Midterm_Project_Report

October 14, 2025

Student Name: Lucas Marques Balbi

Email: lb278@njit.edu

Course: CS634 - Data Mining

Instructor: Dr. Yasser Adbduallah

GitHub Repository: https://github.com/lubalbi/Association-Algorithms-Brute-force-from-scratch-Apriori-and-FP-Growth

1 Introduction

1.1 Problem Statement

The project goal is to find customer frequent patterns among items from retail transactions datasets and generate association rules based on support and confidence values provided by user.

1.2 Project Goals and Scope

Implementation of brute-force algorithm from scratch to find frequent itemsets and generate association rules. We will also be using Python libraries for the Apriori and FP-Growth algorithms.

1.3 Methodology Overview

We will go over in details about the following procedures:

- Creating dataset;
- Brute-force algorithm;
- Apriori and FP-growth implementation;
- Results and conclusion

1.4 System packages requirements

It is necessary to install the following language code and libraries in your system before running the code

- Python 3.9+
- Pandas
  - pip install pandas
- Numpy

  - pip install numpy

- Apyori
  - pip install apyori

- Mlxtend
  - pip install mlxtend

1.5 GitHub Repository Structure

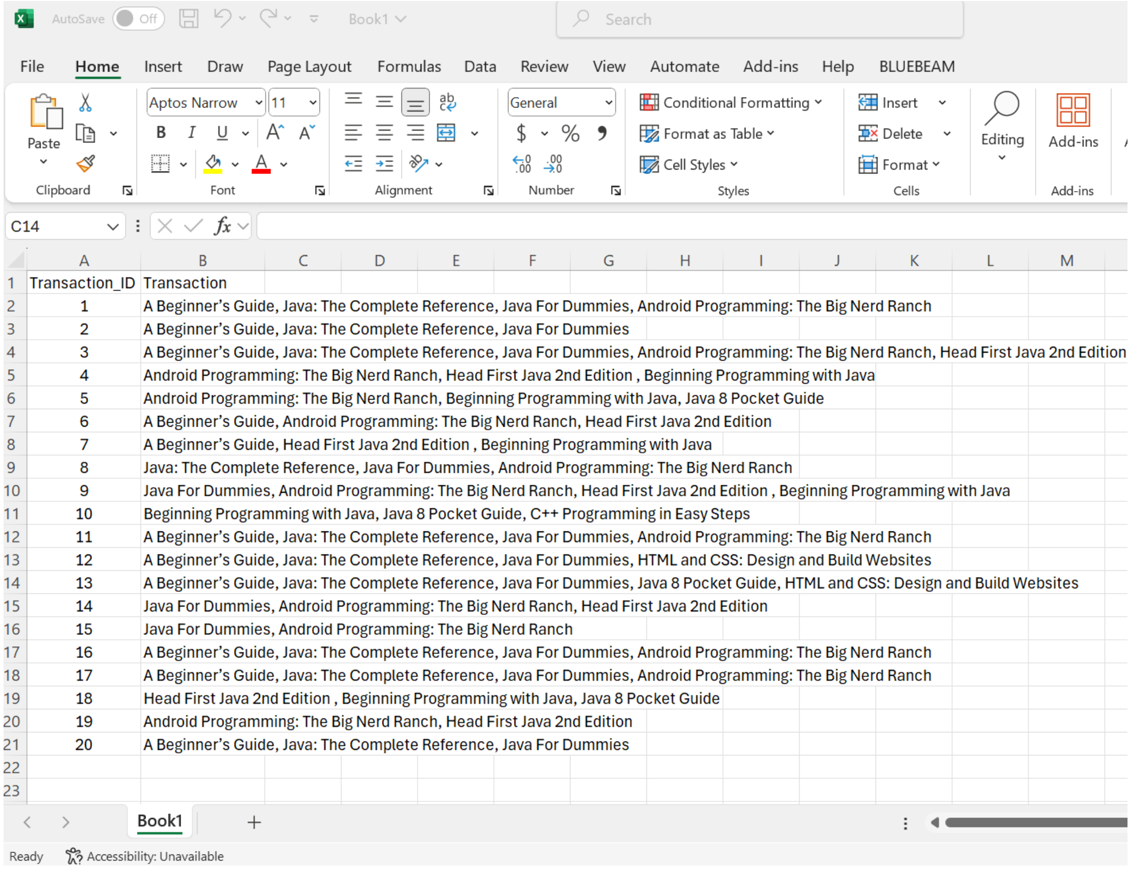
- **Running_all_the_algorithms.ipynb** - The main Jupyter Notebook for the project.
- **Midterm_Project_Report.pdf** - A detailed report and tutorial for the project.
- **readme.txt** - Contains setup instructions and project requirements.
- **Standalone Scripts:**
  - BruteForce.py - Runs only the Brute-Force algorithm from the command line.
  - Apriori.py - Runs only the Apriori algorithm from the command line.
  - FPGrowth.py - Runs only the FP-Growth algorithm from the command line.
- **/Data/** - Folder containing the datasets.
  - amazon.csv
  - bestbuy.csv
  - kmart.csv
  - nike.csv
  - walmart.csv
- **/Screenshots/** - Folder containing relevant screenshots.
  - 01_DataCreation01.png
  - 01_DataCreation02.png
  - 02_Standalone_BruteForce_Script.png
  - 02_Standalone_Apriori_Script.png
  - 02_Standalone_FPGrowth_Script.png

2 Dataset Creation

2.1 Creating data file on excel

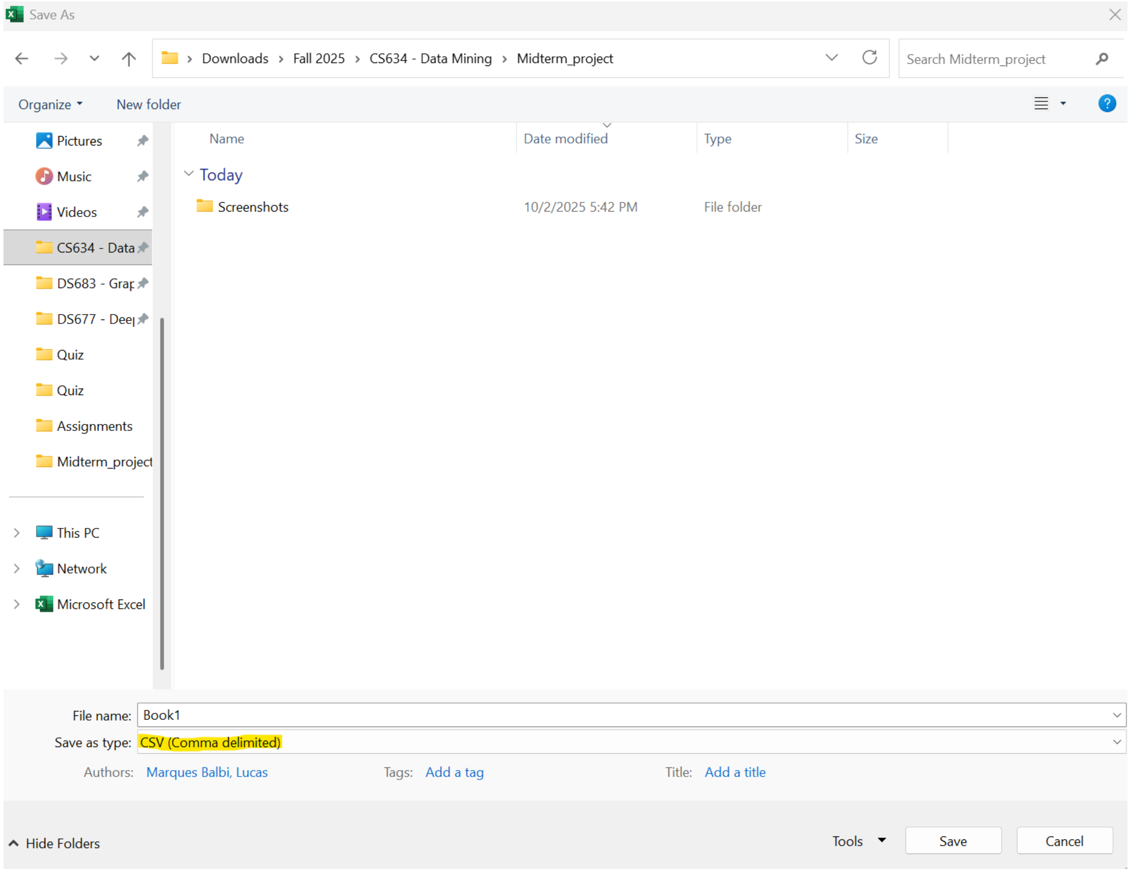
We are going to create 5 databases. It was given 4 databases from the project template file and the last one we will come up with the elements and transactions data.

First step is to create the headers which in our case will be Transaction_ID and Transaction. The transaction id column will be from 1 to 20 because we are going to use 20 transaction datapoints. The transaction column will be filled with the products that a costumer would be buying from a certain retail store. We are going to copy/paste the transactions for the amazon, kmart, bestbuy and nike dataset. The walmart is the only database we will comeup with the transactions by ourselves. Keep in mind that this is a determistic situation not random, so our data has to be estabished at the beginning as we are doing



Transaction_ID	Transaction
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 Java 2nd Edition
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 Websites
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

When you save each dataset make sure you are saving the file using the correct extension (.csv)



Downloads › Fall 2025 › CS634 - Data Mining › Midterm_project

Name	Date modified	Type	Size
Today			
Screenshots	10/2/2025 5:42 PM	File folder	

File name: Book1

Save as type: CSV (Comma delimited)

Authors: Marques Balbi, Lucas   Tags: Add a tag   Title: Add a title

Additional comments: A real life application of this preliminary procedure would be implementing a web scraping. The benefict of the a web scraping is to exctract data from websites.

2.2 Loading data

```
[394]: import pandas as pd
       import numpy as np
       import time
       import os
```

```
[395]: retail_store = "amazon"

       file_path = os.path.join('Data', f'{retail_store}.csv')

       db_amazon = pd.read_csv(file_path, encoding='cp1252')
       db_amazon
```

```
[395]:     Transaction_ID                                   Transaction
       0                1  A Beginner’s Guide, Java: The Complete Referen…
       1                2  A Beginner’s Guide, Java: The Complete Referen…
       2                3  A Beginner’s Guide, Java: The Complete Referen…
       3                4  Android Programming: The Big Nerd Ranch, Head …
```

```
4                   5  Android Programming: The Big Nerd Ranch, Begin…
5                   6  A Beginner’s Guide, Android Programming: The B…
6                   7  A Beginner’s Guide, Head First Java 2nd Editio…
7                   8  Java: The Complete Reference, Java For Dummies…
8                   9  Java For Dummies, Android Programming: The Big…
9                  10  Beginning Programming with Java, Java 8 Pocket…
10                 11  A Beginner’s Guide, Java: The Complete Referen…
11                 12  A Beginner’s Guide, Java: The Complete Referen…
12                 13  A Beginner’s Guide, Java: The Complete Referen…
13                 14  Java For Dummies, Android Programming: The Big…
14                 15  Java For Dummies, Android Programming: The Big…
15                 16  A Beginner’s Guide, Java: The Complete Referen…
16                 17  A Beginner’s Guide, Java: The Complete Referen…
17                 18  Head First Java 2nd Edition , Beginning Progra…
18                 19  Android Programming: The Big Nerd Ranch, Head …
19                 20  A Beginner’s Guide, Java: The Complete Referen…
```

[396]:
```
#clean up header by removing whitespace
db_amazon.columns = db_amazon.columns.str.strip()
db_amazon
```

[396]:
```
    Transaction_ID                                         Transaction
0                   1  A Beginner’s Guide, Java: The Complete Referen…
1                   2  A Beginner’s Guide, Java: The Complete Referen…
2                   3  A Beginner’s Guide, Java: The Complete Referen…
3                   4  Android Programming: The Big Nerd Ranch, Head …
4                   5  Android Programming: The Big Nerd Ranch, Begin…
5                   6  A Beginner’s Guide, Android Programming: The B…
6                   7  A Beginner’s Guide, Head First Java 2nd Editio…
7                   8  Java: The Complete Reference, Java For Dummies…
8                   9  Java For Dummies, Android Programming: The Big…
9                  10  Beginning Programming with Java, Java 8 Pocket…
10                 11  A Beginner’s Guide, Java: The Complete Referen…
11                 12  A Beginner’s Guide, Java: The Complete Referen…
12                 13  A Beginner’s Guide, Java: The Complete Referen…
13                 14  Java For Dummies, Android Programming: The Big…
14                 15  Java For Dummies, Android Programming: The Big…
15                 16  A Beginner’s Guide, Java: The Complete Referen…
16                 17  A Beginner’s Guide, Java: The Complete Referen…
17                 18  Head First Java 2nd Edition , Beginning Progra…
18                 19  Android Programming: The Big Nerd Ranch, Head …
19                 20  A Beginner’s Guide, Java: The Complete Referen…
```

[397]:
```
#summary report from the data
db_amazon.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
```

```
Data columns (total 2 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Transaction_ID  20 non-null     int64 
 1   Transaction     20 non-null     object
dtypes: int64(1), object(1)
memory usage: 452.0+ bytes
```

[398]:
```
#split list of elements from transaction column by comma delimitation
db_amazon['Transaction'] = db_amazon['Transaction'].str.split(',')
db_amazon
```

[398]:
```
    Transaction_ID                                        Transaction
0                1  [A Beginner’s Guide,  Java: The Complete Refer…
1                2  [A Beginner’s Guide,  Java: The Complete Refer…
2                3  [A Beginner’s Guide,  Java: The Complete Refer…
3                4  [Android Programming: The Big Nerd Ranch,  Hea…
4                5  [Android Programming: The Big Nerd Ranch,  Beg…
5                6  [A Beginner’s Guide,  Android Programming: The…
6                7  [A Beginner’s Guide,  Head First Java 2nd Edit…
7                8  [Java: The Complete Reference,  Java For Dummi…
8                9  [Java For Dummies,  Android Programming: The B…
9               10  [Beginning Programming with Java,  Java 8 Pock…
10              11  [A Beginner’s Guide,  Java: The Complete Refer…
11              12  [A Beginner’s Guide,  Java: The Complete Refer…
12              13  [A Beginner’s Guide,  Java: The Complete Refer…
13              14  [Java For Dummies,  Android Programming: The B…
14              15  [Java For Dummies,  Android Programming: The B…
15              16  [A Beginner’s Guide,  Java: The Complete Refer…
16              17  [A Beginner’s Guide,  Java: The Complete Refer…
17              18  [Head First Java 2nd Edition ,  Beginning Prog…
18              19  [Android Programming: The Big Nerd Ranch,  Hea…
19              20  [A Beginner’s Guide,  Java: The Complete Refer…
```

[399]:
```
#remove any whitespace in the elements from each transaction
db_amazon['Transaction'] = db_amazon['Transaction'].apply(lambda x: [item.
 ↪strip() for item in x])
db_amazon
```

[399]:
```
    Transaction_ID                                        Transaction
0                1  [A Beginner’s Guide, Java: The Complete Refere…
1                2  [A Beginner’s Guide, Java: The Complete Refere…
2                3  [A Beginner’s Guide, Java: The Complete Refere…
3                4  [Android Programming: The Big Nerd Ranch, Head…
4                5  [Android Programming: The Big Nerd Ranch, Begi…
5                6  [A Beginner’s Guide, Android Programming: The …
6                7  [A Beginner’s Guide, Head First Java 2nd Editi…
7                8  [Java: The Complete Reference, Java For Dummie…
```

```
8                9  [Java For Dummies, Android Programming: The Bi…
9               10  [Beginning Programming with Java, Java 8 Pocke…
10              11  [A Beginner’s Guide, Java: The Complete Refere…
11              12  [A Beginner’s Guide, Java: The Complete Refere…
12              13  [A Beginner’s Guide, Java: The Complete Refere…
13              14  [Java For Dummies, Android Programming: The Bi…
14              15  [Java For Dummies, Android Programming: The Bi…
15              16  [A Beginner’s Guide, Java: The Complete Refere…
16              17  [A Beginner’s Guide, Java: The Complete Refere…
17              18  [Head First Java 2nd Edition, Beginning Progra…
18              19  [Android Programming: The Big Nerd Ranch, Head…
19              20  [A Beginner’s Guide, Java: The Complete Refere…
```

[400]:

```
#create list of items
amazon_list = []
for i in range(len(db_amazon['Transaction'])):
  for j in range(len(db_amazon['Transaction'][i])):
    if db_amazon['Transaction'][i][j] not in amazon_list:
      amazon_list.append(db_amazon['Transaction'][i][j])
print(amazon_list)

#how many itens on amazon list
print(len(amazon_list))
```

```
['A Beginner’s Guide', 'Java: The Complete Reference', 'Java For Dummies',
'Android Programming: The Big Nerd Ranch', 'Head First Java 2nd Edition',
'Beginning Programming with Java', 'Java 8 Pocket Guide', 'C++ Programming in
Easy Steps', 'HTML and CSS: Design and Build Websites']
9
```

There is one item (book) that was not sold because our list of itens had 10 elements

[402]:

```
#testing the data to make sure we will be able to retrieve necessary info
#what is the first transaction
print(db_amazon['Transaction'][0])

#what is the element of the second transaction
print(db_amazon['Transaction'][0][1])
```

```
['A Beginner’s Guide', 'Java: The Complete Reference', 'Java For Dummies',
'Android Programming: The Big Nerd Ranch']
Java: The Complete Reference
```

[403]:

```
#create list of transactions
amazon_transactions = []
for i in range(len(db_amazon['Transaction'])):
  amazon_transactions.append(db_amazon['Transaction'][i])
print(amazon_transactions)
```

```
[['A Beginner’s Guide', 'Java: The Complete Reference', 'Java For Dummies',
'Android Programming: The Big Nerd Ranch'], ['A Beginner’s Guide', 'Java: The
Complete Reference', 'Java For Dummies'], ['A Beginner’s Guide', 'Java: The
Complete Reference', 'Java For Dummies', 'Android Programming: The Big Nerd
Ranch', 'Head First Java 2nd Edition'], ['Android Programming: The Big Nerd
Ranch', 'Head First Java 2nd Edition', 'Beginning Programming with Java'],
['Android Programming: The Big Nerd Ranch', 'Beginning Programming with Java',
'Java 8 Pocket Guide'], ['A Beginner’s Guide', 'Android Programming: The Big
Nerd Ranch', 'Head First Java 2nd Edition'], ['A Beginner’s Guide', 'Head First
Java 2nd Edition', 'Beginning Programming with Java'], ['Java: The Complete
Reference', 'Java For Dummies', 'Android Programming: The Big Nerd Ranch'],
['Java For Dummies', 'Android Programming: The Big Nerd Ranch', 'Head First Java
2nd Edition', 'Beginning Programming with Java'], ['Beginning Programming with
Java', 'Java 8 Pocket Guide', 'C++ Programming in Easy Steps'], ['A Beginner’s
Guide', 'Java: The Complete Reference', 'Java For Dummies', 'Android
Programming: The Big Nerd Ranch'], ['A Beginner’s Guide', 'Java: The Complete
Reference', 'Java For Dummies', 'HTML and CSS: Design and Build Websites'], ['A
Beginner’s Guide', 'Java: The Complete Reference', 'Java For Dummies', 'Java 8
Pocket Guide', 'HTML and CSS: Design and Build Websites'], ['Java For Dummies',
'Android Programming: The Big Nerd Ranch', 'Head First Java 2nd Edition'],
['Java For Dummies', 'Android Programming: The Big Nerd Ranch'], ['A Beginner’s
Guide', 'Java: The Complete Reference', 'Java For Dummies', 'Android
Programming: The Big Nerd Ranch'], ['A Beginner’s Guide', 'Java: The Complete
Reference', 'Java For Dummies', 'Android Programming: The Big Nerd Ranch'],
['Head First Java 2nd Edition', 'Beginning Programming with Java', 'Java 8
Pocket Guide'], ['Android Programming: The Big Nerd Ranch', 'Head First Java 2nd
Edition'], ['A Beginner’s Guide', 'Java: The Complete Reference', 'Java For
Dummies']]
```

Our data is ready to be imported to our algorithms! Lets create a function to combine all the previous processes in one single function

[405]:

```
#create preprocessing function to clean up raw data
def preprocess(db):
  #clean up header by removing whitespace
  db.columns = db.columns.str.strip()

  #split list of elements from transaction column by comma delimitation
  db['Transaction'] = db['Transaction'].str.split(',')

  #remove any whitespace in the elements from each transaction
  db['Transaction'] = db['Transaction'].apply(lambda x: [item.strip() for item␣
 ↪in x])

  #create list of itens
  item_list = []
  for i in range(len(db['Transaction'])):
    for j in range(len(db['Transaction'][i])):
      if db['Transaction'][i][j] not in item_list:
        item_list.append(db['Transaction'][i][j])

  # #create list of transactions
  # transactions = []
  # for i in range(len(db['Transaction'])):
  #   transactions.append(db['Transaction'][i])

  return db, item_list
```

3 User Input

Before we run the models we need to establish the values of support and confidence.

The support measures the frequency of an item or itemset occurs. It will measure the popularity of itemset in relation to the transactions. For example, we have total transactions of 10 and itens X and Z can be found together in 3 transactions. In this case the support is $3/10 = 30\%$

The confidence will provide how likelihood that the items purchased together. It will measure the correlation of the itens. For example, we have total transaction of 10 and the itens X and Z can found together in 3 transactions but item X can be found in a total of 5 transactions. The X->Z confidence in this case will be $3/5 = 60\%$

In short, high support value improves reliability by filtering out rare patterns, while high confidence value improves prediction by filtering out unreliable associations.

[408]:

```
#list of retail store corresponding to index 0 to 4
all_retail_stores = ('amazon', 'kmart', 'bestbuy', 'nike', 'walmart')

print("Welcome to Apriori 2.0!")
print("Please select one of the following retail store (type corresponding␣
 ↪number only):")
#print out number 1. and the first retail store from the list and so on
for i in range(len(all_retail_stores)):
  print(str(i + 1) + ". " + all_retail_stores[i])
print("6. Exit")

########################################
'''Input of the retail store number'''
########################################
user_input = input()



try:
    #make sure the input is a number
    retail_store_number = int(user_input)
    #check if user input is valid (int number between 1 to 5)
    if 1 <= retail_store_number <= 5:
      retail_store = all_retail_stores[retail_store_number - 1]
      print("You selected " + retail_store + "!")

      #########################################
      '''Input of the minimum support value'''
      #########################################
      #ask user to provide minimum support and store it
      print("Please enter the percentage of minimum support(values between 1 and 100)")
      try:
        min_support = int(input())
        if 1 <= min_support <= 100:
          print("Minimum support is " + str(min_support) + "%")

          ##########################################################
          '''Input of the minimum confidence value'''
          ##########################################################
          #ask user to provide minimum confidence and store it
          print("Please enter the percentage of minimum confidence(values between 1 and 100)")
          try:
            min_confidence = int(input())
            if 1 <= min_confidence <= 100:
              print("Minimum confidence is " + str(min_confidence) + "%")

            #error message if wrong number for confidence input
            else:
              print("Invalid Confidence value input. Please try again.")
              exit
          #error message if not a number for confidence input
          except ValueError:
            print("Invalid Confidence value input. Please try again.")
            exit
          ##########################################################
          '''End of validation of the entered minimum confidence'''
          ##########################################################


        #error message if wrong number for support input
        else:
          print("Invalid Support value input. Please try again.")
          exit
      #error message if not a number for support input
      except ValueError:
        print("Invalid Support value input. Please try again.")
```

```
        exit
      ########################################
      '''End of validation of the entered minimum support value'''
      ########################################


    #check if user wants to exit (number 6)
    elif int(user_input) == 6:
      print("Thank you for using Apriori 2.0!")
      exit

    #if the input is not a valid number it will come out an error message
    else:
      print("Invalid Retail store input. Please try again.")
      exit

#if the input is not a number it will come out an error message
except ValueError:
  print("Invalid Retail store input. Please try again.")
  exit

########################################
'''End of validation of the entered retail store number'''
########################################



#Load dataset selected
db_raw = pd.read_csv('Data/' + ( retail_store + '.csv'), encoding='cp1252')
#Preprocess the data
database, item_list = preprocess(db_raw)

print(item_list)
print(database)
```

```
Welcome to Apriori 2.0!
Please select one of the following retail store (type corresponding number
only):
1. amazon
2. kmart
3. bestbuy
4. nike
5. walmart
6. Exit

 1

You selected amazon!
Please enter the percentage of minimum support(values between 1 and 100)
```

```
 50

Minimum support is 50%
Please enter the percentage of minimum confidence(values between 1 and 100)

 50

Minimum confidence is 50%
['A Beginner's Guide', 'Java: The Complete Reference', 'Java For Dummies',
'Android Programming: The Big Nerd Ranch', 'Head First Java 2nd Edition',
'Beginning Programming with Java', 'Java 8 Pocket Guide', 'C++ Programming in
Easy Steps', 'HTML and CSS: Design and Build Websites']
    Transaction_ID                                        Transaction
0                1  [A Beginner's Guide, Java: The Complete Refere…
1                2  [A Beginner's Guide, Java: The Complete Refere…
2                3  [A Beginner's Guide, Java: The Complete Refere…
3                4  [Android Programming: The Big Nerd Ranch, Head…
4                5  [Android Programming: The Big Nerd Ranch, Begi…
5                6  [A Beginner's Guide, Android Programming: The …
6                7  [A Beginner's Guide, Head First Java 2nd Editi…
7                8  [Java: The Complete Reference, Java For Dummie…
8                9  [Java For Dummies, Android Programming: The Bi…
9               10  [Beginning Programming with Java, Java 8 Pocke…
10              11  [A Beginner's Guide, Java: The Complete Refere…
11              12  [A Beginner's Guide, Java: The Complete Refere…
12              13  [A Beginner's Guide, Java: The Complete Refere…
13              14  [Java For Dummies, Android Programming: The Bi…
14              15  [Java For Dummies, Android Programming: The Bi…
15              16  [A Beginner's Guide, Java: The Complete Refere…
16              17  [A Beginner's Guide, Java: The Complete Refere…
17              18  [Head First Java 2nd Edition, Beginning Progra…
18              19  [Android Programming: The Big Nerd Ranch, Head…
19              20  [A Beginner's Guide, Java: The Complete Refere…
```

4 Brute Force Algorithm

This algorithm will evaluate every possible combination of itens for the provided support and confidence values by the user. This algorithm takes more time to find all the itens combinations that equal or greater than the two metric parameters.

First step is to find all the possible unique combinations of the elements. As we create the combinations we will count how many transactions have that specific combination. The computed number is the frequency of the combination thru all the transactions.

Itemset level is defined as the number of itens we are considering for the all possible combinations. For example, the itemset 1 has one only iten purchased, then itemset 2 has all the possible combinations of purchase of two itens.

We can stop the process when we get to an itemset level where all the combinations have frequency equals to zero because for sure the next level will not have any frequency as well.

[458]:

```
from itertools import combinations
from collections import Counter

#set initial time
start_time_bf = time.time()

#create a dictionary with all the possible unique combinations of itens
all_combinations = {}

'''Initial ideal was to implement loop for each item set
but since we have 9 or 10 different itens it will be very mannual code as you␣
 ↪can see below.
The commented out code below shows the iterations up to three itens.
We will use an alternative way that will save us time and effort.
The 'combinations' function from itertool library will perform the same idea of␣
 ↪what it is written below but for all the itens



#add single itens to dictionary
for item in item_list:
  all_combinations[item] = 0

#add unique pairs of itens to dictionary
for i in range(len(item_list)):
  for j in range(i + 1, len(item_list)):
    all_combinations[item_list[i] + " ; " + item_list[j]] = 0

#add unique combinations of three itens to dictionary
for i in range(len(item_list)):
  for j in range(i + 1, len(item_list)):
    for k in range(j + 1, len(item_list)):
      all_combinations[item_list[i] + " ; " + item_list[j] + " ; " +␣
 ↪item_list[k]] = 0
'''

#define the max number of itemset
max_item_set = len(item_list)

#Start loop thru the each itemset
for item_set in range(1, max_item_set + 1):

  #Add a flag to make sure we can find at least one combination for this␣
 ↪itemset level
  itemset_with_combinations = False

  print(f"Processing {item_set} itemset")
  #compute all the unique itens combination possible for this itemset level
  current_itemset_combinations = {}
  for item_combination in combinations(item_list, item_set):
    key = ' ; '.join(item_combination)
    current_itemset_combinations[key] = 0

  #Count the frequencies for this itemset level
  for transaction in database['Transaction']:
    for item in current_itemset_combinations:
        if set(item.split(' ; ')).issubset(set(transaction)):
            current_itemset_combinations[item] += 1

  #check if there is combinations found in this itemset
  #and record to dictionary with all combinations
  for key, value in current_itemset_combinations.items():
      if value > 0:
          all_combinations[key] = value
          itemset_with_combinations = True

  #if there is no combinations found in this itemset
  if not itemset_with_combinations:
      print(f"No frequent itemsets found for #{item_set} itemset")
      break


#print out the quantity of elements in the dictionary
print("\nThe dictionary with all the combinations has", len(all_combinations),␣
 ↪"different elements")

#print all the possible combinations until first itemset level with all␣
 ↪combination zeros
#print all the frequencies
for key, value in all_combinations.items():
    print(f"{key}: {value}")
```

```
Processing 1 itemset
Processing 2 itemset
Processing 3 itemset
Processing 4 itemset
Processing 5 itemset
Processing 6 itemset
No frequent itemsets found for #6 itemset

The dictionary with all the combinations has 74 different elements
A Beginner's Guide: 11
Java: The Complete Reference: 10
Java For Dummies: 13
```

```
Android Programming: The Big Nerd Ranch: 13
Head First Java 2nd Edition: 8
Beginning Programming with Java: 6
Java 8 Pocket Guide: 4
C++ Programming in Easy Steps: 1
HTML and CSS: Design and Build Websites: 2
A Beginner’s Guide ; Java: The Complete Reference: 9
A Beginner’s Guide ; Java For Dummies: 9
A Beginner’s Guide ; Android Programming: The Big Nerd Ranch: 6
A Beginner’s Guide ; Head First Java 2nd Edition: 3
A Beginner’s Guide ; Beginning Programming with Java: 1
A Beginner’s Guide ; Java 8 Pocket Guide: 1
A Beginner’s Guide ; HTML and CSS: Design and Build Websites: 2
Java: The Complete Reference ; Java For Dummies: 10
Java: The Complete Reference ; Android Programming: The Big Nerd Ranch: 6
Java: The Complete Reference ; Head First Java 2nd Edition: 1
Java: The Complete Reference ; Java 8 Pocket Guide: 1
Java: The Complete Reference ; HTML and CSS: Design and Build Websites: 2
Java For Dummies ; Android Programming: The Big Nerd Ranch: 9
Java For Dummies ; Head First Java 2nd Edition: 3
Java For Dummies ; Beginning Programming with Java: 1
Java For Dummies ; Java 8 Pocket Guide: 1
Java For Dummies ; HTML and CSS: Design and Build Websites: 2
Android Programming: The Big Nerd Ranch ; Head First Java 2nd Edition: 6
Android Programming: The Big Nerd Ranch ; Beginning Programming with Java: 3
Android Programming: The Big Nerd Ranch ; Java 8 Pocket Guide: 1
Head First Java 2nd Edition ; Beginning Programming with Java: 4
Head First Java 2nd Edition ; Java 8 Pocket Guide: 1
Beginning Programming with Java ; Java 8 Pocket Guide: 3
Beginning Programming with Java ; C++ Programming in Easy Steps: 1
Java 8 Pocket Guide ; C++ Programming in Easy Steps: 1
Java 8 Pocket Guide ; HTML and CSS: Design and Build Websites: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies: 9
A Beginner’s Guide ; Java: The Complete Reference ; Android Programming: The Big
Nerd Ranch: 5
A Beginner’s Guide ; Java: The Complete Reference ; Head First Java 2nd Edition:
1
A Beginner’s Guide ; Java: The Complete Reference ; Java 8 Pocket Guide: 1
A Beginner’s Guide ; Java: The Complete Reference ; HTML and CSS: Design and
Build Websites: 2
A Beginner’s Guide ; Java For Dummies ; Android Programming: The Big Nerd Ranch:
5
A Beginner’s Guide ; Java For Dummies ; Head First Java 2nd Edition: 1
A Beginner’s Guide ; Java For Dummies ; Java 8 Pocket Guide: 1
A Beginner’s Guide ; Java For Dummies ; HTML and CSS: Design and Build Websites:
2
A Beginner’s Guide ; Android Programming: The Big Nerd Ranch ; Head First Java
2nd Edition: 2
```

```
A Beginner’s Guide ; Head First Java 2nd Edition ; Beginning Programming with
Java: 1
A Beginner’s Guide ; Java 8 Pocket Guide ; HTML and CSS: Design and Build
Websites: 1
Java: The Complete Reference ; Java For Dummies ; Android Programming: The Big
Nerd Ranch: 6
Java: The Complete Reference ; Java For Dummies ; Head First Java 2nd Edition: 1
Java: The Complete Reference ; Java For Dummies ; Java 8 Pocket Guide: 1
Java: The Complete Reference ; Java For Dummies ; HTML and CSS: Design and Build
Websites: 2
Java: The Complete Reference ; Android Programming: The Big Nerd Ranch ; Head
First Java 2nd Edition: 1
Java: The Complete Reference ; Java 8 Pocket Guide ; HTML and CSS: Design and
Build Websites: 1
Java For Dummies ; Android Programming: The Big Nerd Ranch ; Head First Java 2nd
Edition: 3
Java For Dummies ; Android Programming: The Big Nerd Ranch ; Beginning
Programming with Java: 1
Java For Dummies ; Head First Java 2nd Edition ; Beginning Programming with
Java: 1
Java For Dummies ; Java 8 Pocket Guide ; HTML and CSS: Design and Build
Websites: 1
Android Programming: The Big Nerd Ranch ; Head First Java 2nd Edition ;
Beginning Programming with Java: 2
Android Programming: The Big Nerd Ranch ; Beginning Programming with Java ; Java
8 Pocket Guide: 1
Head First Java 2nd Edition ; Beginning Programming with Java ; Java 8 Pocket
Guide: 1
Beginning Programming with Java ; Java 8 Pocket Guide ; C++ Programming in Easy
Steps: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; Android
Programming: The Big Nerd Ranch: 5
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; Head
First Java 2nd Edition: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; Java 8
Pocket Guide: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; HTML and
CSS: Design and Build Websites: 2
A Beginner’s Guide ; Java: The Complete Reference ; Android Programming: The Big
Nerd Ranch ; Head First Java 2nd Edition: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java 8 Pocket Guide ; HTML
and CSS: Design and Build Websites: 1
A Beginner’s Guide ; Java For Dummies ; Android Programming: The Big Nerd Ranch
; Head First Java 2nd Edition: 1
A Beginner’s Guide ; Java For Dummies ; Java 8 Pocket Guide ; HTML and CSS:
Design and Build Websites: 1
Java: The Complete Reference ; Java For Dummies ; Android Programming: The Big
Nerd Ranch ; Head First Java 2nd Edition: 1
```

```
Java: The Complete Reference ; Java For Dummies ; Java 8 Pocket Guide ; HTML and
CSS: Design and Build Websites: 1
Java For Dummies ; Android Programming: The Big Nerd Ranch ; Head First Java 2nd
Edition ; Beginning Programming with Java: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; Android
Programming: The Big Nerd Ranch ; Head First Java 2nd Edition: 1
A Beginner’s Guide ; Java: The Complete Reference ; Java For Dummies ; Java 8
Pocket Guide ; HTML and CSS: Design and Build Websites: 1
```

Now that we have all the possible combinations and its frequency, our next step will be to compute the support for each element from the dictionary.Then, we will create a support dictionary that includes only the elements that have a support equal or greater than the one entered by the user.

Support calculation is just the number of transactions that can has a certain element from the dictionary divide by the total number of transactions from the dataset. We will come out with the frequency of single item from the dictionary.

[460]:

```python
#define a dictionary for support requirement achieved
support_dict = {}

#define total number of transactions
total_transactions = len(database)

#compute support and add to the support dictionary
for key, value in all_combinations.items():
  support = value / total_transactions
  if support >= (min_support / 100):
    support_dict[key] = support

#title for output
print("Support Dictionary")

#print out the support dictionary
for key, value in support_dict.items():
  print(f"{key}: {value}")
```

```
Support Dictionary
A Beginner’s Guide: 0.55
Java: The Complete Reference: 0.5
Java For Dummies: 0.65
Android Programming: The Big Nerd Ranch: 0.65
Java: The Complete Reference ; Java For Dummies: 0.5
```

Final step is to find the confidence for all possible combinations of the elements from the support dictionary

[462]:

```python
from itertools import permutations

#define confidence dictionary
```

```
confidence_dict = {}

#create function to check all the permutations between itens in a transaction
def find_key(items, dictionary):

    #for one element in the transaction
    if len(items) == 1:
        key = items[0]
        return key if key in dictionary else None

    #for more than 1 element in the transaction
    for p in permutations(items):
        key = ' ; '.join(p)
        if key in dictionary:
            return key

    #for no key found
    return None

#compute confidence for each element from the support dictionary
for support_itemset, support_value in support_dict.items():
  #split into different elements in each itemset
  #before:  Java: The Complete Reference ; Java For Dummies
  #now:   ['Java: The Complete Reference', 'Java For Dummies'])
  support_item = support_itemset.split(' ; ')

  # create rule to to deal with itemset with 2 or more elements, others can be skipped
  if len(support_item) < 2:
    continue

  #compute all combinations of itens (X->Y)
  for i in range(1, len(support_item)):
    #find X(antecedent)
    for combination_x in combinations(support_item, i):

        #find Y(consequent)
        combination_y = list(set(support_item) - set(combination_x))

        #find key for x and y if they exists
        xkey = find_key(list(combination_x), support_dict)
        ykey = find_key(list(combination_y), support_dict)

        #continue if both keys exists
        if xkey and ykey:
            #retrieve support count for X(antecedent)
            support_x = support_dict[xkey]
            #retrieve support count for Y(consequent)
            support_y = support_dict[ykey]

            #compute confidence
            confidence = support_value / support_x

            #store if confidence is greater than minimun confidence
            if confidence >= (min_confidence / 100):
                x_out = ' ; '.join(sorted(list(combination_x)))
                y_out = ' ; '.join(sorted(combination_y))
                confidence_dict[x_out + " -> " + y_out] = confidence

#title for output
print("Confidence Dictionary")

#print out confidence
for key, value in confidence_dict.items():
  print(f"{key}: {value}")
```

```
Confidence Dictionary
Java: The Complete Reference -> Java For Dummies: 1.0
Java For Dummies -> Java: The Complete Reference: 0.7692307692307692
```

[463]:

```
#final results
print("Brute Force Results")

#define the outputs that we are looking for
first_elem_bf = ""
second_elem_bf = ""
confidence_bf = 0
support_bf = 0

#loop thru the confidence dictionary to come up with the results
for key, value in confidence_dict.items():
    #find X and Y for X->Y and store them as first and second element
    first_elem_bf = key.split(" -> ")[0]
    second_elem_bf = key.split(" -> ")[1]

    #store confidence values
    confidence_bf = value

    #create key to search in the support dictionary (A->B)
    all_possible_items = first_elem_bf.split(' ; ') + second_elem_bf.split(' ;␣
 ↪')

    support_key_bf = None
```

```python
    #check all the permutation of the items to find the correct key
    for p in permutations(all_possible_items):
        current_key_option = ' ; '.join(p)
        if current_key_option in support_dict:
            support_key_bf = current_key_option
            break # Stop once the key is found



    #find support value
    if support_key_bf:
        support_bf = support_dict[support_key_bf]

        #print out the results
        print("X:", first_elem_bf)
        print("Y:", second_elem_bf)
        print("Rule:", first_elem_bf, "->", second_elem_bf)
        print("Support: ", support_bf)
        print("Confidence: ", confidence_bf)
        print()

    else:
        print(f"\nWarning: Could not find support for the key '{support_key_bf}'. Skipping.")

#set final time
end_time_bf = time.time()

#compute total time for brute force algorithm
total_time_bf = end_time_bf - start_time_bf
print("Total time for Brute Force Algorithm: ", total_time_bf, "seconds")
```

```
Brute Force Results
X: Java: The Complete Reference
Y: Java For Dummies
Rule: Java: The Complete Reference -> Java For Dummies
Support:  0.5
Confidence:  1.0

X: Java For Dummies
Y: Java: The Complete Reference
Rule: Java For Dummies -> Java: The Complete Reference
Support:  0.5
Confidence:  0.7692307692307692

Total time for Brute Force Algorithm:  0.1014242172241211 seconds
```

5 Apriori

[465]:
```
from apyori import apriori

#set start time
start_time_apriori = time.time()

#extract the transaction column as a python list
#apyori does not support panda format here
transactions = []
for i in range(len(database['Transaction'])):
  transactions.append(database['Transaction'][i])

#run the apriori method
association_rules_apriori = apriori(transactions=transactions, #use list of␣
 ↪transactions as the input
                                    min_support=(min_support/100), #convert the␣
 ↪entered min support value to decimal
                                    min_confidence=(min_confidence/100))␣
 ↪#convert entered confidence value to decimal

#convert all the associations to a list
apriori_results = list(association_rules_apriori)

#if there is no association print out a message
if apriori_results == []:
  print("No association found")
else:
  print("Apriori Results")


#define the outputs that we are looking for
first_elem_apriori = ""
second_elem_apriori = ""
confidence_apriori = 0
support_apriori = 0




#list comes out in this format
#[RelationRecord(items=frozenset({'XX'}), support=XX,␣
 ↪ordered_statistics=[OrderedStatistic(items_base=frozenset(),␣
 ↪items_add=frozenset({'XX'}), confidence=0.55, lift=1.0)]),
#it is a list with a lof of relations recorded
#we need to access inside each relation record so lets create a loop
for relation_record in apriori_results:
  #record the support
```

```python
  support_apriori = relation_record.support

  #create another loop to access inside the ordered statistics data
  for item_recorded in relation_record.ordered_statistics:

    #if there is no value for the first or secound element skip to the end
    if not item_recorded.items_base or not item_recorded.items_add:
      continue

    #find x and y for X->Y and store them as first and second element
    #create another loop to access the item base (X)
    for item in item_recorded.items_base:
      first_elem_apriori = item

    #create another loop to access the item add (Y)
    for item in item_recorded.items_add:
      second_elem_apriori = item

    #store confidence values
    confidence_apriori = item_recorded.confidence

    #print out the results

    print("X:", first_elem_apriori)
    print("Y:", second_elem_apriori)
    print("Rule:", first_elem_apriori, "->", second_elem_apriori)
    print("Support: ", support_apriori)
    print("Confidence: ", confidence_apriori)
    print()

#set end time
end_time_apriori = time.time()

#compute total time for apriori algorithm
total_time_apriori = end_time_apriori - start_time_apriori
print("Total time for Apriori Algorithm: ", total_time_apriori, "seconds")
```

```
Apriori Results
X: Java For Dummies
Y: Java: The Complete Reference
Rule: Java For Dummies -> Java: The Complete Reference
Support:  0.5
Confidence:  0.7692307692307692

X: Java: The Complete Reference
Y: Java For Dummies
Rule: Java: The Complete Reference -> Java For Dummies
Support:  0.5
```

```
Confidence:  1.0

Total time for Apriori Algorithm:  0.0010137557983398438 seconds
```

6 FP-Growth

[467]:

```
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules

#set start time
start_time_fpg = time.time()

#extract the transaction column as a python list
#mlxtend requires one hot encoded format
transactions = []
for i in range(len(database['Transaction'])):
  transactions.append(database['Transaction'][i])

#convert the list into one-hot encoded columns (T/F for each item)
transac_encoder = TransactionEncoder()

#take all the unique values and transform it to a matrix
transac_array = transac_encoder.fit(transactions).transform(transactions)

#convert the matrix to a Panda dataframe
transac_df = pd.DataFrame(transac_array, columns=transac_encoder.columns_)

#print out result
print(transac_df)
```

```
    A Beginner’s Guide  Android Programming: The Big Nerd Ranch  \
0                 True                                     True
1                 True                                    False
2                 True                                     True
3                False                                     True
4                False                                     True
5                 True                                     True
6                 True                                    False
7                False                                     True
8                False                                     True
9                False                                    False
10                True                                     True
11                True                                    False
12                True                                    False
13               False                                     True
14               False                                     True
```

```
15                True                                     True
16                True                                     True
17               False                                    False
18               False                                     True
19                True                                    False

    Beginning Programming with Java  C++ Programming in Easy Steps  \
0                             False                          False
1                             False                          False
2                             False                          False
3                              True                          False
4                              True                          False
5                             False                          False
6                              True                          False
7                             False                          False
8                              True                          False
9                              True                           True
10                            False                          False
11                            False                          False
12                            False                          False
13                            False                          False
14                            False                          False
15                            False                          False
16                            False                          False
17                             True                          False
18                            False                          False
19                            False                          False

    HTML and CSS: Design and Build Websites  Head First Java 2nd Edition  \
0                                     False                        False
1                                     False                        False
2                                     False                         True
3                                     False                         True
4                                     False                        False
5                                     False                         True
6                                     False                         True
7                                     False                        False
8                                     False                         True
9                                     False                        False
10                                    False                        False
11                                     True                        False
12                                     True                        False
13                                    False                         True
14                                    False                        False
15                                    False                        False
16                                    False                        False
17                                    False                         True
18                                    False                         True
```

```
19                                    False                        False

    Java 8 Pocket Guide  Java For Dummies  Java: The Complete Reference
0                 False              True                          True
1                 False              True                          True
2                 False              True                          True
3                 False             False                         False
4                  True             False                         False
5                 False             False                         False
6                 False             False                         False
7                 False              True                          True
8                 False              True                         False
9                  True             False                         False
10                False              True                          True
11                False              True                          True
12                 True              True                          True
13                False              True                         False
14                False              True                         False
15                False              True                          True
16                False              True                          True
17                 True             False                         False
18                False             False                         False
19                False              True                          True
```

[468]:

```
#run fp growth algorithm
frequent_itemsets_fpg = fpgrowth(transac_df, min_support=(min_support/100),␣
 ↪use_colnames=True)

#print out result
print(frequent_itemsets_fpg)
```

```
   support                                          itemsets
0     0.65                                (Java For Dummies)
1     0.65         (Android Programming: The Big Nerd Ranch)
2     0.55                              (A Beginner’s Guide)
3     0.50                    (Java: The Complete Reference)
4     0.50  (Java For Dummies, Java: The Complete Reference)
```

[469]:

```
#generate association rules from the frequent itemsets
association_rules_fpg = association_rules(frequent_itemsets_fpg,␣
 ↪min_threshold=(min_confidence/100))

#make sure that association rules ha no empty X or Y for X->Y
association_rules_fpg = association_rules_fpg.dropna(subset=['antecedents',␣
 ↪'consequents'])

#print out result
print(association_rules_fpg)
```

```
                      antecedents                     consequents  \
0              (Java For Dummies)  (Java: The Complete Reference)
1  (Java: The Complete Reference)              (Java For Dummies)

   antecedent support  consequent support  support  confidence      lift  \
0                0.65                0.50      0.5    0.769231  1.538462
1                0.50                0.65      0.5    1.000000  1.538462

   representativity  leverage  conviction  zhangs_metric   jaccard  certainty  \
0               1.0     0.175    2.166667            1.0  0.769231   0.538462
1               1.0     0.175         inf            0.7  0.769231   1.000000

   kulczynski
0    0.884615
1    0.884615
```

[470]:

```python
#print out results
print("FP-Growth Results")

#define the outputs that we are looking for
first_elem_fpg = ""
second_elem_fpg = ""
confidence_fpg = 0
support_fpg = 0

#start loop to go thru each row collecting X, Y, confidence and support and print
for index, row in association_rules_fpg.iterrows():
  first_elem_fpg = list(row['antecedents'])
  second_elem_fpg = list(row['consequents'])
  confidence_fpg = row['confidence']
  support_fpg = row['support']
  print("X:", first_elem_fpg)
  print("Y:", second_elem_fpg)
  print("Rule:", first_elem_fpg, "->", second_elem_fpg)
  print("Support: ", support_fpg)
  print("Confidence: ", confidence_fpg)
  print()

#set end time
end_time_fpg = time.time()

#compute total time for fpg algorithm
total_time_fpg = end_time_fpg - start_time_fpg
print("Total time for FP-Growth Algorithm: ", total_time_fpg, "seconds")
```

```
FP-Growth Results
X: ['Java For Dummies']
```

```
Y: ['Java: The Complete Reference']
Rule: ['Java For Dummies'] -> ['Java: The Complete Reference']
Support:  0.5
Confidence:  0.7692307692307692

X: ['Java: The Complete Reference']
Y: ['Java For Dummies']
Rule: ['Java: The Complete Reference'] -> ['Java For Dummies']
Support:  0.5
Confidence:  1.0

Total time for FP-Growth Algorithm:  0.0731959342956543 seconds
```

7 Results and Conclusion

The project successfully implemented Brute-Force, Apriori and FP-Growth algorithms to find frequent itemsets and generate association rules from retail datasets.

The output was expected to be the same for all the three methods and our code also provided same results. For example, if we use minimum support of 50% and a minimum confidence of 50% for amazon dataset, we will have the following association rules:

[‘Java For Dummies’] -> [‘Java: The Complete Reference’]

[‘Java: The Complete Reference’] -> [‘Java For Dummies’]

The main difference between the output of the three algorithms is when it comes to the execution time as you can see below.

[473]:

```python
print("Total time for Brute Force Algorithm: ", total_time_bf, "seconds")
print("Total time for Apriori Algorithm: ", total_time_apriori, "seconds")
print("Total time for FP-Growth Algorithm: ", total_time_fpg, "seconds")
```

```
Total time for Brute Force Algorithm:  0.1014242172241211 seconds
Total time for Apriori Algorithm:  0.0010137557983398438 seconds
Total time for FP-Growth Algorithm:  0.0731959342956543 seconds
```

The Brute-Force method takes more time because it checks every possible combination of items making it not as much efficient compared to the other two methods.

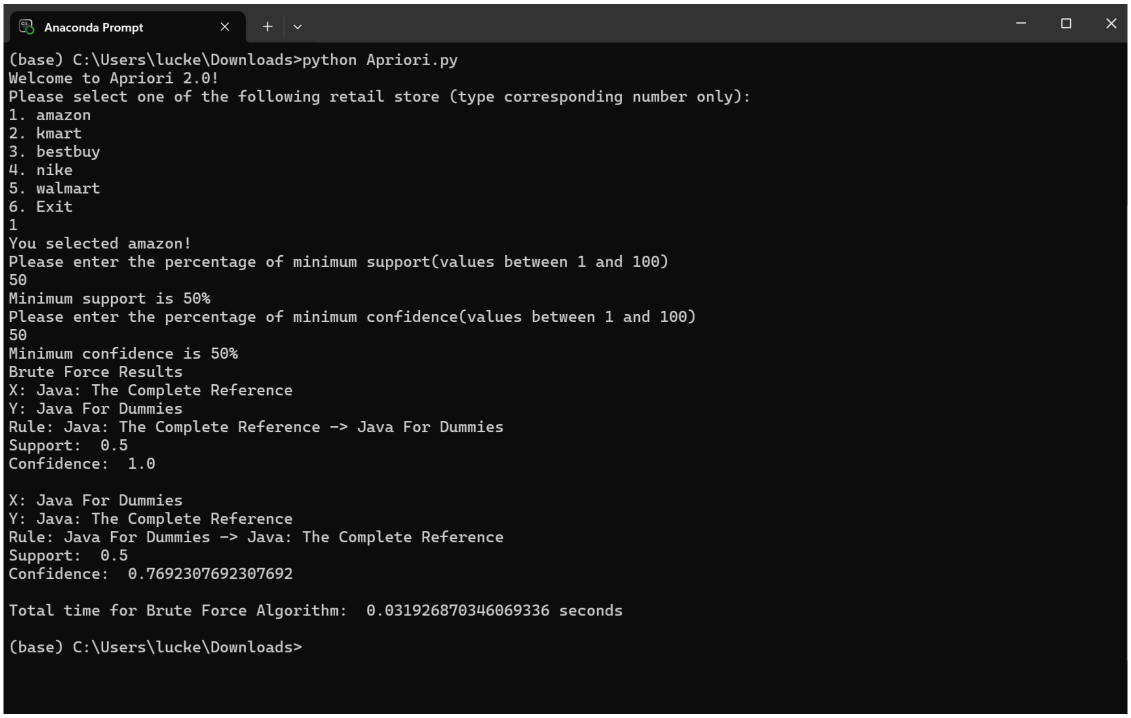
Apriori method use of pruning to avoid going thru all the possible combinations. However, FP-Growth method is typically the fastest because it has a compact tree like data structure. In our project, we noticed that the FP-Growth method was not faster than the Apriori. The reason for Apriori being faster in this project is the number of input data. The FP-Growth method has an initial high time for setup but it can find all the combinations faster once you passed thru the intial stage. In short, the FP-Growth method would be faster for a more complex and large dataset.

Finally, this project was a create opportunity to have some hands on experience with association algorithm. As the number of data increases the algorithm has to have a more robust architeture to deal with a big amount of data.

8 Running each algorithm as standalone scripts

Make sure you have the Data folder saved in the folder where you have the script files

8.1 Brute Force



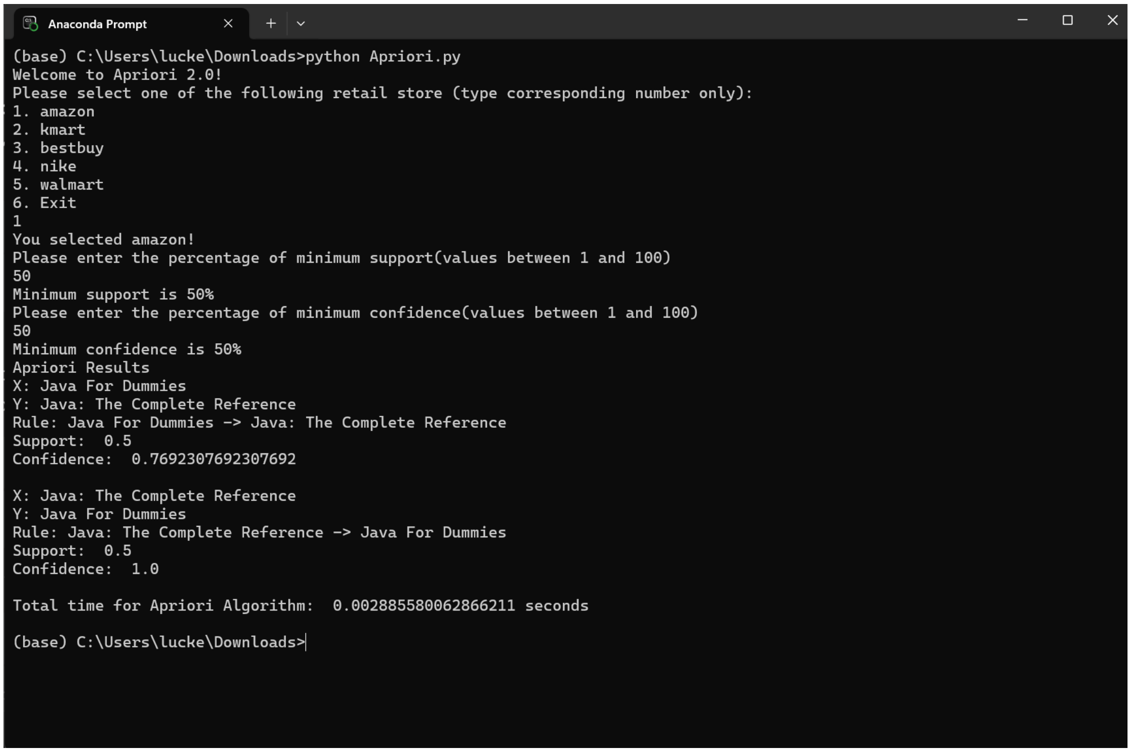
```
(base) C:\Users\lucke\Downloads>python Apriori.py
Welcome to Apriori 2.0!
Please select one of the following retail store (type corresponding number only):
1. amazon
2. kmart
3. bestbuy
4. nike
5. walmart
6. Exit
1
You selected amazon!
Please enter the percentage of minimum support(values between 1 and 100)
50
Minimum support is 50%
Please enter the percentage of minimum confidence(values between 1 and 100)
50
Minimum confidence is 50%
Brute Force Results
X: Java: The Complete Reference
Y: Java For Dummies
Rule: Java: The Complete Reference -> Java For Dummies
Support:  0.5
Confidence:  1.0

X: Java For Dummies
Y: Java: The Complete Reference
Rule: Java For Dummies -> Java: The Complete Reference
Support:  0.5
Confidence:  0.7692307692307692

Total time for Brute Force Algorithm:  0.031926870346069336 seconds

(base) C:\Users\lucke\Downloads>
```

8.2 Apriori



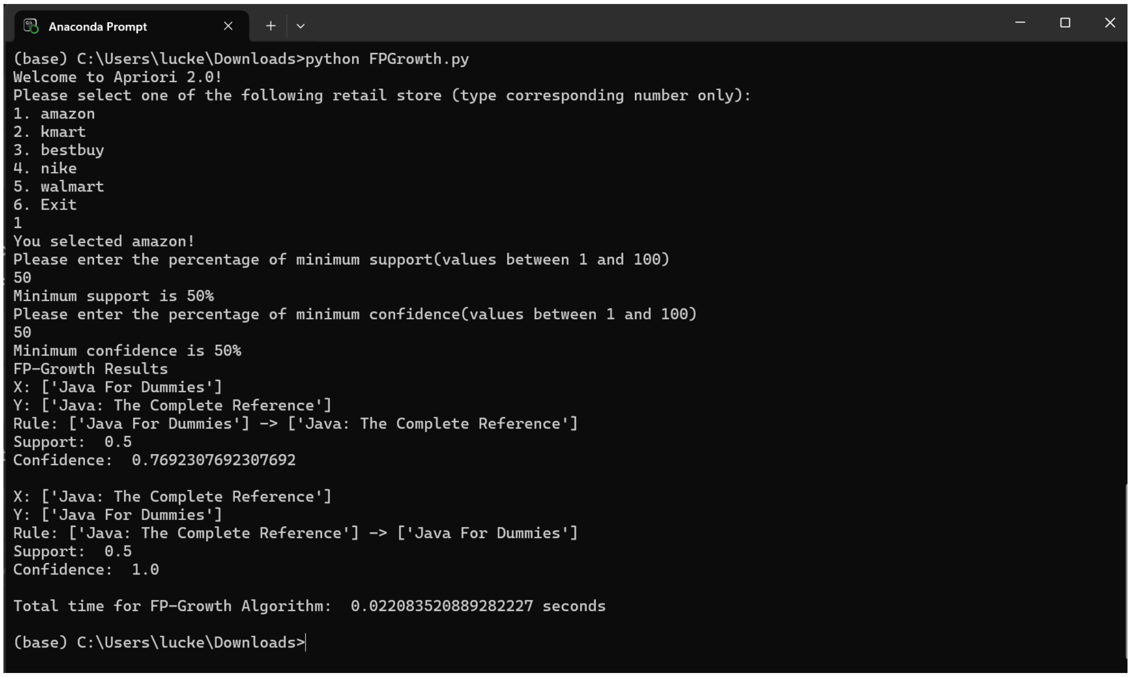
```
(base) C:\Users\lucke\Downloads>python Apriori.py
Welcome to Apriori 2.0!
Please select one of the following retail store (type corresponding number only):
1. amazon
2. kmart
3. bestbuy
4. nike
5. walmart
6. Exit
1
You selected amazon!
Please enter the percentage of minimum support(values between 1 and 100)
50
Minimum support is 50%
Please enter the percentage of minimum confidence(values between 1 and 100)
50
Minimum confidence is 50%
Apriori Results
X: Java For Dummies
Y: Java: The Complete Reference
Rule: Java For Dummies -> Java: The Complete Reference
Support:  0.5
Confidence:  0.7692307692307692

X: Java: The Complete Reference
Y: Java For Dummies
Rule: Java: The Complete Reference -> Java For Dummies
Support:  0.5
Confidence:  1.0

Total time for Apriori Algorithm:  0.002885580062866211 seconds

(base) C:\Users\lucke\Downloads>
```

8.3 FP-Growth



```
(base) C:\Users\lucke\Downloads>python FPGrowth.py
Welcome to Apriori 2.0!
Please select one of the following retail store (type corresponding number only):
1. amazon
2. kmart
3. bestbuy
4. nike
5. walmart
6. Exit
1
You selected amazon!
Please enter the percentage of minimum support(values between 1 and 100)
50
Minimum support is 50%
Please enter the percentage of minimum confidence(values between 1 and 100)
50
Minimum confidence is 50%
FP-Growth Results
X: ['Java For Dummies']
Y: ['Java: The Complete Reference']
Rule: ['Java For Dummies'] -> ['Java: The Complete Reference']
Support:  0.5
Confidence:  0.7692307692307692

X: ['Java: The Complete Reference']
Y: ['Java For Dummies']
Rule: ['Java: The Complete Reference'] -> ['Java For Dummies']
Support:  0.5
Confidence:  1.0

Total time for FP-Growth Algorithm:  0.022083520889282227 seconds

(base) C:\Users\lucke\Downloads>
```