NAME: Sameer Khan Mohammed  
NJIT UCID: sm3677  
Email Address: [sm3677@njit.edu](mailto:sm3677@njit.edu)  
Date: 03-02-2025  
Professor: Yasser Abduallah   
CS 634-854 Data Mining

**Midterm Project Report**

**Abstract**

Market Basket Analysis is a popular technique in data mining that helps uncover relationships between different items in large transaction datasets. This report explores the implementation of three algorithms: Brute Force, Apriori, and FP-Growth, which are used for association rule mining and finding frequent itemsets. The study evaluates these algorithms based on their accuracy, efficiency, and scalability. The results highlight the trade-offs between computational complexity and the ability to discover meaningful patterns in data.

**Introduction**

Association Rule Mining (ARM) is an important concept in data science, especially in retail and e-commerce, where businesses use it to analyze customer purchasing behavior. Understanding these patterns helps improve marketing strategies and inventory management. In this report, three different approaches to ARM are examined: Brute Force, Apriori, and FP-Growth. The Brute Force method systematically generates all possible itemsets, making it simple but highly inefficient for large datasets. The Apriori algorithm improves this process by iteratively generating itemsets and pruning infrequent ones, while FP-Growth uses a tree structure to enhance efficiency. This report describes the implementation of these algorithms and compares their performance.

**System Preparation:**  
Installation of Software’s and Packages for the project  
  
Software’s needed:

* Python(3.12.0): Download and install from [www.python.org](http://www.python.org)
* Pip (24.2): Included in python
* Jupyter Notebook: Ensure you have python installed first. Then in you terminal run “pip install notebook” to install Jupyter notebook

Packages to install:

* pandas
* mlxtend.frequent\_patterns
* timeit

**Core Concepts and Principles**

The process of association rule mining involves several key steps:

1. **Frequent Itemset Generation:** Identifying sets of items that frequently appear together in transactions.
2. **Support Calculation:** Measuring how often an itemset appears relative to the total number of transactions**.**
3. **Confidence Calculation:** Evaluating the probability of one item appearing given that another item is present.
4. **Rule Generation:** Extracting useful relationships between items from frequent itemsets.
5. **Efficiency Considerations:** Optimizing computational performance to handle large datasets effectively.

**Algorithms Used**

1. **Brute Force Method**
   * The Brute Force method works by generating all possible itemsets and counting their occurrences.
   * It guarantees finding all possible rules but is highly inefficient when dealing with large datasets due to its exponential time complexity.
   * This method is most practical for small datasets where an exhaustive approach is feasible.
2. **Apriori Algorithm**
   * The Apriori algorithm is based on the principle that if an itemset is frequent, then all its subsets must also be frequent.
   * It follows a bottom-up approach, gradually expanding itemsets and eliminating infrequent ones to reduce the search space.
   * Despite its optimizations, Apriori still requires multiple database scans, which can be computationally expensive for very large datasets.
3. **FP-Growth Algorithm**
   * The FP-Growth algorithm introduces a more efficient approach by using an FP-Tree to represent transactions in a compact form.
   * Instead of generating candidate itemsets explicitly, it recursively mines the FP-Tree to find frequent patterns.
   * This algorithm is significantly faster than Apriori and Brute Force, especially for large datasets, as it avoids unnecessary computations.

**Results and Evaluation**

To compare these algorithms, a dataset containing transaction records from different shopping companies was analyzed. The evaluation criteria included:

1. **Efficiency:** Measuring the time taken by each algorithm to generate frequent itemsets and association rules.
2. **Scalability:** Testing how well each algorithm performs as the dataset size increases.
3. **Accuracy:** Comparing the relevance and validity of the generated rules.

The results showed that while the Brute Force method provides a complete and exhaustive search, it is impractical for large datasets due to its high computational cost. The Apriori algorithm offers better efficiency by reducing the number of candidate itemsets, though it still requires multiple database scans. In contrast, the FP-Growth algorithm outperforms both, as it efficiently identifies frequent itemsets without needing to generate candidate sets explicitly. This makes FP-Growth the best choice for large-scale market basket analysis.

**Conclusion**

This report explores different methods for association rule mining, each with its own advantages and limitations. The Brute Force approach, while conceptually straightforward, is only useful for small datasets. The Apriori algorithm provides better efficiency but still struggles with large datasets due to repeated database scans. The FP-Growth algorithm, with its tree-based structure, offers the most efficient solution for mining frequent itemsets. Future research could explore hybrid models that combine the strengths of these algorithms to further optimize performance in practical applications.

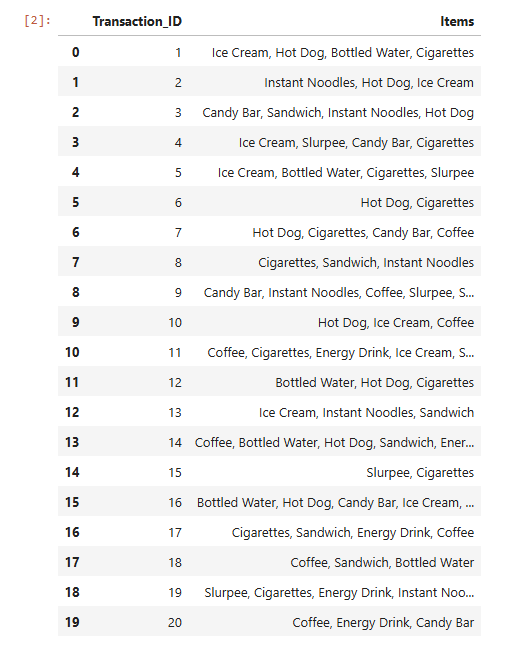
**Datasets Used:**  I have created 6 random datasets of supermarkets using a python randomizer code. Saved the output as CSV file and used them in the project.

**Program Running:**

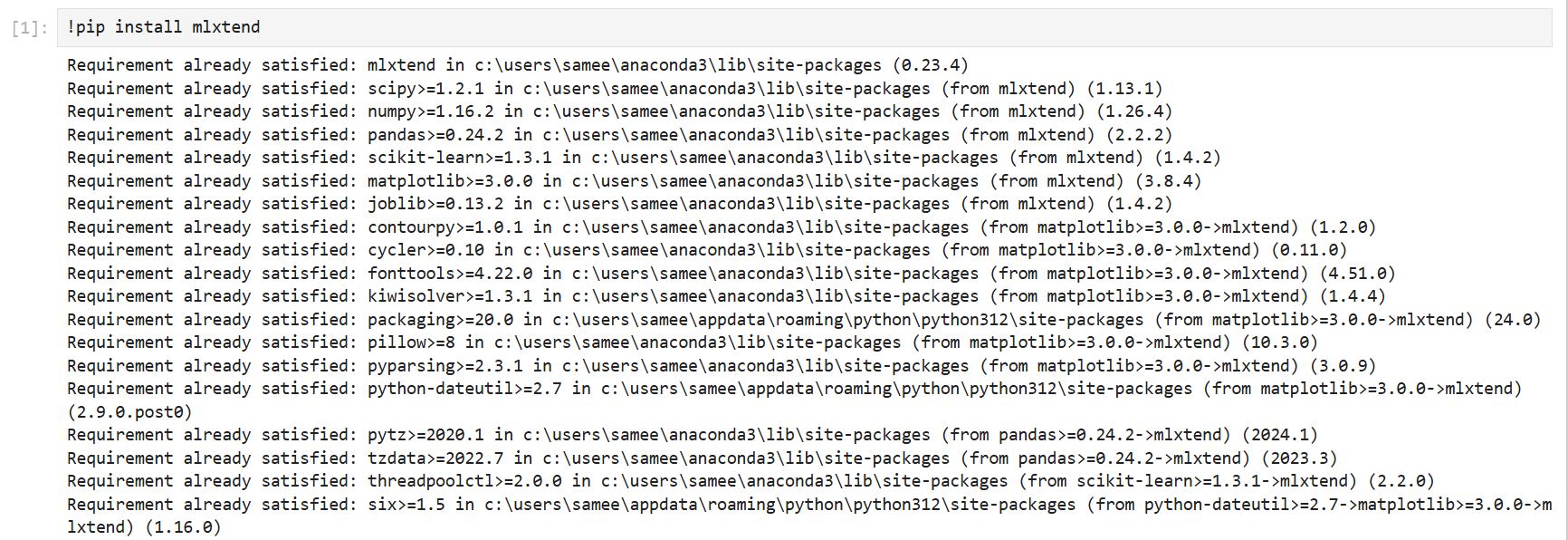
1. Install Jupyter Notebook on your local machine.
2. Launch the command prompt and run the command jupyter notebook.
3. Create a new Jupyter Notebook in your desired directory.
4. Inside the notebook, install the required package by executing !pip install mlxtend.
5. Restart the notebook after the installation is complete.
6. Import the necessary libraries: pandas, apriori, fpgrowth, association\_rules, and timeit.
7. Run the program in a code cell.
8. A prompt will appear, asking you to select a company from the list. Choose the corresponding number for your desired company.
9. Enter the minimum support and confidence values when prompted.
10. The program will execute and display the results of all three algorithms, along with the time taken for execution.
11. To exit, press "7".

Screenshots

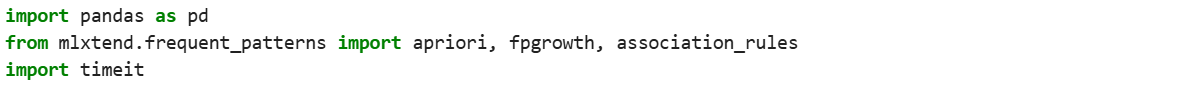
Example CSV file Screenshot

7-11.csv  


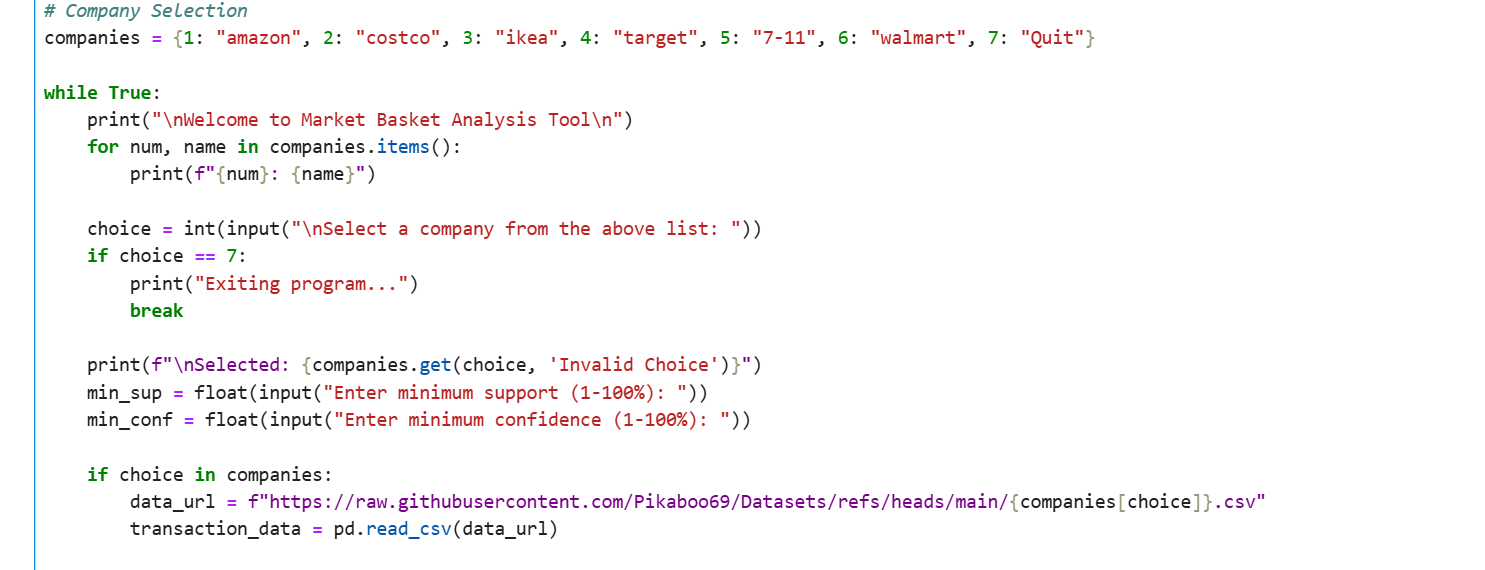
1. Firstly, we install the important package of the project which is mlxtend, by running the below command. It contains Apriori and FP- Tree Growth algorithms.



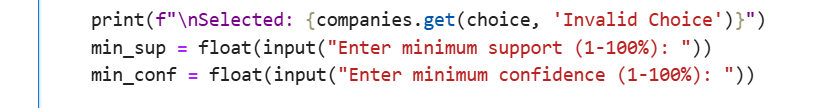
2. Import pandas for data handling, apriori, fpgrowth, and association\_rules from mlxtend.frequent\_patterns, and timeit for execution time analysis.



3. Allow user to select from six company datasets hosted on GitHub. Read the dataset from a CSV file corresponding to the selected company

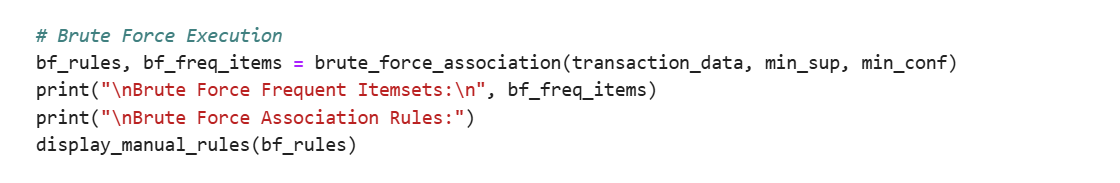


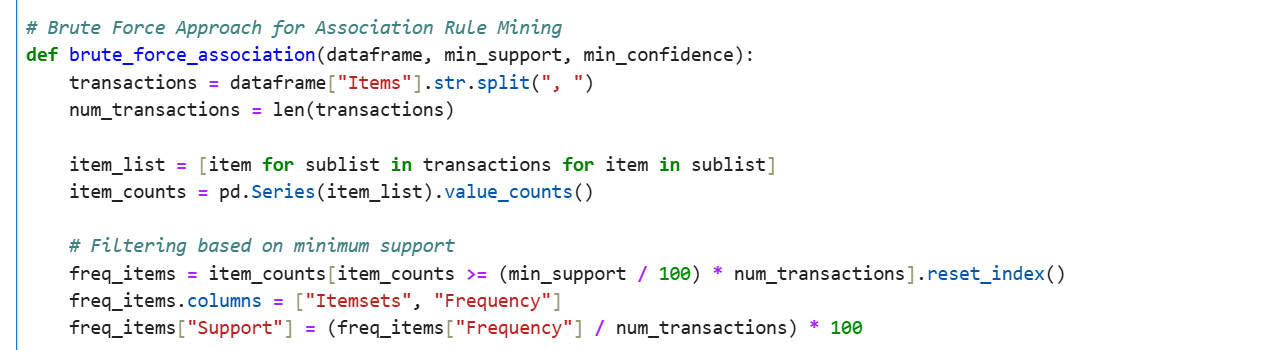
4. Collect user-defined thresholds for support and confidence values.

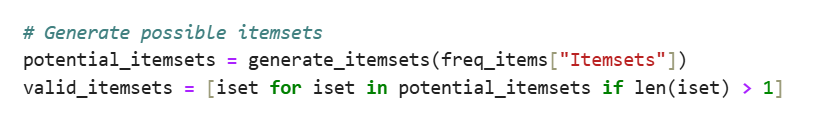
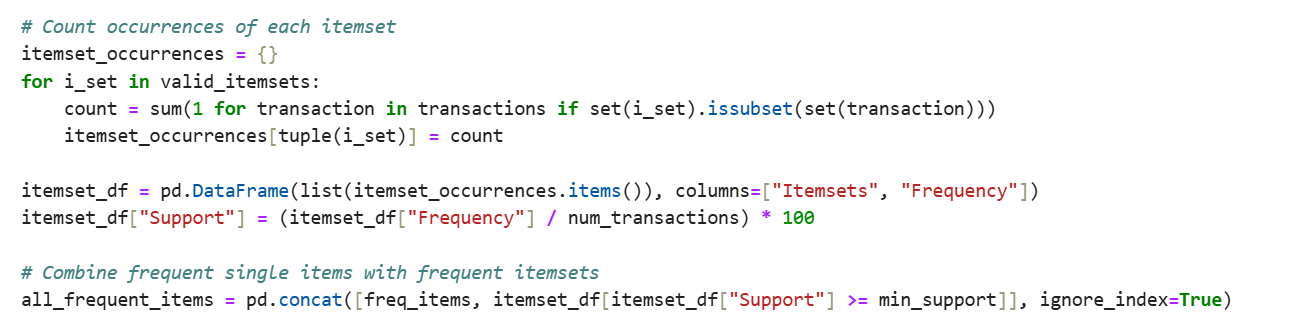
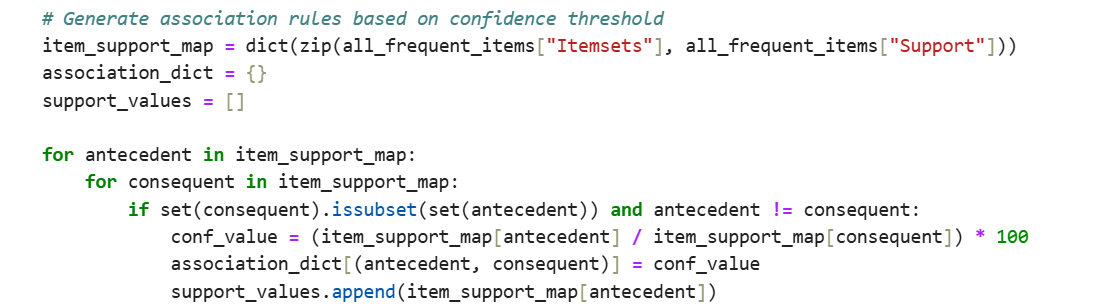
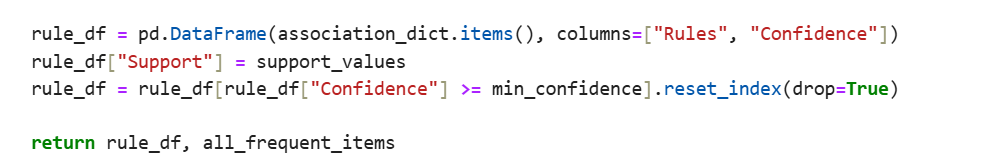


5. **Performing the Brute Force Algorithm**

* Generate all frequent itemsets by manually iterating through transactions.
* Calculate frequency and filter itemsets based on the minimum support threshold.
* Generate association rules by computing confidence and filtering based on the minimum confidence.

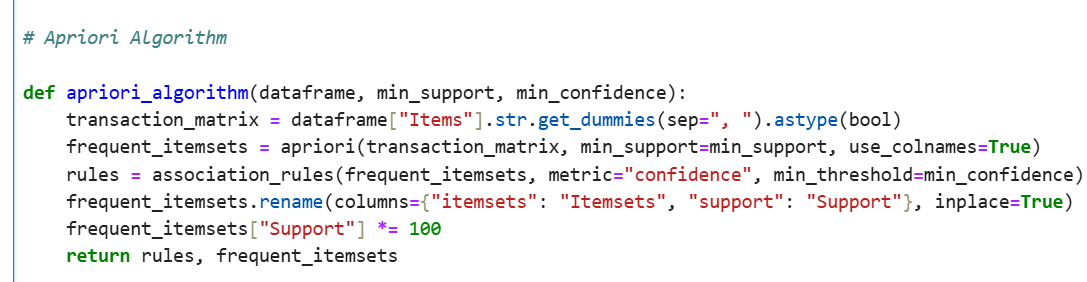
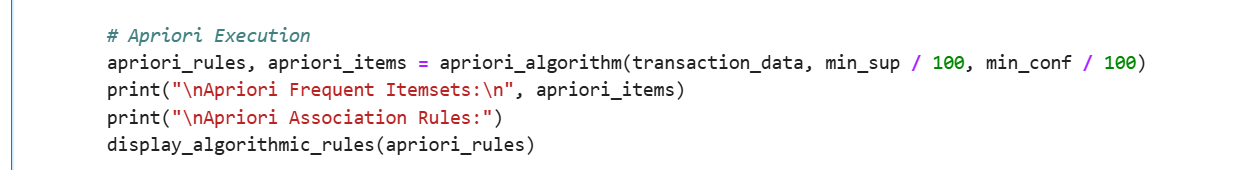




* 1. **Generating Itemsets**: Generate all possible itemsets from the remaining items after filtering based on the minimum support threshold.  
     
  2. **Itemset count & Filtering:** Count the occurrences of all frequent itemsets and filter them based on the minimum support, removing any non-frequent itemsets.   
     
  3. **Filtering Based on Confidence**: Filter the Data Frame using the provided confidence threshold.  
     
  4. **Displaying Final Association Rules:** Print the final association rules generated by the brute force algorithm in a clear and proper format.  
     

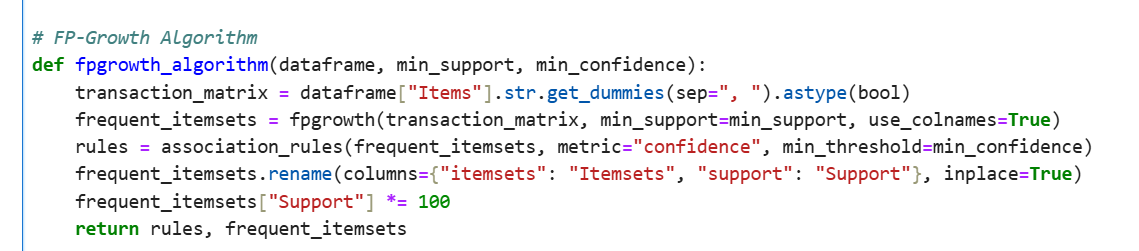
6. **Performing the Apriori Algorithm**

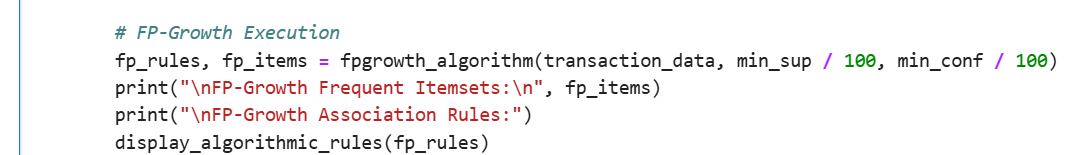
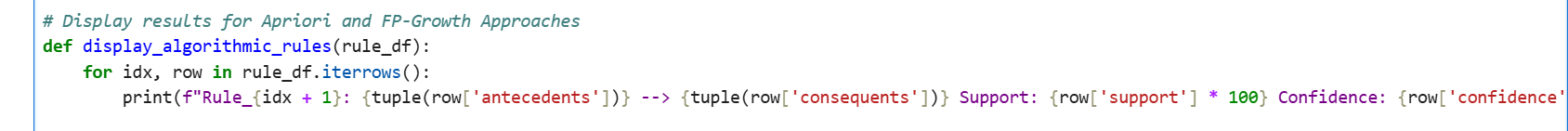
* Convert the dataset into a transaction matrix format.
* Use the Apriori algorithm to find frequent itemsets.
* Generate association rules based on confidence.

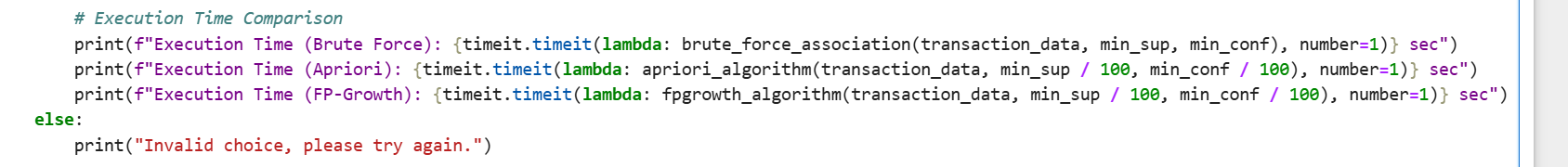
  


7. **Performing the FP-Growth Algorithm**

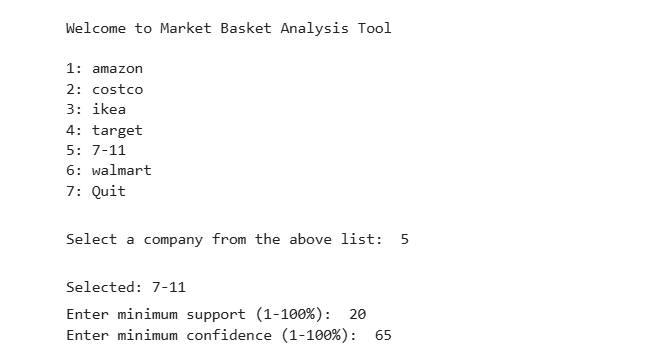
* Convert the dataset into a transaction matrix format.
* Apply the FP-Growth algorithm for frequent itemset discovery.
* Generate association rules from the identified itemsets.

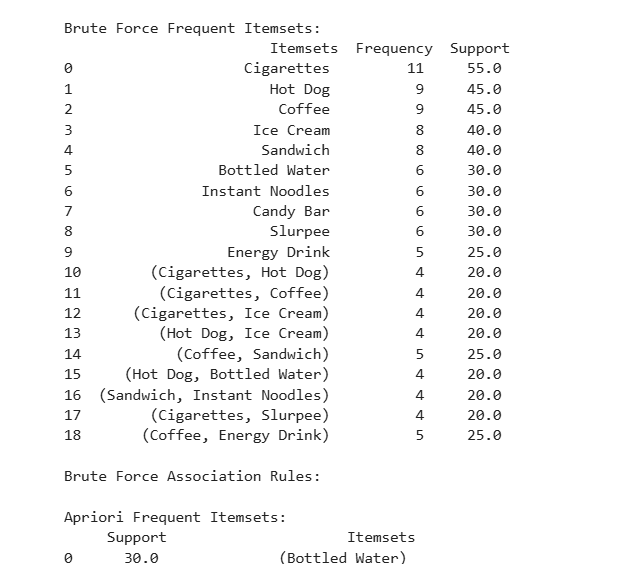


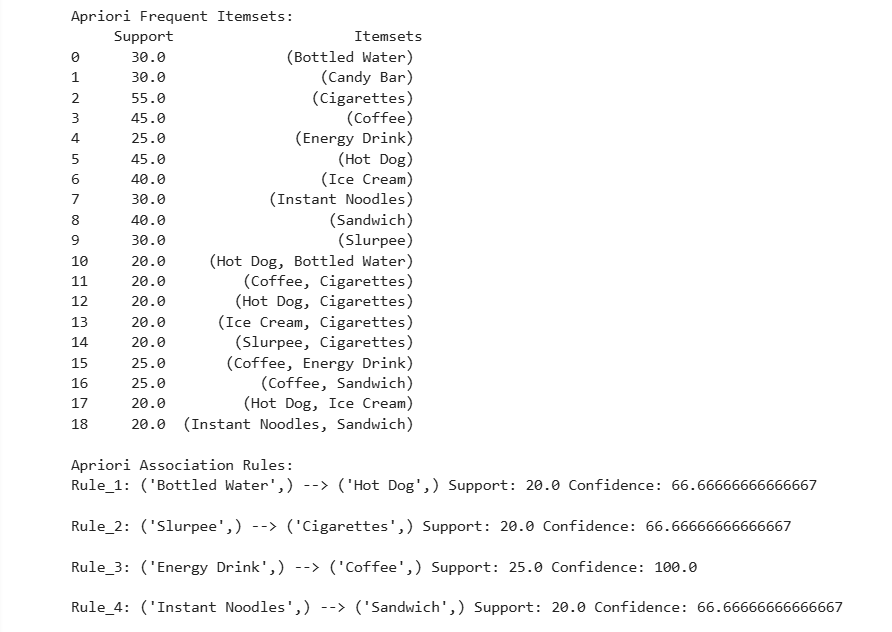
  


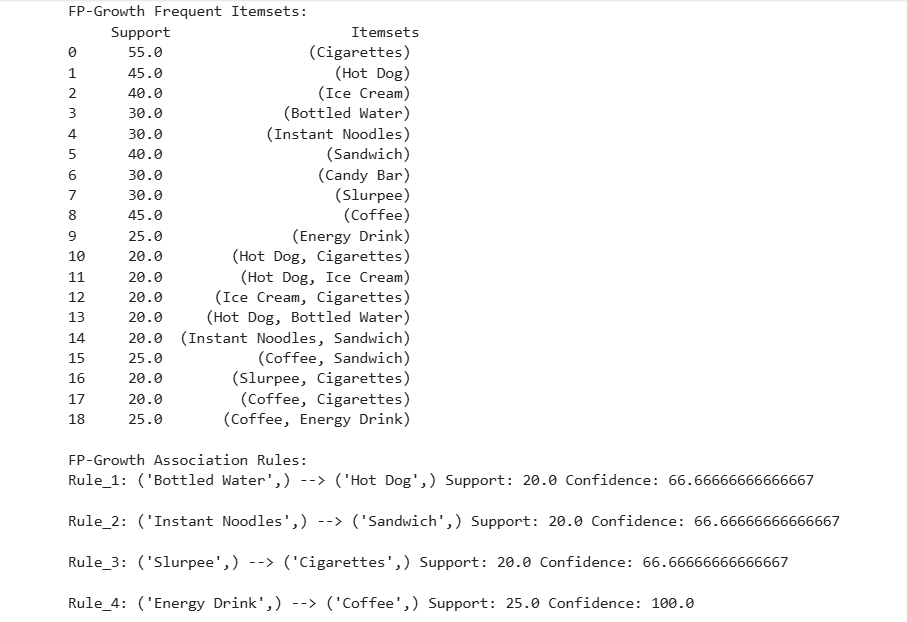
8. **Comparing Execution Times:** Finally, compare the time taken to execute each algorithm to determine which one is faster and more efficient.  


**OUTPUT:**

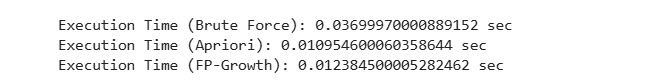








**EXECUTION TIMES OF THREE ALGORITHMS**



All three algorithms give different results.

Apriori is the fastest one with 0.109 sec.

The source code (.py file) and data sets (.csv files) will be attached to the zip file.

Link to Git Repository

<https://github.com/Pikaboo69/Mohammed_Sameer_Khan_MidTerm_Project>