Experiment 9

Aim:
Experiment to implement Association mining algorithm(Apriori) using Rapid Miner and Python.

Theory:-
Association Mining Rule:
Association Rule Mining, as the name suggests, association rules are simple If/Then statements that help discover relationships between seemingly independent relational databases or other data repositories. Most machine learning algorithms work with numeric datasets and hence tend to be mathematical. However, association rule mining is suitable for non-numeric, categorical data and requires just a little bit more than simple counting. Association rule mining is a procedure which aims to observe frequently occurring patterns, correlations, or associations from datasets found in various kinds of databases such as relational databases, transactional databases, and other forms of repositories.

The Association rule is a learning technique that helps identify the dependencies between two data items. Based on the dependency, it then maps accordingly so that it can be more profitable. Association rule furthermore looks for interesting associations among the variables of the dataset. It is undoubtedly one of the most important concepts of Machine Learning and has been used in different cases such as association in data mining and continuous production, among others. However, like all other techniques, association in data mining, too, has its own set of disadvantages

Types of Association Rule Learning

Association rule learning can be divided into three algorithms:

- **Apriori Algorithm:** This algorithm uses frequent datasets to generate association rules. It is designed to work on the databases that contain transactions. This algorithm uses a breadth-first search and Hash Tree to calculate the itemset efficiently. It is mainly used for market basket analysis and helps to understand the products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.
- **Eclat Algorithm:** Eclat algorithm stands for Equivalence Class Transformation. This algorithm uses a depth-first search technique to find frequent itemsets in a transaction database. It performs faster than the Apriori Algorithm.
- **F-P Growth Algorithm:** The F-P growth algorithm stands for Frequent Pattern, and it is the improved version of the Apriori Algorithm. It represents the database in the form of a tree structure that is known as a frequent pattern or tree. The purpose of this frequent tree is to extract the most frequent patterns.

The Association mining function computes association rules. The set of the computed association rules is called a rule model. The first part of an association rule is called the rule body. The second part of an association rule is called rule head. You can build an association rule model by using the BuildRuleModel procedure.

- **Transaction tables:** If you use the Associations mining function on transaction tables, for example, retail transaction data, the rule body and the rule head might represent articles that occur in retail transactions, for example, chocolate, or candy.
  The association rule might look like this:

      If customers buy chocolate, they also buy candy.
- **Relation tables:** If you use the Associations mining function on relational tables, for example, customer data of a bank, the rule body and the rule head might represent column value pairs, for example, online-access=YES.
  The association rule might look like this:

      If online-access=YES then bankcard=YES
- **Confidence:** An association rule might not always be valid. For example, if the following association rule has a confidence value of 60%, this association rule indicates that if customers buy chocolate only in 60% of the cases they also buy candy:

      If chocolate then candy

  The degree of validity is indicated by the confidence value of an association rule. The value for confidence is shown in percent. It states how often the association rule head occurs in a transaction given that the rule body occurs in the transaction.
- **Support:** Another attribute of an association rule is the support value. The support value indicates the relative frequency that the conditions in the rule head and in the rule body hold.
  For example:
  If the following association rule has a support value of 2%, it means that 2% of all sales transactions contain chocolate and candy:

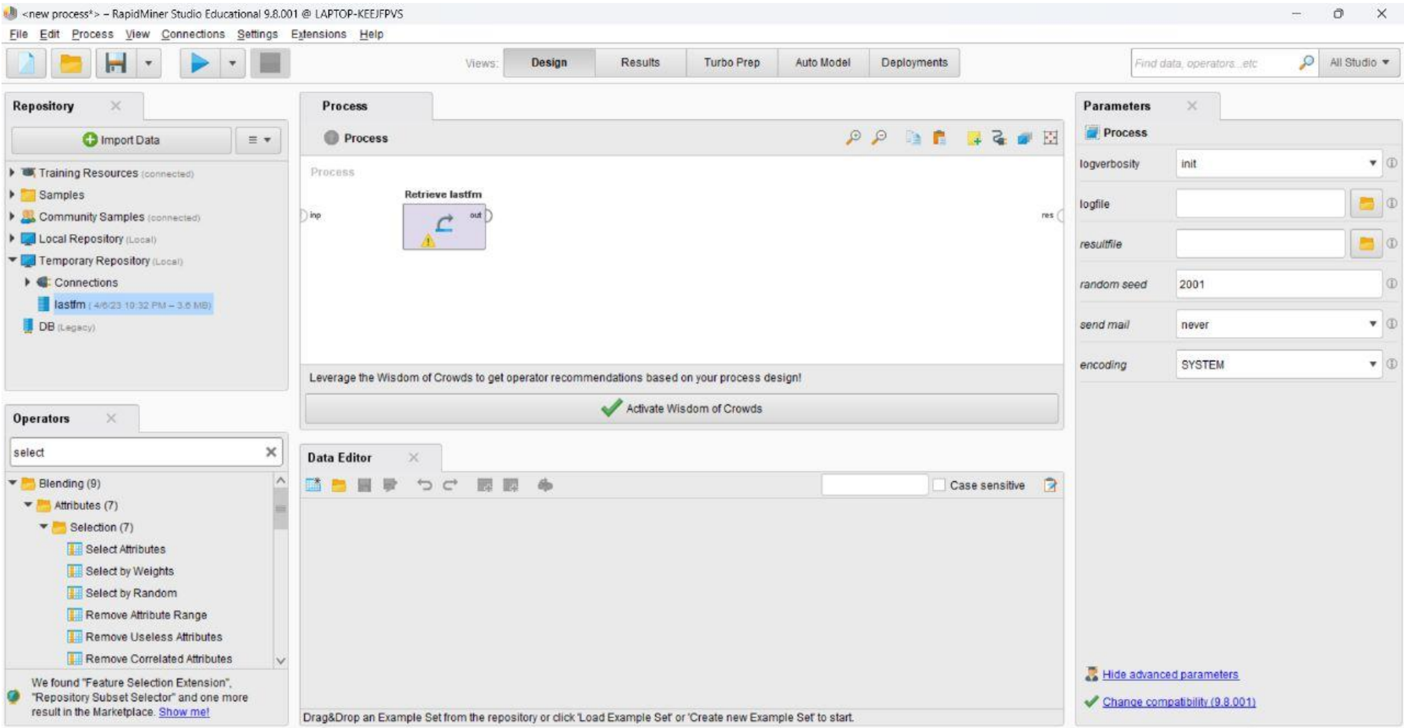
      If chocolate then candy

  If the following association rule has a support value of 4%, it means that for 4% of all records the values of the ONLINE_ACCESS column and the BANKCARD column are both YES.

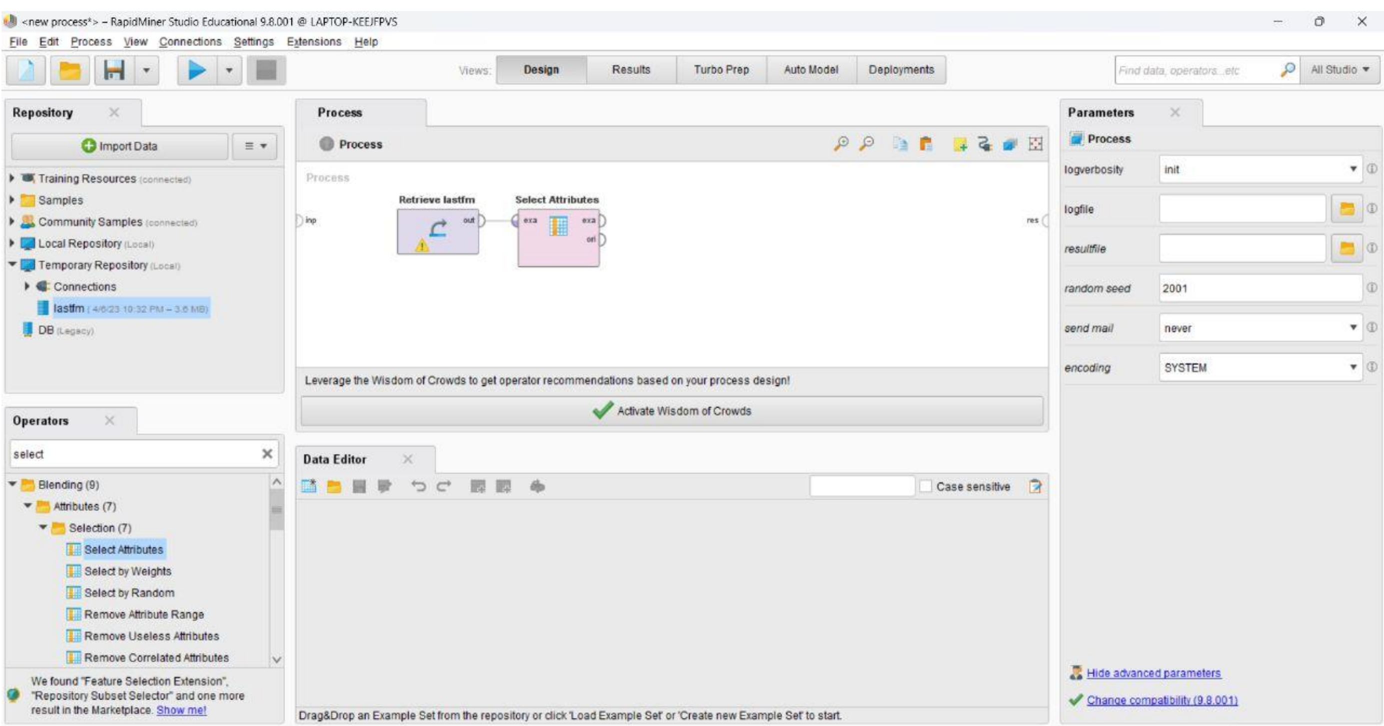
      If online-access=YES then bankcard=YES

Implementing Association Mining using Rapid Miner

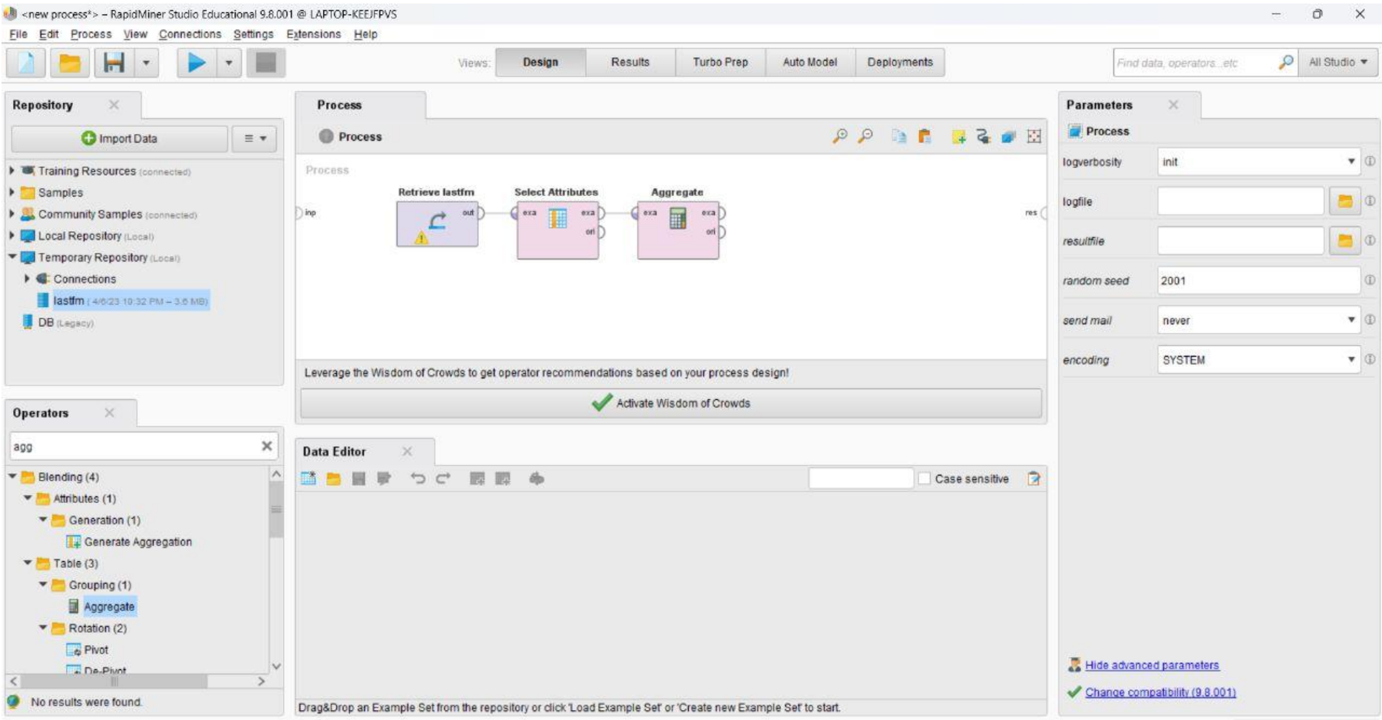
Import Dataset -



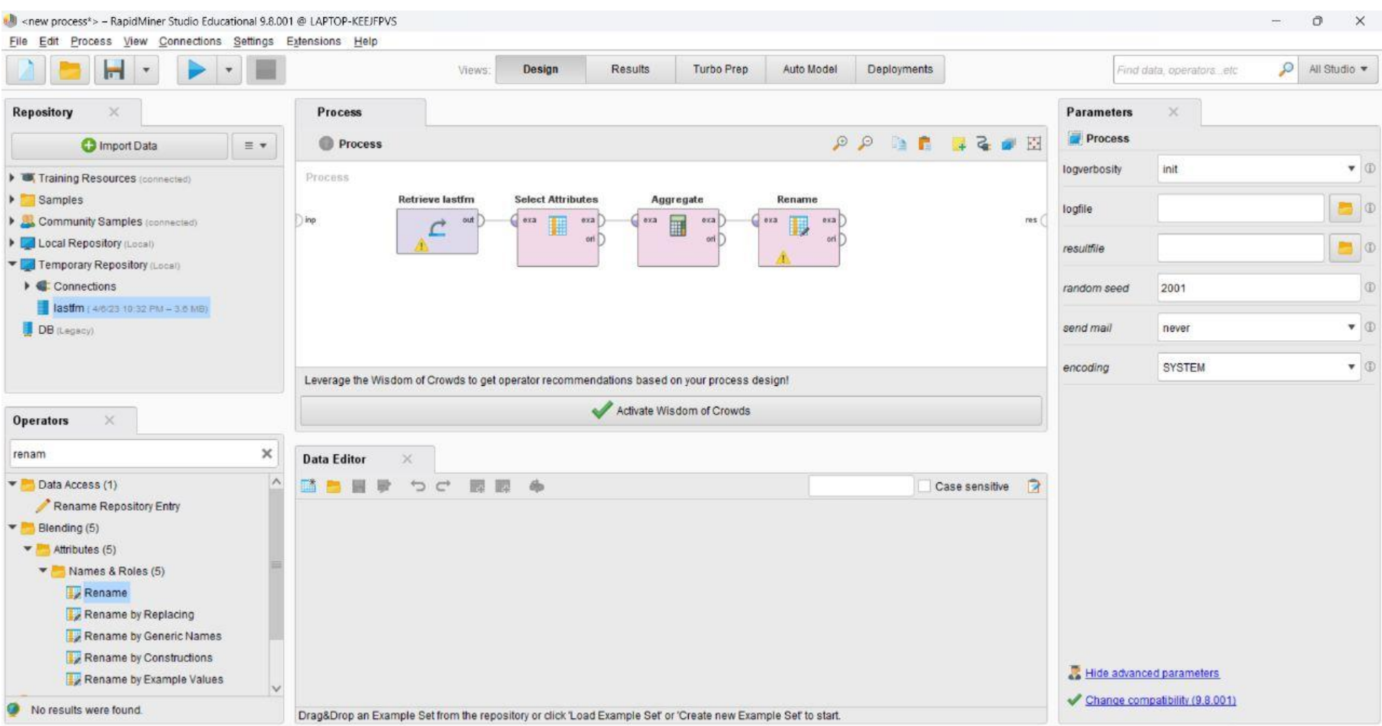
Select Attributes -



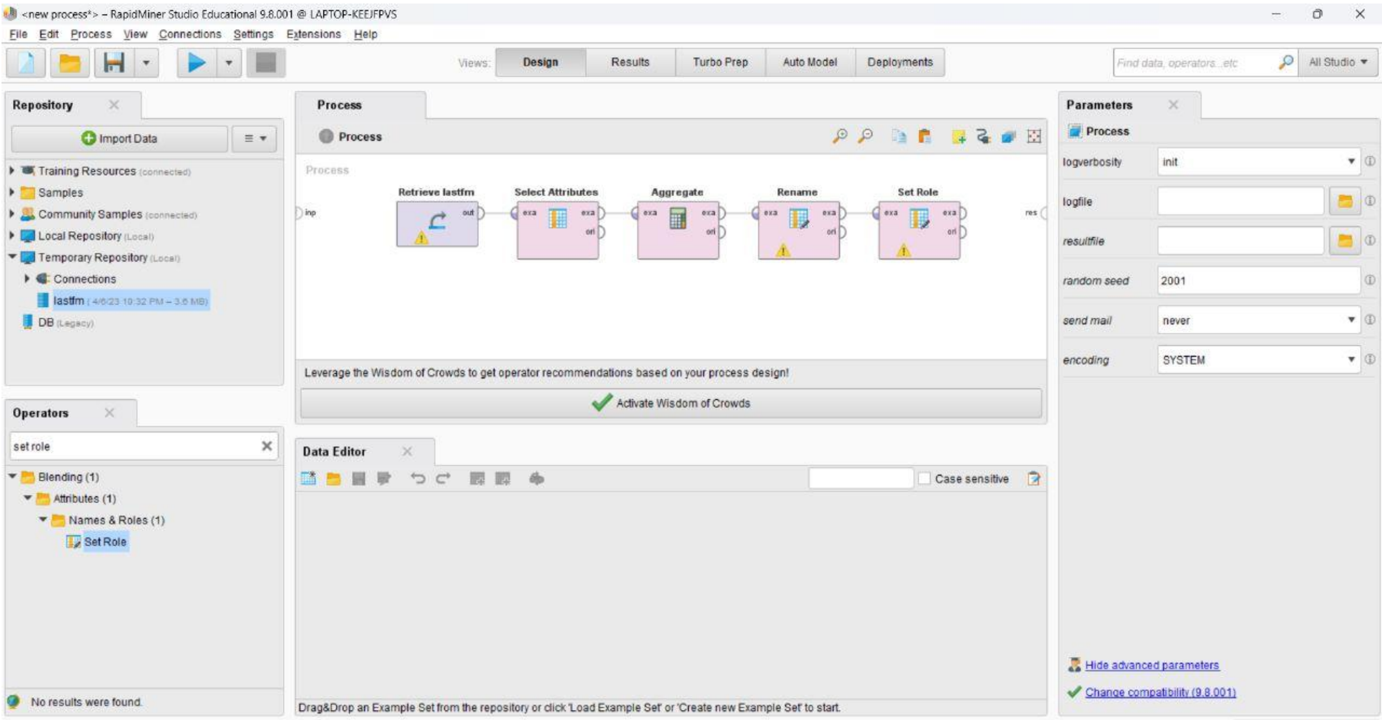
Data Aggregation -



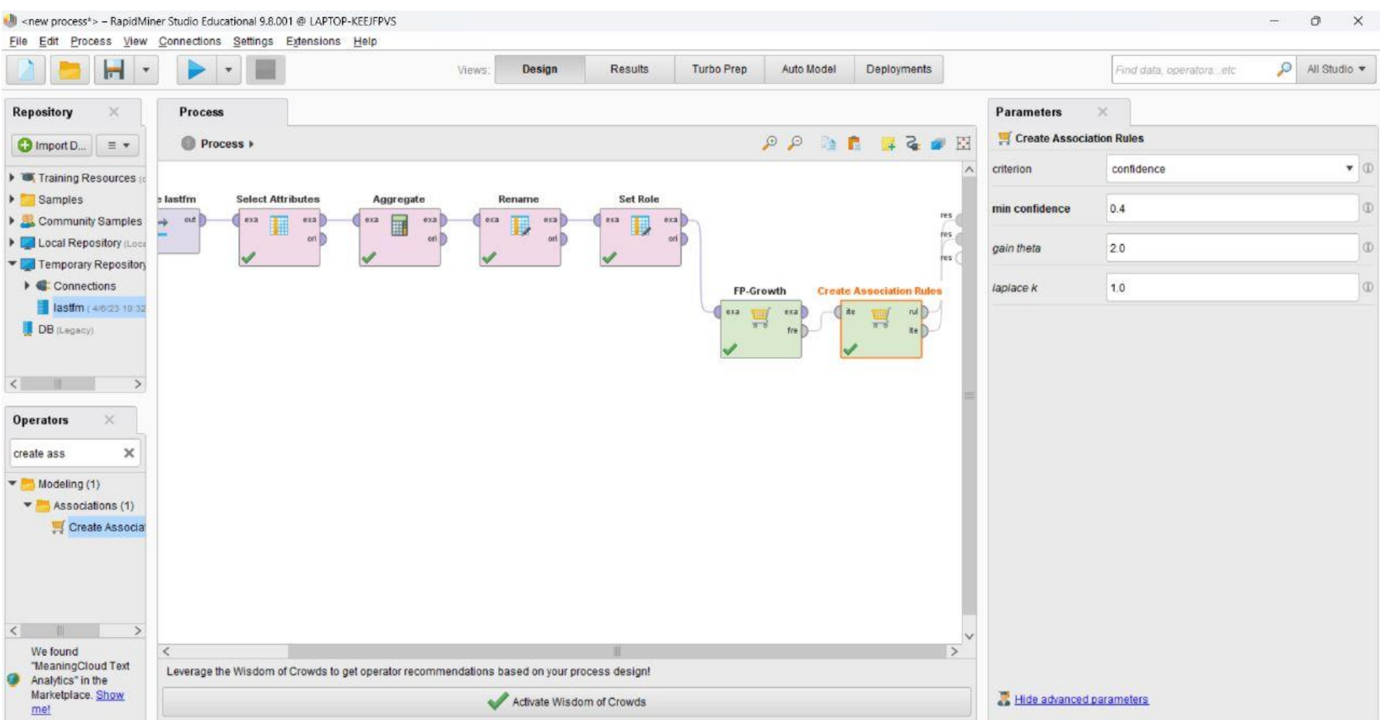
Rename -



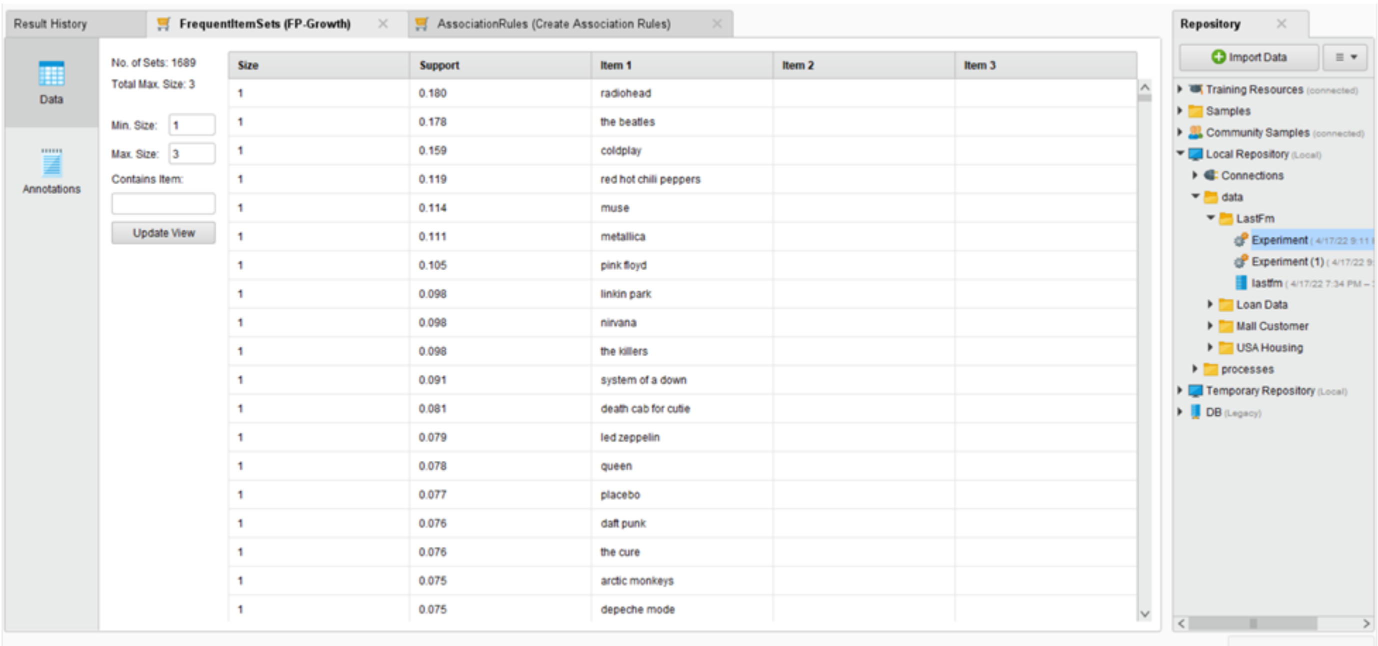
Set role -



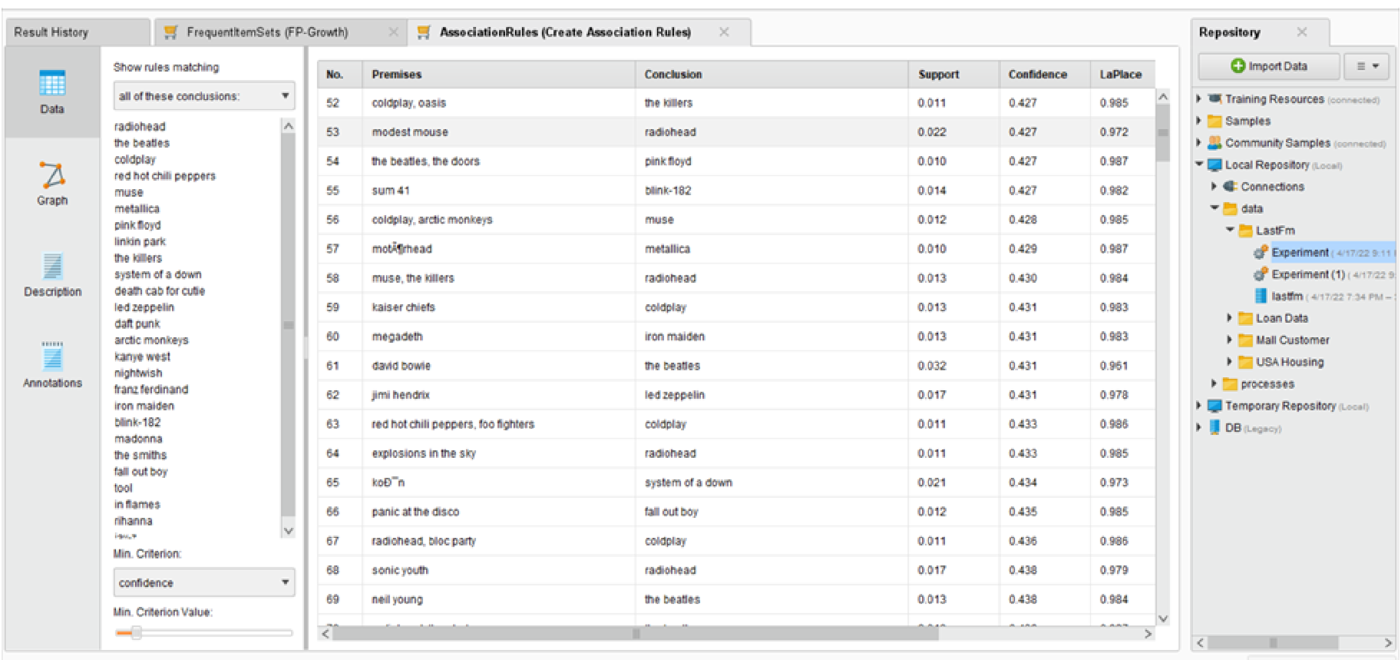
Create Association Rules -



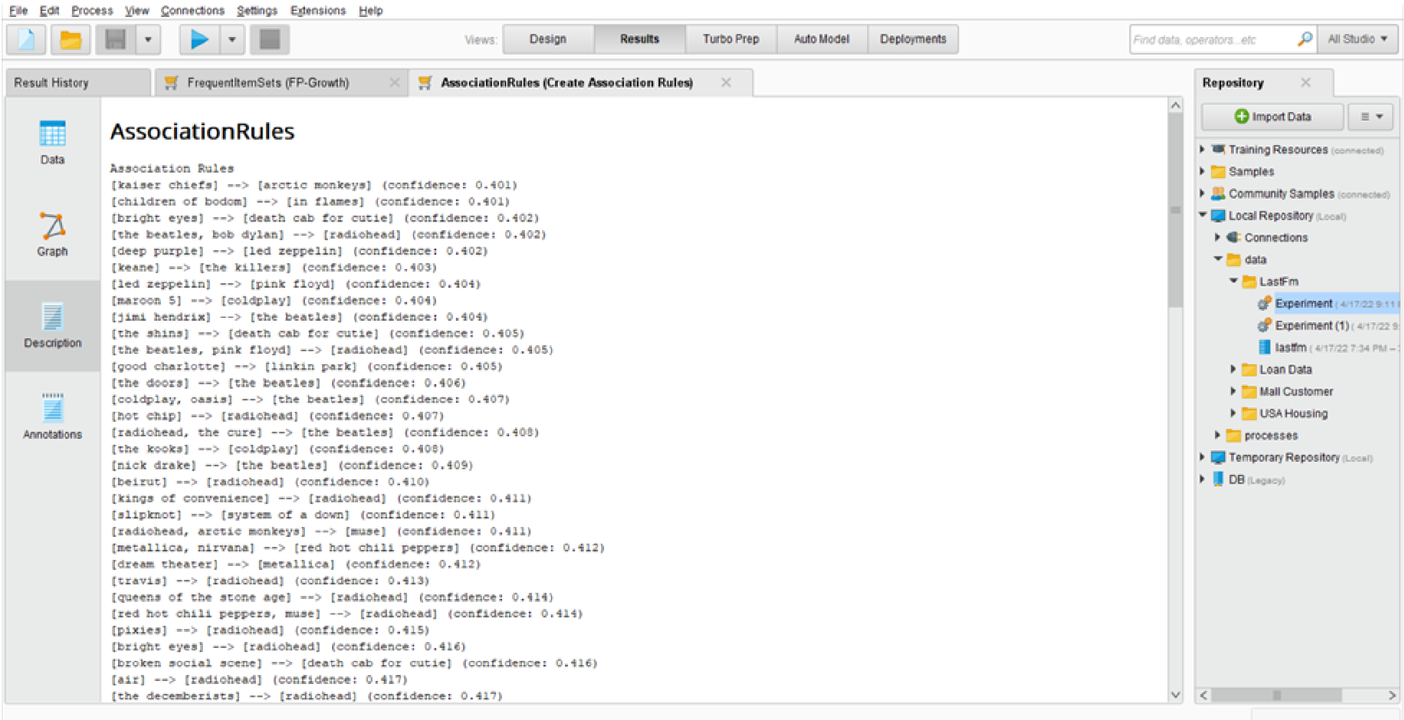
Results -



Size	Support	Item 1	Item 2	Item 3
1	0.180	radiohead		
1	0.178	the beatles		
1	0.159	coldplay		
1	0.119	red hot chili peppers		
1	0.114	muse		
1	0.111	metallica		
1	0.105	pink floyd		
1	0.098	linkin park		
1	0.098	nirvana		
1	0.098	the killers		
1	0.091	system of a down		
1	0.081	death cab for cutie		
1	0.079	led zeppelin		
1	0.078	queen		
1	0.077	placebo		
1	0.076	daft punk		
1	0.076	the cure		
1	0.075	arctic monkeys		
1	0.075	depeche mode		



No.	Premises	Conclusion	Support	Confidence	LaPlace
52	coldplay, oasis	the killers	0.011	0.427	0.985
53	modest mouse	radiohead	0.022	0.427	0.972
54	the beatles, the doors	pink floyd	0.010	0.427	0.987
55	sum 41	blink-182	0.014	0.427	0.982
56	coldplay, arctic monkeys	muse	0.012	0.428	0.985
57	motÃ¶rhead	metallica	0.010	0.429	0.987
58	muse, the killers	radiohead	0.013	0.430	0.984
59	kaiser chiefs	coldplay	0.013	0.431	0.983
60	megadeth	iron maiden	0.013	0.431	0.983
61	david bowie	the beatles	0.032	0.431	0.961
62	jimi hendrix	led zeppelin	0.017	0.431	0.978
63	red hot chili peppers, foo fighters	coldplay	0.011	0.433	0.986
64	explosions in the sky	radiohead	0.011	0.433	0.985
65	koÐ˜n	system of a down	0.021	0.434	0.973
66	panic at the disco	fall out boy	0.012	0.435	0.985
67	radiohead, bloc party	coldplay	0.011	0.436	0.986
68	sonic youth	radiohead	0.017	0.438	0.979
69	neil young	the beatles	0.013	0.438	0.984



Implementing Association Mining using Python

First install Apriori

```
[1] !pip install apyori

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5976 sha256=83cba0c69134721c54a4e223c2dedc61764335694dcfa3d2fdf3500653f469fc
  Stored in directory: /root/.cache/pip/wheels/32/2a/54/10c595515f385f3726642b10c60bf788029e8f3a1323e3913a
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

Importing Library

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from apyori import apriori
%matplotlib inline
import os
```

Importing dataset -

```
lastfm1 = pd.read_csv("./lastfm.csv")
lastfm = lastfm1
lastfm.head(5)
```

	user	artist	sex	country
0	1	red hot chili peppers	f	Germany
1	1	the black dahlia murder	f	Germany
2	1	goldfrapp	f	Germany
3	1	dropkick murphys	f	Germany
4	1	le tigre	f	Germany

Shape and description -

```
[5] lastfm.shape
```

```
(289955, 4)
```

```
[6] lastfm.describe()
```

	user
count	289955.000000
mean	9852.460447
std	5692.355041
min	1.000000
25%	4935.000000
50%	9838.000000
75%	14769.000000
max	19718.000000

Removing Duplicates

```
#Since we are looking at user artist listening patterns, we only take those attributes into consideration
lastfm = lastfm[['user','artist']]
```

```
[8] lastfm = lastfm.drop_duplicates()
lastfm.shape
```

```
(289953, 2)
```

```
[9] records = []
for i in lastfm['user'].unique():
    records.append(list(lastfm[lastfm['user'] == i]['artist'].values))
print(type(records))
```

```
<class 'list'>
```

Using Apriori Algorithm -

```
[10] association_rules = apriori(records, min_support=0.01, min_confidence=0.4, min_lift=3, min_length=2)
     association_results = list(association_rules)
```

```
[11] print(f"There are {len(association_results)} relations derived.")

     There are 91 relations derived.
```

Results of few rules-

```
[12] i=1
     for item in association_results:
         # first index of the inner list
         # Contains base item and add item
         pair = item[0]
         items = [x for x in pair]
         print("Rule: " + items[0] + " ==> " + items[1])

         # second index of the inner list
         print("Support: " + str(item[1]))

         # third index of the list located at 0th
         # of the third index of the inner list

         print("Confidence: " + str(item[2][0][2]))
         print("Lift: " + str(item[2][0][3]))
         print("=============================================================================")
         i+=1
         if i==10:
           break
```

```
Rule: a perfect circle ==> tool
Support: 0.016266666666666665
Confidence: 0.44283121597096187
Lift: 8.717149920688225
=============================================================================
Rule: kaiser chiefs ==> arctic monkeys
Support: 0.012533333333333334
Confidence: 0.4008528784648188
Lift: 5.3116547499755145
=============================================================================
Rule: rihanna ==> beyoncé
Support: 0.013933333333333334
Confidence: 0.46860986547085204
Lift: 10.88103402796096
=============================================================================
Rule: metallica ==> black sabbath
Support: 0.0172
Confidence: 0.45263157894736844
Lift: 4.06555310431768
=============================================================================
Rule: sum 41 ==> blink-182
Support: 0.014133333333333333
Confidence: 0.42741935483870963
Lift: 7.420474910394264
=============================================================================
Rule: breaking benjamin ==> linkin park
Support: 0.0108
Confidence: 0.4426229508196721
Lift: 4.507362024640246
=============================================================================
Rule: death cab for cutie ==> bright eyes
Support: 0.0152
Confidence: 0.4021164021164021
Lift: 4.944054124381993
=============================================================================
```

Comparison of these implementations

At min support count of 0.01 and confidence of 0.4 the following table represents the number of rules which are generated.

Python	**91**
Rapid Miner	**212**

Conclusion

Thus we have learnt what association mining is, how it is important and also implemented different ways to perform association mining.