**Rayon Myrie**

[**Rm25@NJIT.EDU**](mailto:Rm25@NJIT.EDU)

**Student ID: 21269817**

**CS 634 – Data Mining**

**Mid-term Project**

**Association Rule using Aproiri Algorithm**

**Introduction:**

The purpose of the project is to create a program that will discover interesting relations among variables in a dataset, using the association rule and Apriori Algorithm.

**Programming Language:**

Python, using Pandas

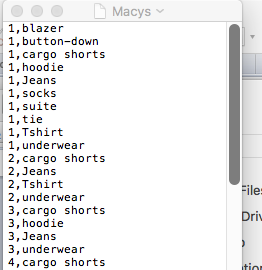
**Program:**

Anaconda’s Jupyter Notebook

**Project Details:**

For this project I created 5 databases with 20 transactions that contains items that can be seen in Shop-Rite, Guitar Center, Macy’s, Micro-Center and The Home Depot. Customers normally purchase these items and the order number and item number are normally saved in the Orders tables. For the purpose of this project I am going to extract from the database the order number and the item type. For example, if someone orders ‘Horizon Milk’, I will have the category ‘Milk’ for the item, instead of the brand. In addition, the column headers were excluded from the extract files.

I am going to save the order numbers and items as a list in a comma delimited file and use python (jupytor Notebook) to find the association rule using Apriori Algorithm.



**Method:**

**Association Rule:**

Goal is to uncover hidden patterns between transactions using data pulled from retail stores.

**Application:**

Association rules are widely used in by stores when they are stocking items for sale. Company’s us the market basket analysis to collect data by scanning bar codes to identify items that are being bought together. From the pattern collected the store managers then rearrange to the stores to place the more selective items together to increase sales and market items better to their customers by providing group discounts.

Application rules can also be used in other industries, such as Medical, finance and HR outsourcing.

**Apriori Algorithm for Program:**

1. Provide file name, minimum support and minimum confidence
2. Scan for the count of each Candidate
3. Compare candidate support count with minimum support count and drop candidates that do not have minimum support
4. Generate candidates of frequent sets
5. Compare candidates for frequent set, get a count of how many times the set is repeated, then calculate the support of the frequent set
6. Remove any frequent set that does not meet the minimum requirement
7. Find the confidence for each item in the set.
8. Remove any item that has confidence less than the minimum confidence

**Data Preparation:**

Input CSV files used in the program in comma delimited format

|  |  |
| --- | --- |
| Company | Data file |
| Macy’s |  |
| Guitar Center |  |
| Home Depot |  |
| Micro Center |  |
| Shop -Rite |  |

**Major Challenges:**

* **Learning Curve**: This was the first time design and program using python. It required many hours our research and reviewing online material to find the approach and functions that would work.
* **Finding the frequent item-sets in python**: I could not find a straightforward function that would allow me to query DataFrames using pandas. There was no like % option in pandas, that I am used to SQL. Which made it difficult to compare records. I found a work around to find the frequent sets, but I was unable to find a way to count the occurrences. I ended up using this at the end of my code to show that the program does not work.
* **Learning Resources**: There are many learning resources that show you how to code in python, but they are vague. They do not get into the details on how to do more complex data analysis; they just show you the basics.

**Testing and Evaluation:**

After testing the program, I found it did not produce the expected results. The program only produced frequent pairs, N <2. The items with n > 2 were excluded from the results. The reason for this was the module only used a series 2 X 2 , any other series would cause the program to fail.

**Test Scenario 1:  
Results from the Microcenter Dataset:**

For the Microcenter.csv test case, I used a minimum support of 60% and minimum confidence of 85%.

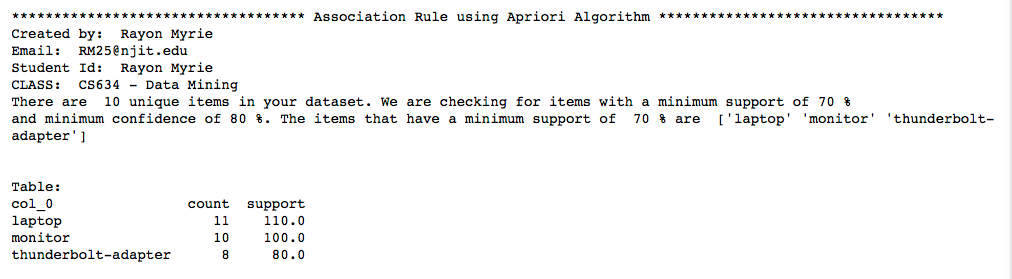
Screen Shot:

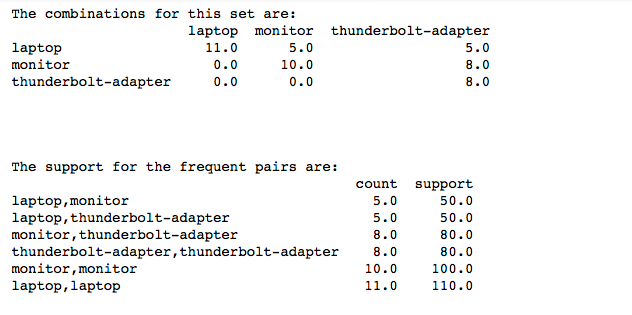
**Load Input file:  
**

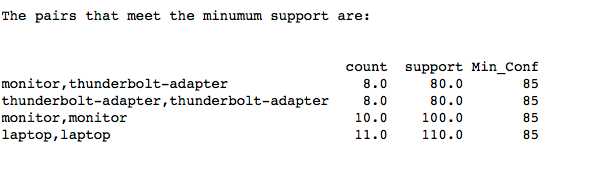
**Enter minimum Support and minimum Confidence:**

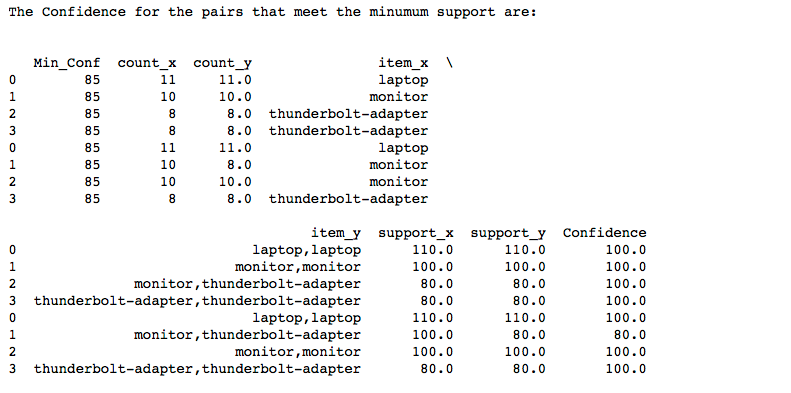
****

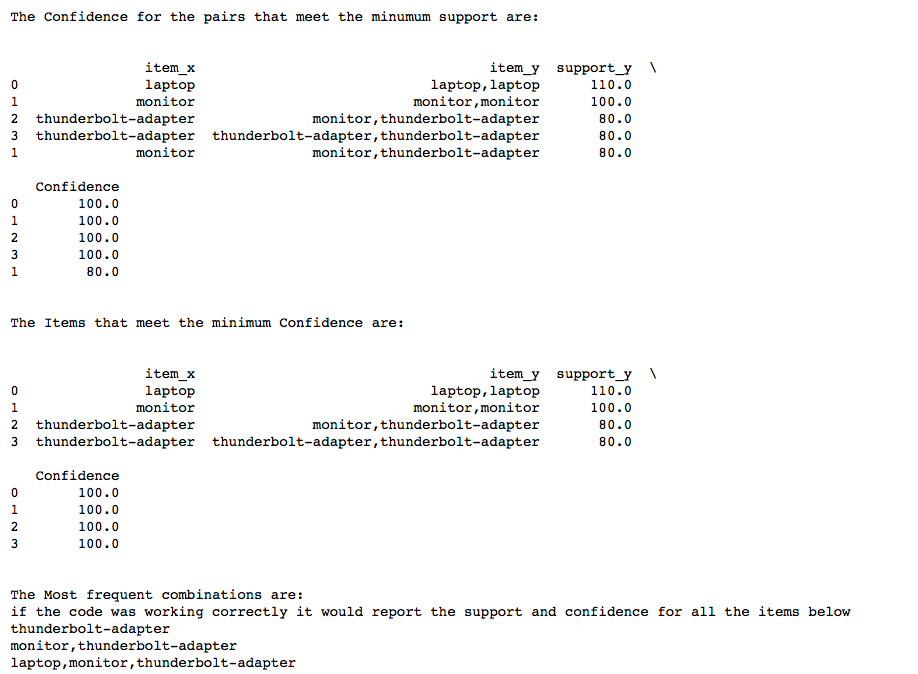
**Results:**

****

****

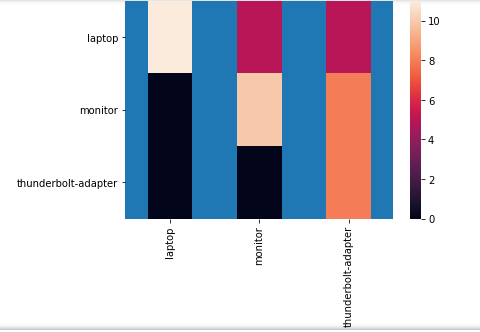
****

****

****

**Expected Results**

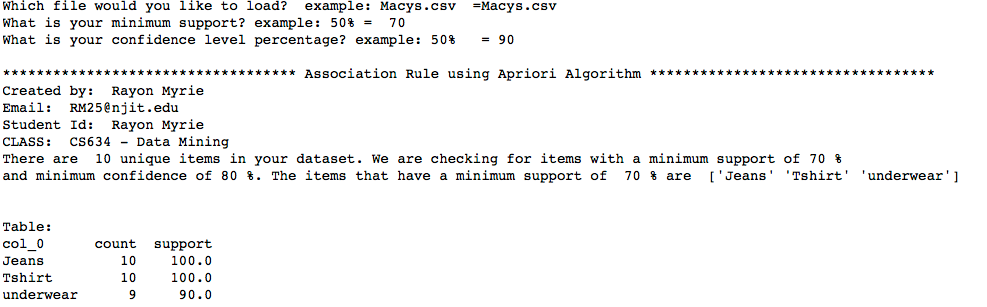
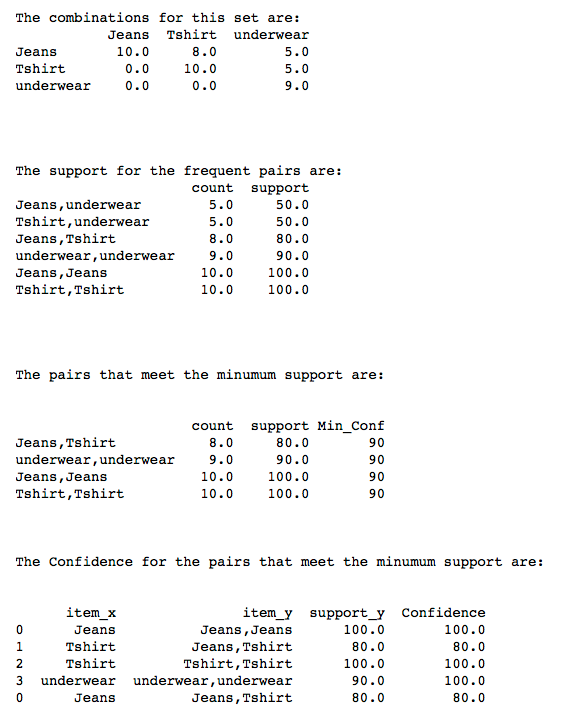
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item** | **Most frequent Sets** | **Count of Item** | **Count of Set** | **Support\_item** | **Support of Set** | **Confidence** |
| thunderbolt-adapter | thunderbolt-adapter | 8 | 8 | 80 | 100 | 125 |
| laptop,monitor | laptop,monitor,thunderbolt-adapter | 5 | 5 | 50 | 100 | 200 |
| thunderbolt-adapter | monitor,thunderbolt-adapter | 8 | 8 | 80 | 100 | 125 |
| laptop | laptop | 11 | 11 | 110 | 100 | 90.90909091 |

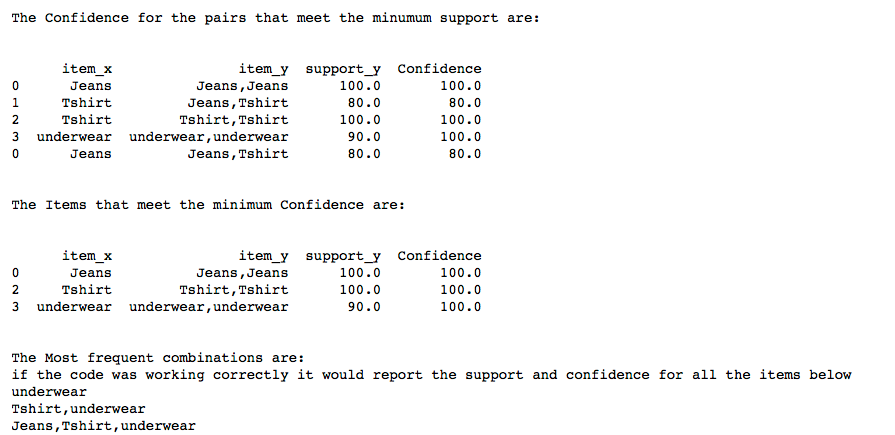
****

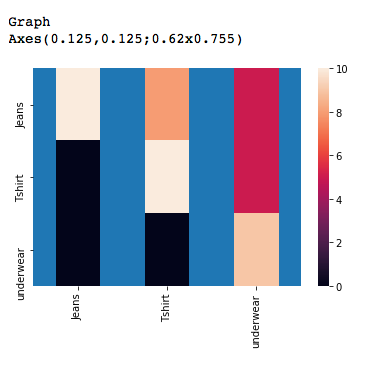
**Test Scenario 2:**

**Results from the Macy’s Dataset:**

For the Macys.csv test case, I used a minimum support of 70% and minimum confidence of 90%.

**  
**

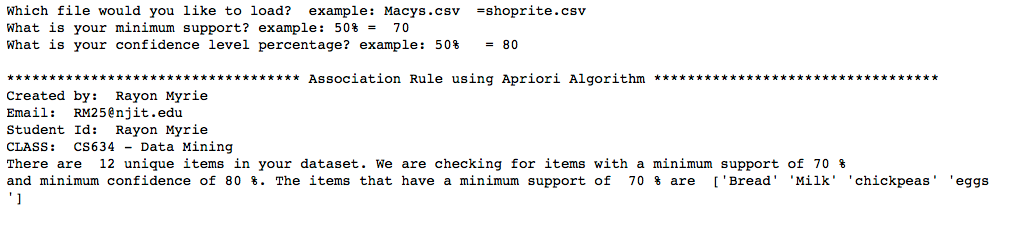
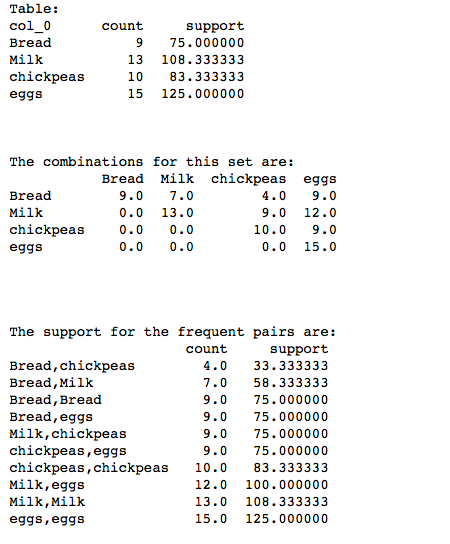
****

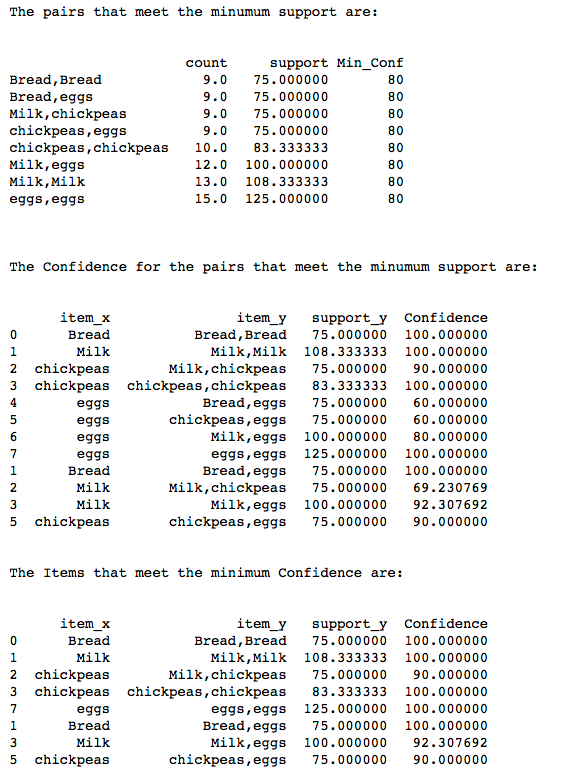
****

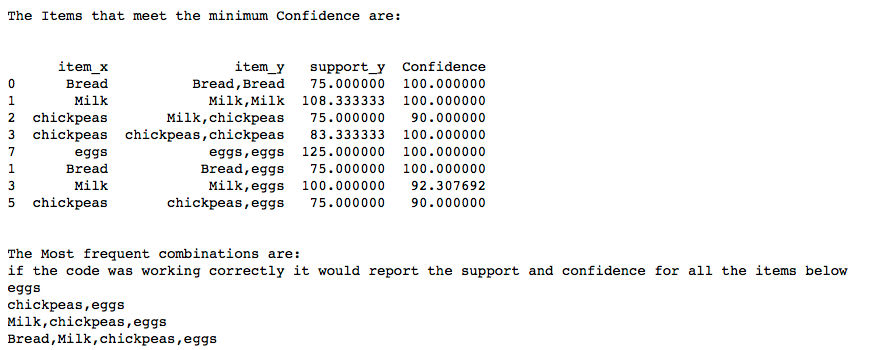
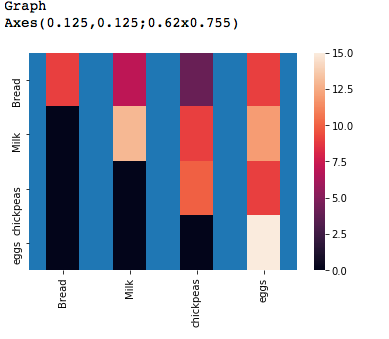
**Test Scenario 3:**

**Results from the Shoprite Dataset:**

For the shopright.csv test case, I used a minimum support of 70% and minimum confidence of 80%.

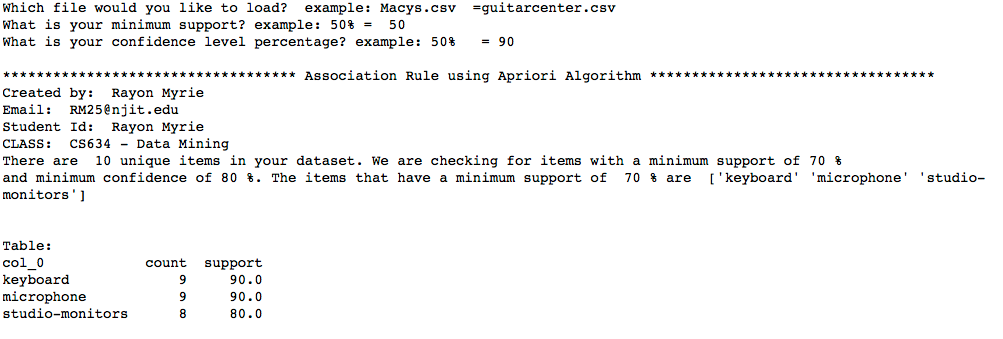
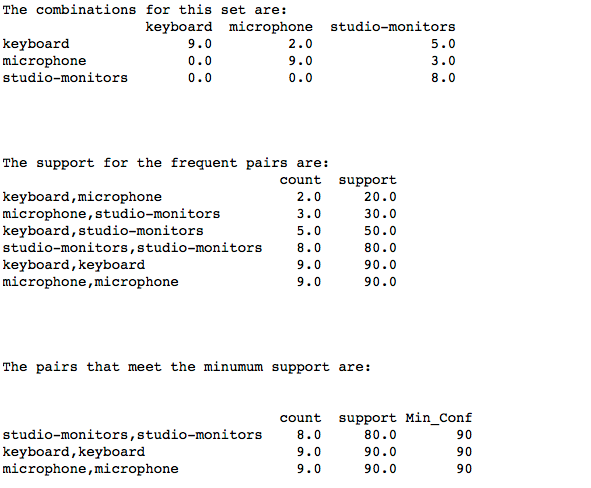
**  
**

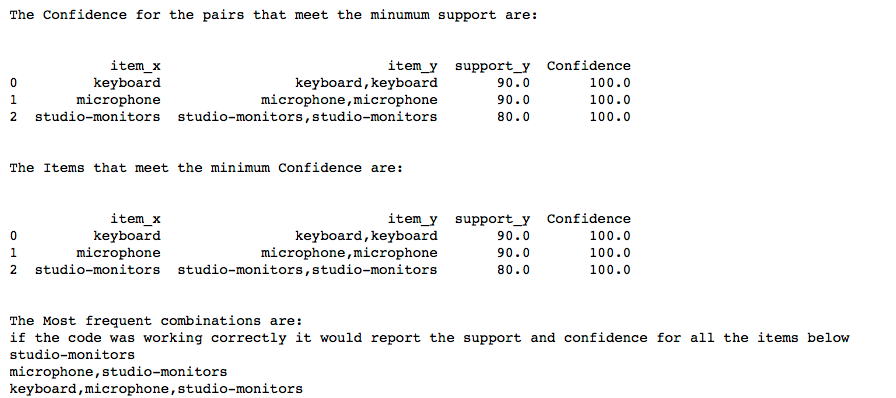
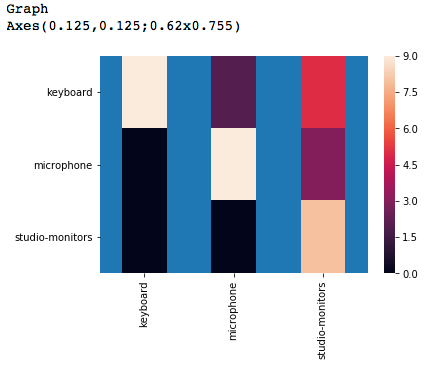
****

**  
  
Test Scenario 4:**

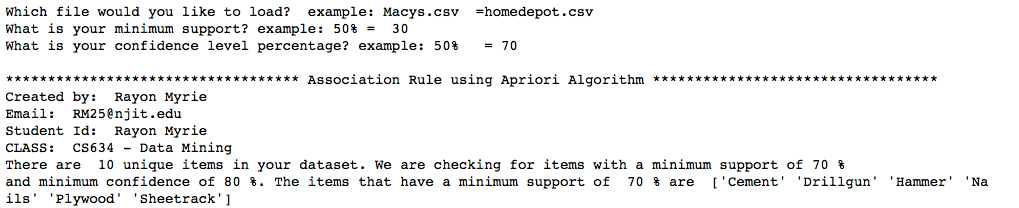
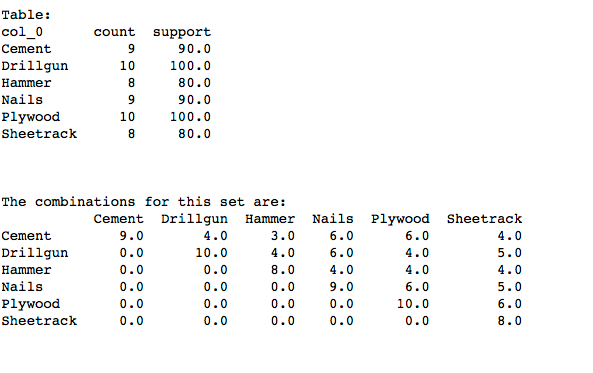
**Results from the Guitar Center Dataset:**

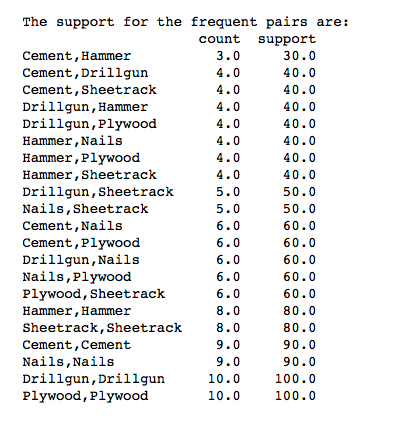
For the guitarcenter.csv test case, I used a minimum support of 50% and minimum confidence of 90%.

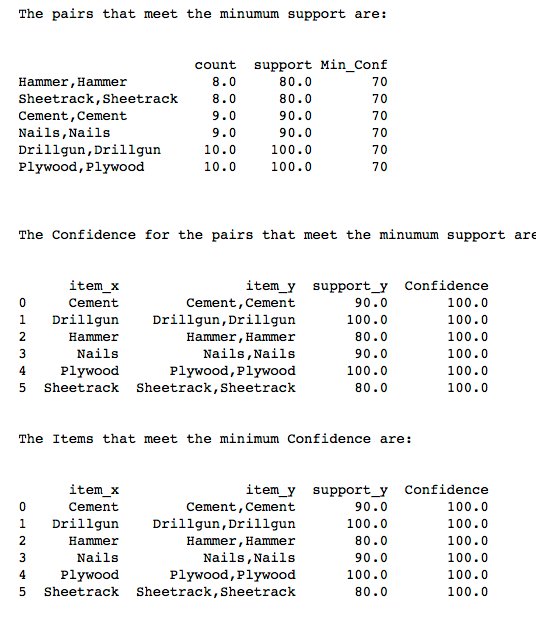
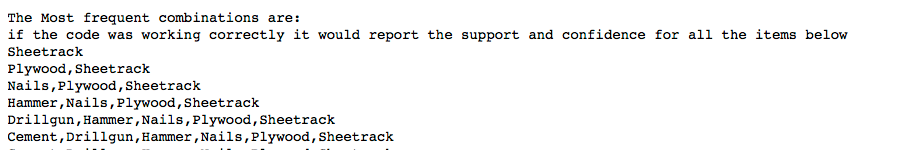
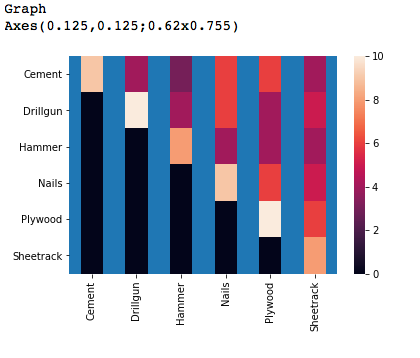
**  
**

**  
  
  
Test Scenario 5:**

**Results from the HomeDepot Dataset:**

For the homedepot.csv test case, I used a minimum support of 30% and minimum confidence of 70%.  
  
  


****

**  
  
**

**Deployment:**

The program was written in Anaconda’s Jupyter Notebook and can be deployed on any machine running Python 3.0. It contains 4 output CSV files, which are saved in the same directory as the program. 2 of these files are need for the program to work.

|  |  |
| --- | --- |
| Output file: | Descriptions |
| cartMatrix.csv | This is a matrix of all the items, in zeros and ones. |
| Freq\_Matrix.csv | This contains a matrix of all the Frequent items in zeros and ones |
| ind\_support.csv | This contains a table of all the items that meets the minimum support and is exported by the program and then used as a input to calculate support and confidence |
| pair\_support.csv | This contains a table with the pairs that meets the minimum support and is exported by the program and then used as a input to calculate support and confidence |

**Code:**Attached you will find a copy of the code. It will need to be copied and saved as a .py or ipynb file before it will run.



**Conclusion:**

This program will not correctly calculate the frequent item sets in a transaction; It will need to be modified to read vector matrices greater that 2 X 2.

**References**

* **Data Science from Scratch,** First Principles with Python**,**By Joel Grus
* **Data Mining: Concepts and Techniques, 3rd Edition**. Waltham, MA: Morgan Kaufmann
* **Real-World Data Mining: Applied Business Analytics and Decision Making**. Delen, D. (2015).Upper Saddle River, NJ: Pearson.
* **Code Project**:  <https://www.codeproject.com>
* **Pandas:** https://pandas.pydata.org