SNAP

April 22, 2024

[22]:
```
import os
import tempfile
import shutil
import urllib
import zipfile
import pandas as pd
```

1.

[23]:
```
import pandas as pd

#  CSV
def read_snap_dataset(file_path):
    #  CSV       SOURCE, TARGET, RATING, TIME
    try:
        df = pd.read_csv(file_path, header=None, names=["SOURCE", "TARGET",␣
 ↪"RATING", "TIME"])
        return df
    except FileNotFoundError:
        print("          ")

#  CSV
file_path = "./input/soc-sign-bitcoinalpha.csv"

#
snap_dataset = read_snap_dataset(file_path)

snap_dataset
```

[23]:

	SOURCE	TARGET	RATING	TIME
0	7188	1	10	1407470400
1	430	1	10	1376539200
2	3134	1	10	1369713600
3	3026	1	10	1350014400
4	3010	1	10	1347854400
…	…	…	…	…
24181	7604	7601	10	1364270400
24182	7601	7604	10	1364270400

```
24183    7604    7602      10  1364270400
24184    7602    7604      10  1364270400
24185    7604    7603     -10  1364270400

[24186 rows x 4 columns]
```

[24]:
```
snap_dataset = snap_dataset.dropna()  #
snap_dataset
```

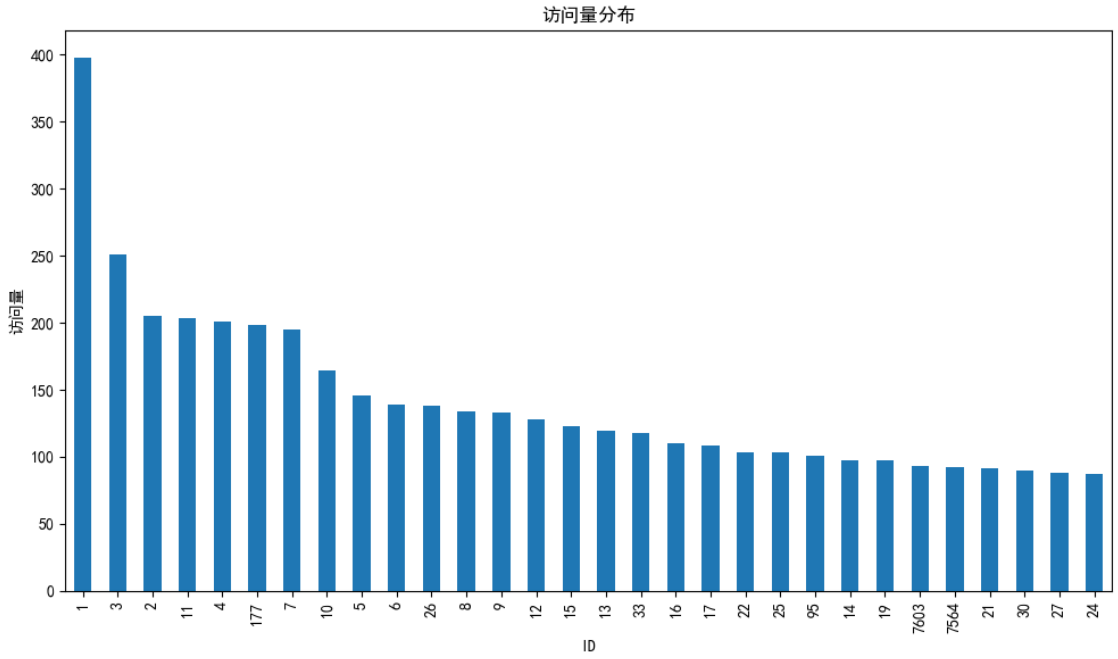
[24]:
```
        SOURCE  TARGET  RATING        TIME
0         7188       1      10  1407470400
1          430       1      10  1376539200
2         3134       1      10  1369713600
3         3026       1      10  1350014400
4         3010       1      10  1347854400
…          …     …       …          …
24181    7604    7601      10  1364270400
24182    7601    7604      10  1364270400
24183    7604    7602      10  1364270400
24184    7602    7604      10  1364270400
24185    7604    7603     -10  1364270400

[24186 rows x 4 columns]
```

[25]:
```
#
page_visits = snap_dataset['TARGET'].value_counts()

#
most_visited_pages = page_visits.head(30)

#
print("     ")
print(most_visited_pages)
```

```
TARGET
1       398
3       251
2       205
11      203
4       201
177     198
7       195
10      164
5       146
6       139
26      138
8       134
```

```
9       133
12      128
15      123
13      119
33      118
16      110
17      108
22      103
25      103
95      101
14       97
19       97
7603     93
7564     92
21       91
30       90
27       88
24       87
Name: count, dtype: int64
```

[26]:
```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']  #       SimHei

#
plt.figure(figsize=(10, 6))
most_visited_pages.plot(kind='bar')
plt.xlabel("ID")
plt.ylabel("  ")
plt.title("   ")
plt.tight_layout()  #
plt.show()
```



30 “ ” Long Tail Distribution

3. : Apriori FP-growth Apriori

[30]:
```
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
#      Apriori
user_data = []
last_user = '24186'
tmp = []

for index, row in snap_dataset.iterrows():
    user_id = row['SOURCE']
    vroot_id = row['TARGET']

    if user_id == last_user:
        tmp.append(vroot_id)
    else:
        user_data.append(tmp)
        tmp = []
        tmp.append(vroot_id)
    last_user = user_id

user_data.append(tmp)
# user_data
```

```
te = TransactionEncoder()
data_encoded = te.fit_transform(user_data)
df = pd.DataFrame(data_encoded, columns=te.columns_)

df
```

[30]:

```
         1      2      3      4      5      6      7      8      9      10    \
0      False  False  False  False  False  False  False  False  False  False
1       True  False  False  False  False  False  False  False  False  False
2       True  False  False  False  False  False  False  False  False  False
3       True  False  False  False  False  False  False  False  False  False
4       True  False  False  False  False  False  False  False  False  False
…       …     …      …      …      …      …      …      …      …      …
12869  False  False  False  False  False  False  False  False  False  False
12870  False  False  False  False  False  False  False  False  False  False
12871  False  False  False  False  False  False  False  False  False  False
12872  False  False  False  False  False  False  False  False  False  False
12873  False  False  False  False  False  False  False  False  False  False

       …   7595   7596   7597   7598   7599   7600   7601   7602   7603  \
0      …  False  False  False  False  False  False  False  False  False
1      …  False  False  False  False  False  False  False  False  False
2      …  False  False  False  False  False  False  False  False  False
3      …  False  False  False  False  False  False  False  False  False
4      …  False  False  False  False  False  False  False  False  False
…    …    …      …      …      …      …      …      …      …      …
12869  …  False  False  False  False  False  False   True  False  False
12870  …  False  False  False  False  False  False  False  False  False
12871  …  False  False  False  False  False  False  False   True  False
12872  …  False  False  False  False  False  False  False  False  False
12873  …  False  False  False  False  False  False  False  False   True

        7604
0      False
1      False
2      False
3      False
4      False
…      …
12869  False
12870   True
12871  False
12872   True
12873  False

[12874 rows x 3754 columns]
```

[47]:
```
#  Apriori
frequent_itemsets = apriori(df, min_support=0.01, use_colnames=True)

#
print("Frequent Itemsets:")
# print(frequent_itemsets)
frequent_itemsets
```

```
Frequent Itemsets:
```

[47]:

	support	itemsets
0	0.030915	(1)
1	0.015924	(2)
2	0.019497	(3)
3	0.015613	(4)
4	0.011341	(5)
..	…	…
437	0.001010	(7602, 7604, 7598)
438	0.001010	(7601, 7604, 7599)
439	0.001010	(7602, 7604, 7599)
440	0.001010	(7601, 7604, 7598, 7599)
441	0.001010	(7602, 7604, 7598, 7599)

[442 rows x 2 columns]

[49]:
```
rules = association_rules(frequent_itemsets, metric="confidence",␣
 ↪min_threshold=0.3)

#
print("\nAssociation Rules:")
# print(rules)
rules
```

```
Association Rules:
```

[49]:

	antecedents	consequents	antecedent support	consequent support	\
0	(47)	(31)	0.003418	0.005593	
1	(47)	(145)	0.003418	0.006525	
2	(136)	(177)	0.002563	0.015380	
3	(7595)	(177)	0.003651	0.015380	
4	(7598)	(7599)	0.001476	0.001476	
..	…	…	…	…	
79	(7604, 7599)	(7602, 7598)	0.001243	0.001010	
80	(7598, 7599)	(7602, 7604)	0.001165	0.001165	
81	(7602)	(7604, 7598, 7599)	0.001320	0.001165	
82	(7598)	(7602, 7604, 7599)	0.001476	0.001010	
83	(7599)	(7602, 7604, 7598)	0.001476	0.001010	

```
     support  confidence        lift  leverage  conviction  zhangs_metric
0   0.001243    0.363636   65.020202  0.001224    1.562640       0.987997
1   0.001087    0.318182   48.765152  0.001065    1.457097       0.982853
2   0.001010    0.393939   25.614019  0.000970    1.624623       0.963428
3   0.001243    0.340426   22.134537  0.001187    1.492811       0.958320
4   0.001165    0.789474  534.930748  0.001163    4.742990       0.999606
..       …           …          …         …           …              …
79  0.001010    0.812500  804.625000  0.001009    5.327948       1.000000
80  0.001010    0.866667  743.831111  0.001008    7.491261       0.999821
81  0.001010    0.764706  656.321569  0.001008    4.245048       0.999797
82  0.001010    0.684211  677.578947  0.001008    3.163469       1.000000
83  0.001010    0.684211  677.578947  0.001008    3.163469       1.000000

[84 rows x 10 columns]
```

[50]:
```
#
rules['lift'] = rules['lift'].apply(lambda x: round(x, 2))

#
observed = rules['support'] * len(df)  #
expected = rules['antecedent support'] * rules['consequent support'] * len(df)
 ↪#
chi_squared = ((observed - expected) ** 2 / expected).sum()  #

#
print("    :")
print(rules[['antecedents', 'consequents', 'lift']])
print("\n    :")
print("  :", chi_squared)
```

```
    :
     antecedents         consequents    lift
0           (47)                (31)   65.02
1           (47)               (145)   48.77
2          (136)               (177)   25.61
3         (7595)               (177)   22.13
4         (7598)              (7599)  534.93
..           …                   …     …
79  (7604, 7599)        (7602, 7598)  804.62
80  (7598, 7599)        (7602, 7604)  743.83
81        (7602)  (7604, 7598, 7599)  656.32
82        (7598)  (7602, 7604, 7599)  677.58
83        (7599)  (7602, 7604, 7598)  677.58

[84 rows x 3 columns]

   :
```

: 626814.4944881184

4. ”support” “confidence” ”lift”

0 (47) (31) 65.02 1 (47) (145) 48.77 2 (136) (177) 25.61 3 (7595) (177) 22.13

[ ]: