] -> ['Java For Dummies', 'Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 81.82%
Rule 19: ['Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 50.00% (count=10)  Confidence: 76.92%
Rule 20: ['Head First Java 2nd Edition'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 75.00%
Rule 21: ['Java For Dummies'] -> ['A Beginner's Guide']
  Support: 45.00% (count=9)  Confidence: 69.23%


Rule 22: ['Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 23: ['Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 24: ['Java For Dummies'] -> ['A Beginner's Guide', 'Java: The Complete Reference']
  Support: 45.00% (count=9)  Confidence: 69.23%
Rule 25: ['Beginning Programming with Java'] -> ['Head First Java 2nd Edition']
  Support: 20.00% (count=4)  Confidence: 66.67%
Rule 26: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 30.00% (count=6)  Confidence: 66.67%
Rule 27: ['Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 28: ['Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch', 'Java For Dummies']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 29: ['Java For Dummies', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 60.00%
Rule 30: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 31: ['A Beginner's Guide', 'Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 32: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['A Beginner's Guide']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 33: ['A Beginner's Guide', 'Java For Dummies'] -> ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 34: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch', 'Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 35: ['Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['A Beginner's Guide', 'Java: The Complete Reference']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 36: ['A Beginner's Guide', 'Java For Dummies', 'Java: The Complete Reference'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 55.56%
Rule 37: ['A Beginner's Guide'] -> ['Android Programming: The Big Nerd Ranch']
  Support: 30.00% (count=6)  Confidence: 54.55%
Rule 38: ['Head First Java 2nd Edition'] -> ['Beginning Programming with Java']
  Support: 20.00% (count=4)  Confidence: 50.00%
Rule 39: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 50.00%
Rule 40: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies']
  Support: 25.00% (count=5)  Confidence: 50.00%
Rule 41: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch']
  Support: 25.00% (count=5)  Confidence: 50.00%

Timing: mining=0.0011s  rules=0.0002s
```

- Amazon **70% / 70%** (strict thresholds):

Run 2 - Amazon (minsup=70%, minconf=70%)

Very strict thresholds. Likely few/no frequent pairs and no rules.

[17]:
```
csv = root / "data/amazon.csv"
minsup, minconf = 0.70, 0.70
_ = run_scenario(csv, minsup, minconf)
```

```
Dataset: amazon.csv | transactions=20 | minsup=70% | minconf=70%
No frequent itemsets at this support.

Final Association Rules (A -> B):
No rules at these thresholds.

Timing: mining=0.0001s  rules=0.0000s
```

- Walmart **20% / 50%** (grocery-style pairs):

Run 3 - Walmart (minsup=20%, minconf=50%)

Moderate thresholds on grocery-style data; expect a few L2 pairs and rules.

[18]:
```
csv = root / "data/walmart.csv"
minsup, minconf = 0.20, 0.50
txns_walmart_2050 = run_scenario(csv, minsup, minconf)
```

```
Dataset: walmart.csv | transactions=25 | minsup=20% | minconf=50%

L1 (frequent 1-itemsets):
['bread'] | count=14 | support=56.00%
['butter'] | count=6 | support=24.00%
['cereal'] | count=7 | support=28.00%
['eggs'] | count=11 | support=44.00%
['milk'] | count=10 | support=40.00%

L2 (frequent 2-itemsets):
['bread', 'eggs'] | count=6 | support=24.00%
['bread', 'milk'] | count=6 | support=24.00%

Final Association Rules (A -> B):
Rule 1: ['milk'] -> ['bread']
  Support: 24.00% (count=6)  Confidence: 60.00%
Rule 2: ['eggs'] -> ['bread']
  Support: 24.00% (count=6)  Confidence: 54.55%

Timing: mining=0.0003s  rules=0.0000s
```

- **Library methods: Apriori** and **FP-Growth** are run on the same Amazon 20/50 transactions with top 10 rules and timing.

Apriori on the same data (Amazon 20% / 50%)

Reuse the **Amazon** transactions from Run 1 with the **same thresholds** (*minsup = 20%, minconf = 50%*), then print the **top 10 rules**.
If no rules meet the threshold, a clear message is shown

*[22]:
```
t0 = perf_counter()
ap_rules = mine_with_apriori(txns, minsup, minconf)
t1 = perf_counter()

print("Apriori (top 10 rules):")
if ap_rules:
    print_rules(ap_rules, 10)
else:
    print("No rules at these thresholds.")
t2 = perf_counter()

print(f"\nTiming: apriori={t1 - t0:0.4f}s  rules={t2 - t1:0.4f}s")
```

```
Apriori (top 10 rules):
Rule 1: ['Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 50.00%  Confidence: 100.00%
Rule 2: ['A Beginner's Guide', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 45.00%  Confidence: 100.00%
Rule 3: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 45.00%  Confidence: 100.00%
Rule 4: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 30.00%  Confidence: 100.00%
Rule 5: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 25.00%  Confidence: 100.00%
Rule 6: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 25.00%  Confidence: 100.00%
Rule 7: ['Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00%  Confidence: 90.00%
Rule 8: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00%  Confidence: 90.00%
Rule 9: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 45.00%  Confidence: 90.00%
Rule 10: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 25.00%  Confidence: 83.33%

Timing: apriori=0.0381s  rules=0.0046s
```

FP-Growth on the same data (Amazon 20% / 50%)

Run **FP-Growth** from `mlxtend` on the **same transactions and thresholds** as Apriori (*minsup = 20%, minconf = 50%*).
Print the **top 10 rules**, or show a message if none qualify.

*[23]:
```
t0 = perf_counter()
fp_rules = mine_with_fpgrowth(txns, minsup, minconf)
t1 = perf_counter()

print("\nFP-Growth (top 10 rules):")
if fp_rules:
    print_rules(fp_rules, 10)
else:
    print("No rules at these thresholds.")
t2 = perf_counter()

print(f"\nTiming: fp-growth={t1 - t0:0.4f}s  rules={t2 - t1:0.4f}s")
```

```
FP-Growth (top 10 rules):
Rule 1: ['Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 50.00%  Confidence: 100.00%
Rule 2: ['A Beginner's Guide', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 45.00%  Confidence: 100.00%
Rule 3: ['A Beginner's Guide', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 45.00%  Confidence: 100.00%
Rule 4: ['Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 30.00%  Confidence: 100.00%
Rule 5: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java For Dummies'] -> ['Java: The Complete Reference']
  Support: 25.00%  Confidence: 100.00%
Rule 6: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch', 'Java: The Complete Reference'] -> ['Java For Dummies']
  Support: 25.00%  Confidence: 100.00%
Rule 7: ['Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00%  Confidence: 90.00%
Rule 8: ['Java For Dummies', 'Java: The Complete Reference'] -> ['A Beginner's Guide']
  Support: 45.00%  Confidence: 90.00%
Rule 9: ['Java: The Complete Reference'] -> ['A Beginner's Guide', 'Java For Dummies']
  Support: 45.00%  Confidence: 90.00%
Rule 10: ['A Beginner's Guide', 'Android Programming: The Big Nerd Ranch'] -> ['Java For Dummies']
  Support: 25.00%  Confidence: 83.33%

Timing: fp-growth=0.0452s  rules=0.0015s
```

5. How it Works

a) Support & confidence:

- **Support(X)**: fraction of transactions that contain X.
- **Confidence(A→B)**: support(A∪B) / support(A) – how often B appears when A appears.
- In the outputs, support is shown both as a count (rounded) and a percentage; minsup/minconf are entered as percent but applied as fractions during filtering.

b) Brute-force miner (from scratch)

1. Count L1 (all single items); keep those with support ≥ minsup.
2. Join L1 to form L2 candidates; count support by scanning transactions; keep frequent pairs.
3. Repeat k → k+1 until a level has no new frequent itemsets.
4. For each frequent set X (|X|≥2), enumerate non-empty A ⊂ X, let B = X\A.
    - Keep A → B if confidence ≥ minconf.
5. Print grouped lists: L1, L2, L3… and Final Association Rules.

c) Library methods (mlxtend)

- Apriori: generates and prunes candidates level-by-level using the Apriori property - any superset of an infrequent set is infrequent. Hence, many candidates are eliminated early.
- FP-Growth: compresses data into an FP-tree and grows patterns without explicit candidate generation.
- Transactions are converted to a one-hot boolean DataFrame; dtype is explicitly set to bool.

6. Results and Observations:

a) Threshold Sensitivity:

- **Why thresholds matter:**
  - *Support* filters by how common a pattern is. High minsup throws out anything that isn't very frequent.
  - *Confidence* filters by how reliable a rule is when the left-hand side occurs.
- **Amazon (70% / 70%) - essentially "too strict":**
  - With only 20 transactions, demanding ≥70% support means at least 14 co-occurrences for any pair—rare even for popular book combos.
  - Result: no frequent itemsets at k≥2 ⇒ no rules.
- **Amazon (20% / 50%) - a "sweet spot":**
  - At minsupp=20% (≥4 of 20 transactions), common pairings survive.
  - Expect multiple L2/L3 itemsets and high-confidence rules that reflect co-purchased titles (e.g., beginner programming books).
  - Typical strengths: 25–50% support, confidence often ≈100% for top rules, which matches the intuition that certain titles are bought together.

- **Walmart (20% / 50%) - classic grocery patterns:**
  - Realistic grocery-style pairs emerge, e.g.:
    - ['milk'] → ['bread'] - Support 24%, Confidence 60%
    - ['eggs'] → ['bread'] - Support 24%, Confidence ≈55%
  - These are not "certainty" rules; rather, they indicate useful lift in probability and common co-occurrence at store scale.

- **Takeaway:**
  - Tight thresholds can wipe out signal completely; moderate thresholds reveal stable, interpretable itemsets.
  - If nothing shows up, the first knob to turn is minsup/minconf - *not* the algorithm.

b) Agreement across methods (brute force vs. Apriori vs. FP-Growth)

- On Amazon (20% / 50%), the library methods (Apriori via mlxtend and FP-Growth) match the from-scratch brute-force rules (differences only in ordering/duplicates).
- This is a sanity check for correctness: the custom miner enumerates 1-, 2-, 3-itemsets and filters by the same thresholds, then generates rules the same way libraries do.

- Practically, this means the implementation and thresholds, not the library - drive the result set on these datasets.

c) Timing snapshot

- Apriori: mining ≈ 0.038 s, rules printing ≈ 0.0046 s
- FP-Growth: mining ≈ 0.045 s, rules printing ≈ 0.0015 s
- Why times look so small: the datasets have 20–25 transactions, so even naive enumeration is fast.
- Algorithmic note:
    - Brute force scales poorly in general (combinatorial explosion), but is fine here.
    - Apriori prunes by support level (downward closure).
- FP-Growth compresses the dataset and often scales best when itemsets get longer or data gets larger.

So, for small, clean datasets, all three are effectively instantaneous; threshold choice dominates whether meaningful rules appear. Scaling differences would show up with larger or denser data.

d) Overall interpretation

- If no rules appear, lower minsup, then minconf, in that order.
- When support is moderate (≈20–30%) and confidence is ≥50%, results tend to be stable across algorithms and easy to interpret.
- Brute force is useful as a teaching/validation baseline; for larger data, one must switch to Apriori/FP-Growth.
- Clean, normalized items (and removal of duplicates/empties) lead to clearer, more reliable patterns.
- Pruning ultra-rare items keeps the search focused and improves rule quality without losing meaningful signal.