D-ASSOC data

File Edit View Insert Runtime Tools Help Last edit in 2 days ago

Comment Share

+ Code + Text

Connect

```
!pip install apyori
```

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from apyori import apriori
import time
import time

path_name = '/content/'
store_data = pd.read_csv(path_name, header = None)

display(store_data.head())
print(store_data.shape)
```

	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
0	0.77	0.1	0.3	0.35	0.25	0.35	0.25	0.35	0.25	0.35	...	0.3	0.25	0.35	0.35	0.25	0.35	0.25	0.35	0.25	0.35
1	0.2	0.2	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	...	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
2	0.2	0.2	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	...	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
3	0.2	0.2	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	...	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
4	0.2	0.2	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	...	0.3	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15

5 rows × 24 columns

(1414, 24)

```
[ ] records = []
    for i in range(0, 1414):
        records.append([str(store_data.values[i, j]) for j in range(0, 24)])
    print(pre_records[0])
```

```
'nan' 'nan'
```

Support 20
Confidence 70

```
[ ] begin = time.time()
    association_rules = apriori(records, min_support=0.1, min_confidence=0.7)
    end = time.time()

    association_results = list(association_rules)
    print("There are 1 Relative Arrival ", format(association_results))
    print("Total runtime of the program is {:end - begin}".format(end-begin))
```

```
There are 1414 Relative Arrival
Total runtime of the program is 0.5014004.0700.5012 milliseconds
```

```
[ ] for i in range(0, len(association_results)):
        print(association_results[i])[1]
```

```
RelationRecord(ordered_positions=frozenset([('nan', 0)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 1)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 2)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 3)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 4)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 5)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 6)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 7)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 8)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 9)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 10)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 11)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 12)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 13)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 14)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 15)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 16)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 17)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 18)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 19)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 20)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 21)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 22)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 23)]), items=frozenset({'nan'}))
```

```
[ ] for item in association_results:
        # Find rules of the lower list
        # and the upper rules of the lower list
        pair = item[2]
        print("Rule: " + str(pair[0]))
        print("RHS: " + str(pair[1][0]))
        print("LHS: " + str(pair[1][1]))
        print("-"*20)

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

        print("Confidence: " + str(pair[1][2]))
        print("Lift: " + str(pair[1][3]))
        print("Relation: " + str(item[0]))
```

```
Traceback (most recent call last): ValueError: not enough values to unpack (expected 3, got 2)
```

Support 25
Confidence 80

```
[ ] begin = time.time()
    association_rules = apriori(records, min_support=0.25, min_confidence=0.8)
    end = time.time()

    association_results = list(association_rules)
    print("There are 11 Relative Arrival ", format(association_results))
    print("Total runtime of the program is {:end - begin}".format(end-begin))
```

```
There are 11 Relative Arrival
Total runtime of the program is 0.5014004.0700.5012 milliseconds
```

```
[ ] for i in range(0, len(association_results)):
        print(association_results[i])[1]
```

```
RelationRecord(ordered_positions=frozenset([('nan', 0)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 1)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 2)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 3)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 4)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 5)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 6)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 7)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 8)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 9)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 10)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 11)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 12)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 13)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 14)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 15)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 16)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 17)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 18)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 19)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 20)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 21)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 22)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 23)]), items=frozenset({'nan'}))
```

```
[ ] for item in association_results:
        # Find rules of the lower list
        # and the upper rules of the lower list
        pair = item[2]
        print("Rule: " + str(pair[0]))
        print("RHS: " + str(pair[1][0]))
        print("LHS: " + str(pair[1][1]))
        print("-"*20)

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

        print("Confidence: " + str(pair[1][2]))
        print("Lift: " + str(pair[1][3]))
        print("Relation: " + str(item[0]))
```

```
Traceback (most recent call last): ValueError: not enough values to unpack (expected 3, got 2)
```

Support 40
Confidence 90

```
[ ] begin = time.time()
    association_rules = apriori(records, min_support=0.4, min_confidence=0.9)
    end = time.time()

    association_results = list(association_rules)
    print("There are 1 Relative Arrival ", format(association_results))
    print("Total runtime of the program is {:end - begin}".format(end-begin))
```

```
There are 1 Relative Arrival
Total runtime of the program is 0.5014004.0700.5012 milliseconds
```

```
[ ] for i in range(0, len(association_results)):
        print(association_results[i])[1]
```

```
RelationRecord(ordered_positions=frozenset([('nan', 0)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 1)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 2)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 3)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 4)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 5)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 6)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 7)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 8)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 9)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 10)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 11)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 12)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 13)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 14)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 15)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 16)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 17)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 18)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 19)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 20)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 21)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 22)]), items=frozenset({'nan'}))
RelationRecord(ordered_positions=frozenset([('nan', 23)]), items=frozenset({'nan'}))
```

```
[ ] for item in association_results:
        # Find rules of the lower list
        # and the upper rules of the lower list
        pair = item[2]
        print("Rule: " + str(pair[0]))
        print("RHS: " + str(pair[1][0]))
        print("LHS: " + str(pair[1][1]))
        print("-"*20)

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

        print("Confidence: " + str(pair[1][2]))
        print("Lift: " + str(pair[1][3]))
        print("Relation: " + str(item[0]))
```

```
Traceback (most recent call last): ValueError: not enough values to unpack (expected 3, got 2)
```