人，事件，演算法

RC, Data Team, CathayBk

客戶歷程

Roger

金融行為 刷了**台塑卡**, 消費了**3000元**於**莫爾頓牛排**

2015/05/27 20:48 - 於**忠孝東路XXX 45F**

金融行為 兌換了**紅利商品 - 茶葉蛋**

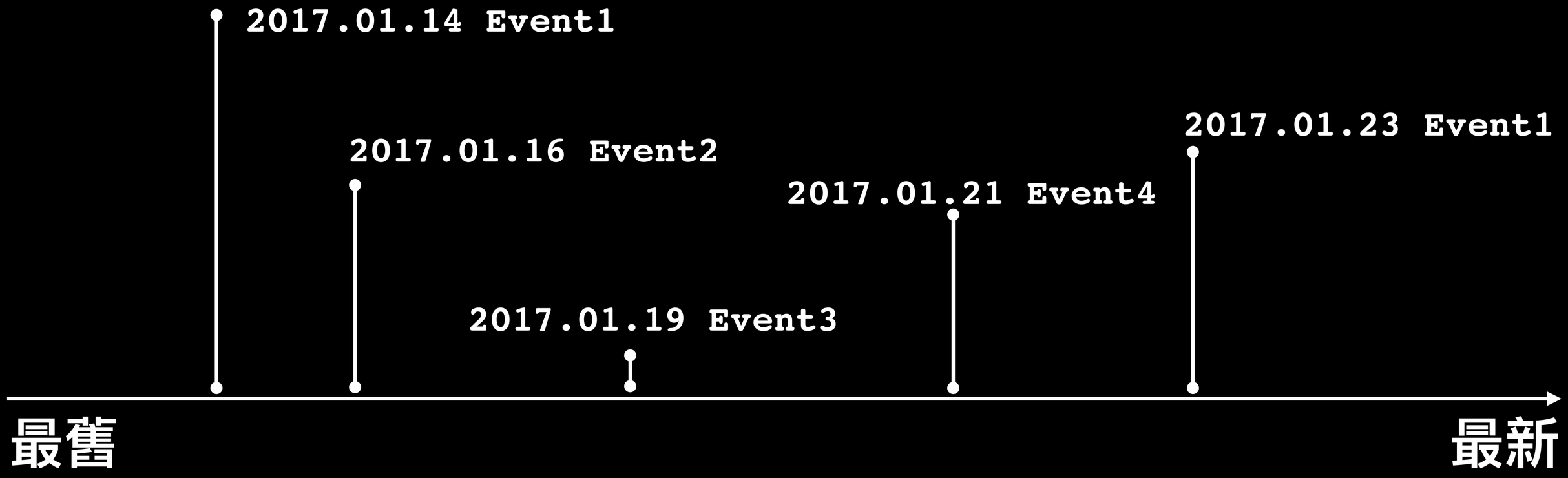
2015/04/29 12:08 - 於**MyRewards**

金融行為 開辦了**信用卡戶 - 長榮無限**

2015/02/29 9:34 - 於**信義分行**

最舊 → **最新**

客戶歷程


2017.01.14 Event1
2017.01.16 Event2
2017.01.19 Event3
2017.01.21 Event4
2017.01.23 Event1
最舊
最新

people_name	timestamp	event
people_0001	1497235308	BIJGQW
people_0001	1496826431	QESBUC
people_0001	1496895498	OFYWUW
people_0001	1497033773	SGNQMI
people_0001	1497082655	HCSLCD
people_0001	1496806746	HGYHDQ
people_0001	1497354271	QAUAGR
people_0001	1497242682	ITISOQ
people_0001	1496980722	JOLLAM
people_0001	1496951831	LFESHV
people_0001	1497369453	LVMCQT
…		
…		
people_0003	1496939024	OHLQUS
people_0003	1497238367	DWTSQF
people_0003	1497276203	OKCOMB
people_0003	1496970950	XTVMPR
people_0003	1497398979	JDMQDN
people_0003	1496996677	QDQCRW
people_0003	1497021217	NIYXMI
people_0004	1497404216	TMEPPP
people_0004	1497141274	BICRHM
people_0004	1496824223	PJKKXV
…		
…		

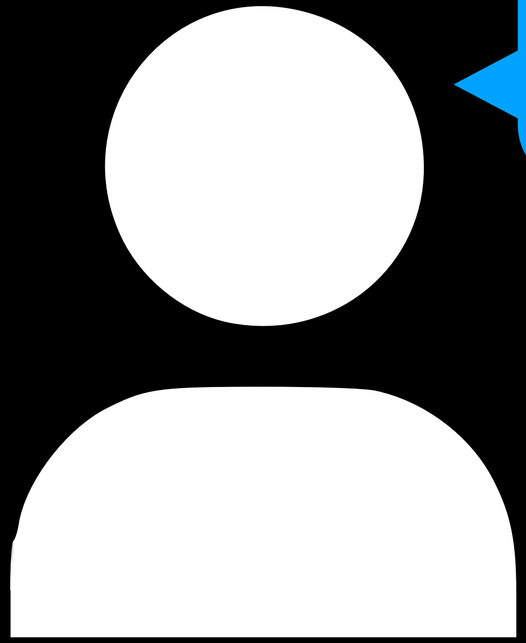
事件發生時間

事件發生人

某事件發生

資料格式

people_name	timestamp	event
people_0001	1497235308	BIJGQW
people_0001	1496826431	QESBUC
people_0001	1496895498	OFYWUW
people_0001	1497033773	SGNQMI
people_0001	1497082655	HCSLCD
people_0001	1496806746	HGYHDQ
people_0001	1497354271	QAUAGR
people_0001	1497242682	ITISOQ
people_0001	1496980722	JOLLAM
people_0001	1496951831	LFESHV
people_0001	1497369453	LVMCQT
…		
…		
people_0003	1496939024	OHLQUS
people_0003	1497238367	DWTSQF
people_0003	1497276203	OKCOMB
people_0003	1496970950	XTVMPR
people_0003	1497398979	JDMQDN
people_0003	1496996677	QDQCRW
people_0003	1497021217	NIYXMI
people_0004	1497404216	TMEPPP
people_0004	1497141274	BICRHM
people_0004	1496824223	PJKKXV
…		
…		



資料分析101

Data Profile

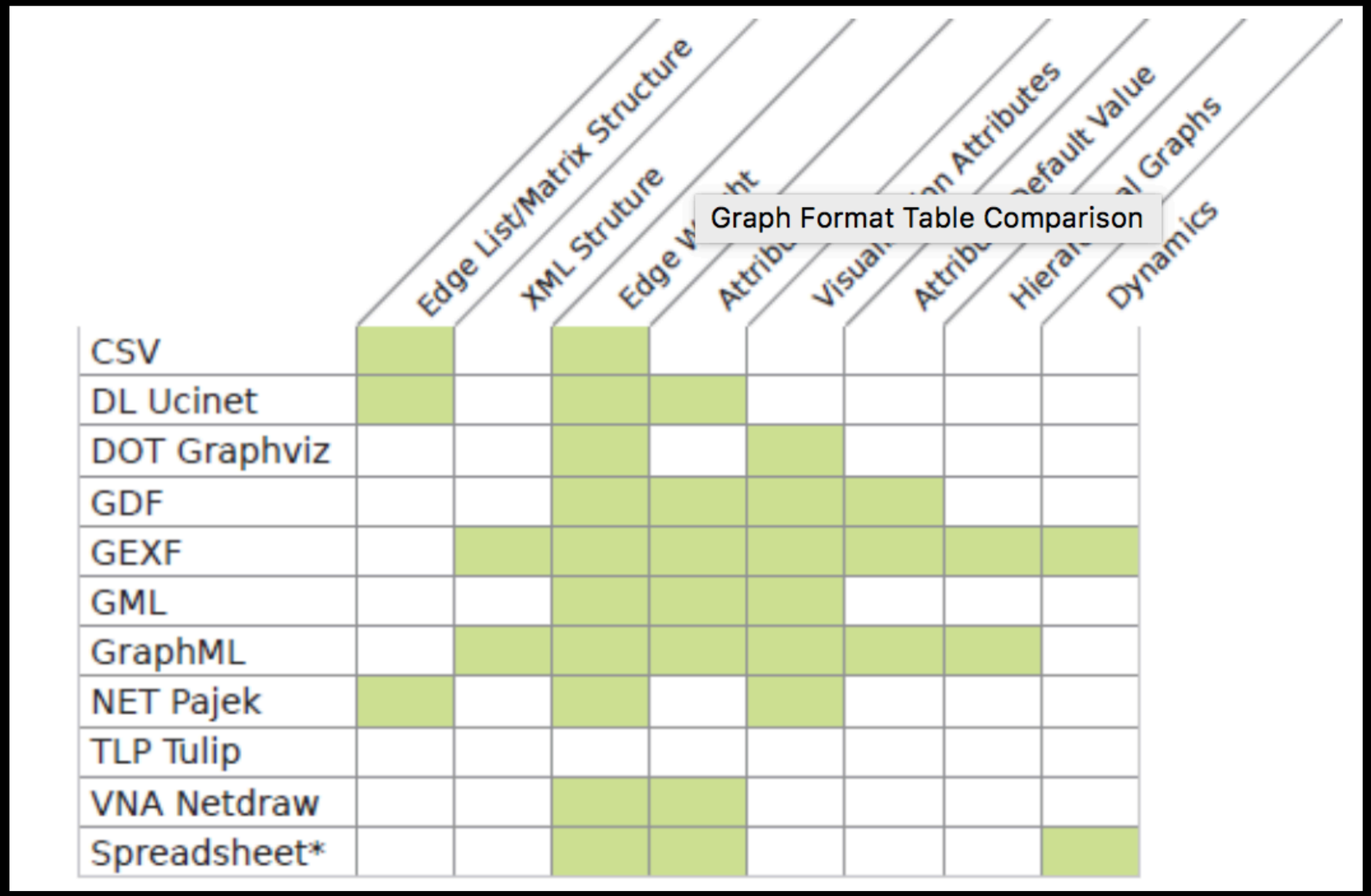
給大家十分鐘

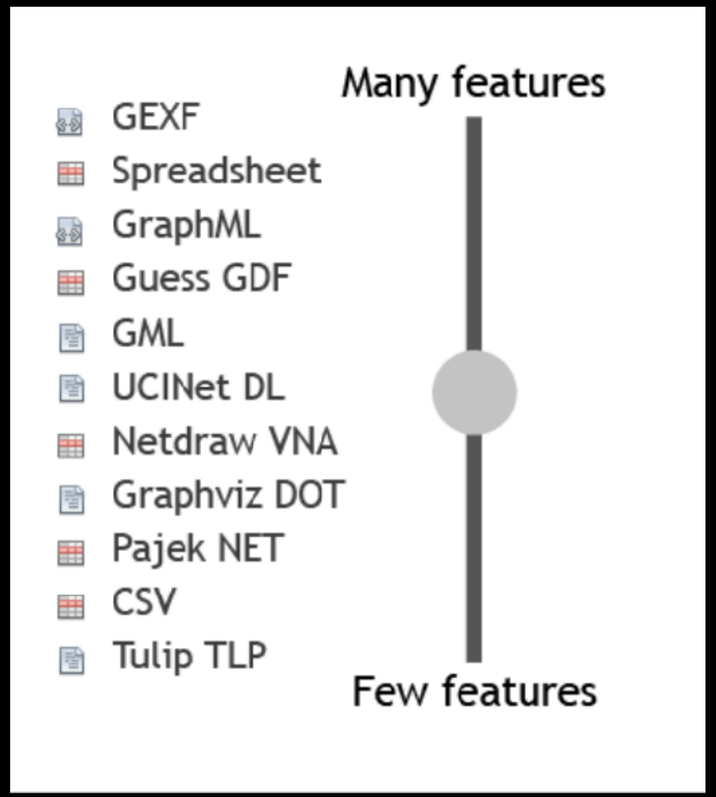
用任何方式做 Data Profiling

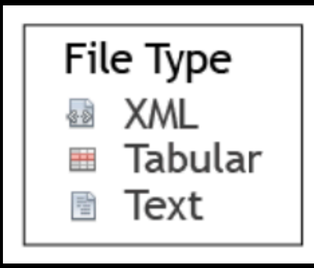
Pre-requirements

- pip install networkx
- pip install pygraphviz
  - If failed in mac, use the following command instead
    - pip install pygraphviz –install-option="--include-path=/usr/local/include/graphviz/" –install-option="--library-path=/usr/local/lib/graphviz"
- download visualization tool
  - https://gephi.org
- community detection installation
  - pip install community
  - https://github.com/taynaud/python-louvain

Graphic File Format







DOT Format

strict: 無向圖

district: 有向圖

```
strict graph  {
    nodeA -- nodeB [key=“edge",weight=70];
    nodeA -- nodeC [key="edge",weight=37];
    nodeA -- nodeN [key="edge",weight=102];
    nodeA -- nodeM [key="edge",weight=18];
    nodeA -- nodeOO [key=“edge",weight=20];
    nodeB -- nodeCC [key=“edge",weight=31];
    nodeB -- nodeDD [key="edge",weight=49];
    nodeB -- nodeZ [key="edge",weight=48];
    …
    …
    nodeZZ -- nodeA [key="edge",weight=34];
    nodeZZ -- nodeB [key=“edge",weight=60];
…
}
```

節點

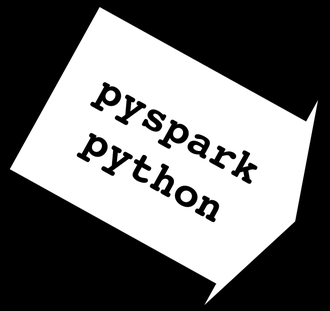
邊上面的屬性

兩節點有連結

節點

動手實作

people_name	timestamp	event
people_0001	1497235308	A
people_0001	1496826431	B
people_0002	1496895498	A
people_0002	1497033773	C
people_0003	1497082655	A
people_0004	1496806746	B
people_0005	1497354271	Z
…		
…		
people_0007	1496939024	B
people_0007	1497238367	C
people_0007	1497276203	D
people_0007	1496970950	A
people_0009	1497398979	Z
people_0009	1496996677	B
…		

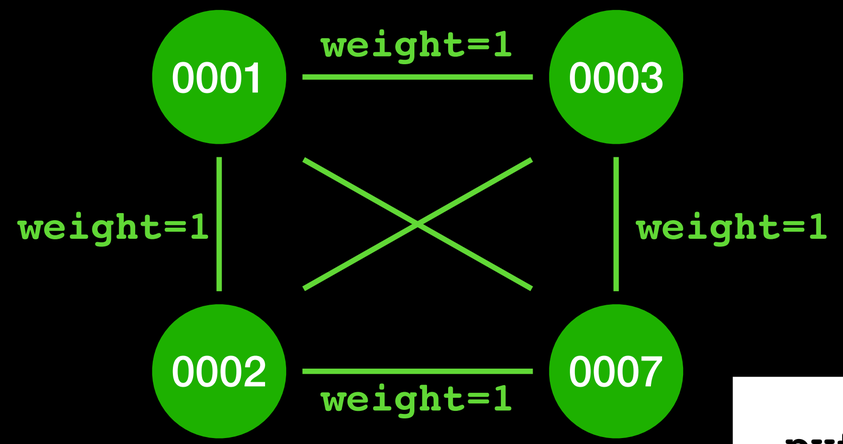


event	people
A	people_0001,people_0002,people_0003,people_0007…
B	people_0001,people_0004,people_0007,people_0009…
C	people_0002,people_0007…
D	people_0007…
…	
…	
…	

動手實作

event people
A people_0001,people_0002,people_0003,people_0[illegible]
B people_0001,people_0004,people_0007,people_000[illegible]
C people_0002,people_0007…
D people_0007…
…
…
…

python



python networks

```
strict graph  {
    people_0029 -- people_0028 [key=“edge,weight=70];
    people_0029 -- people_0020 [key=“edge",weight=37];
    people_0029 -- people_0023 [key="edge",weight=102];
    people_0029 -- people_0025 [key="edge",weight=18];
    people_0029 -- people_0024 [key="edge",weight=20];
    people_0029 -- people_0027 [key="edge",weight=31];
    …
    people_0029 -- people_0140 [key="edge",weight=101];
    people_0029 -- people_0144 [key="edge",weight=50];
    people_0029 -- people_0145 [key=“edge”,weight=43];
    …
    }
```

程式碼講解

宣告一個無向圖 →

```
g = nx.Graph()
with open("../data/event_relation.tsv", "rb") as in_file:
    in_file.next()
    for line in in_file:
        event, people = line.strip().split(sep)
        people = people.split(",")

        for curr_idx in range(len(people)):
            for next_idx in range(curr_idx+1, len(people)):
                curr_person = people[curr_idx]
                next_person = people[next_idx]

                if g.has_edge(curr_person, next_person):
                    data = g.get_edge_data(curr_person, next_person)
                    g.add_edge(curr_person, next_person, key="edge", weight=data['weight']+1)
                else:
                    g.add_edge(curr_person, next_person, weight=1)
```

檢查graph g 是否有這個邊，若有

取出這個邊 →

更新新的邊 →

放入新的邊 →

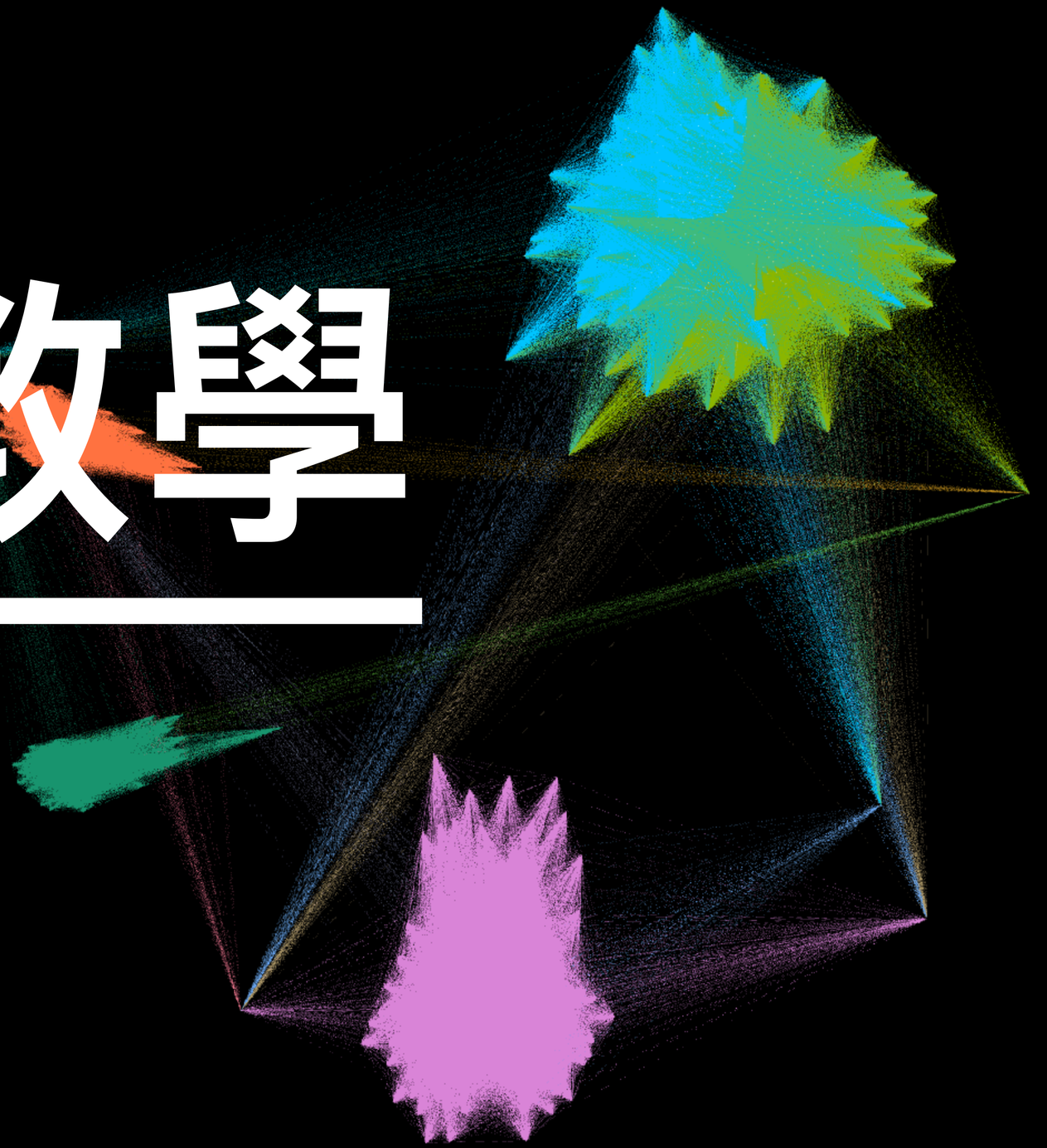
載入函式庫 →

```
import pygraphviz
from networkx.drawing.nx_agraph import write_dot
write_dot(g, "../data/find_similar_people.dot")
```

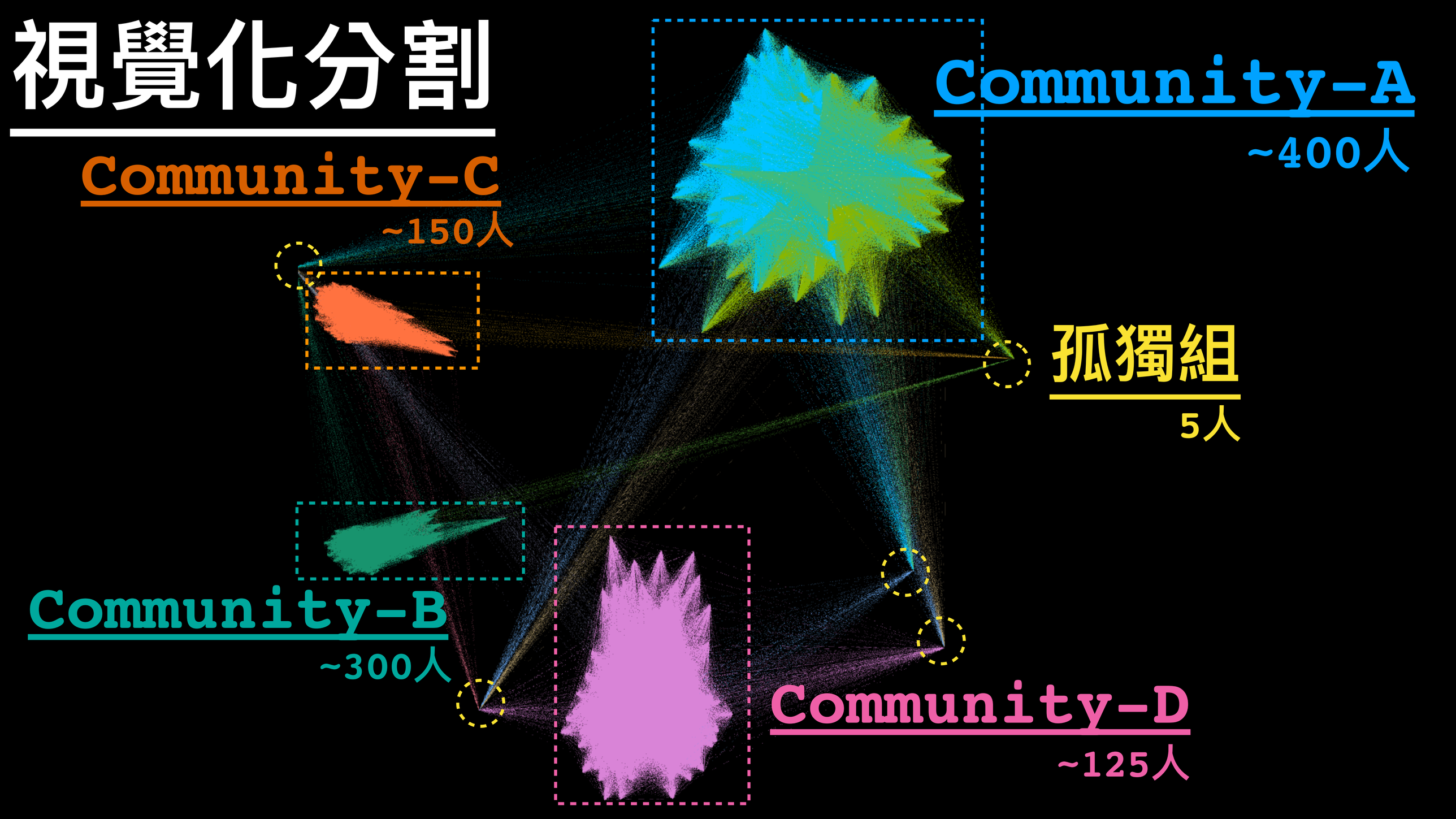
← 將 graph 寫入檔案

Gephi教學

Visualisation Partition
Code Partition



視覺化分割


Community-A
~400人
Community-C
~150人
孤獨組
5人
Community-B
~300人
Community-D
~125人

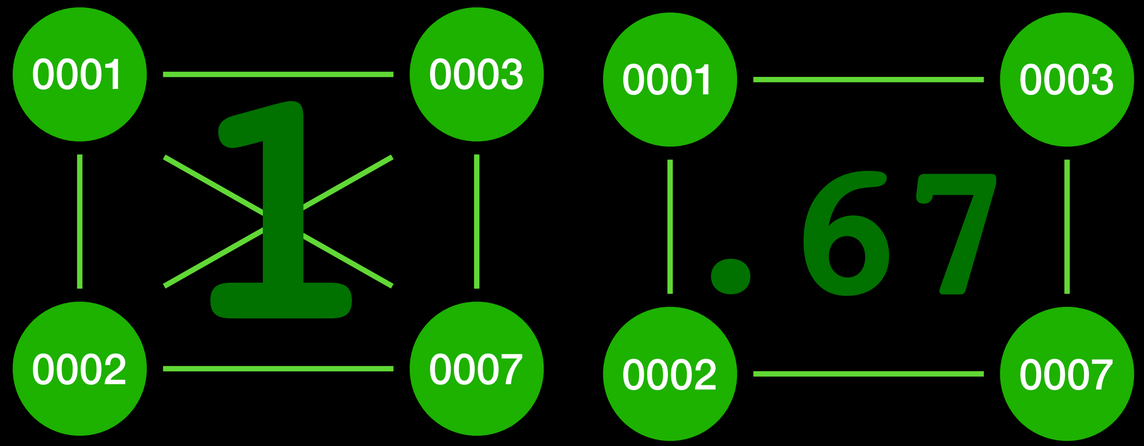
簡單指標

Betweenness Centrality

若節點為大家的"連結點"，則此值會高

Graph Density

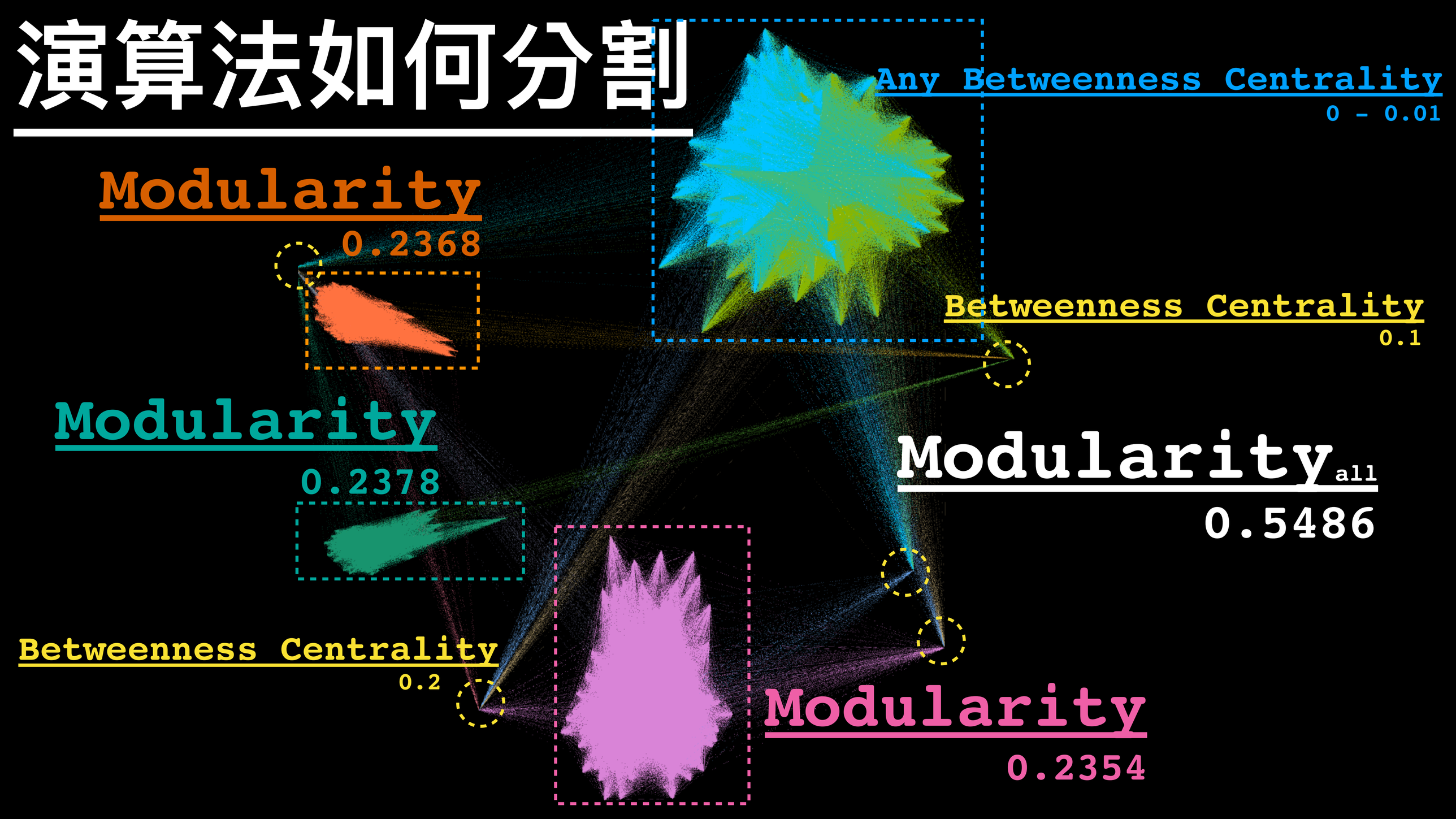
圖形密度



Modularity Index

此值會與圖形密度值成反比

演算法如何分割


Any Betweenness Centrality
0 - 0.01
Modularity
0.2368
Betweenness Centrality
0.1
Modularity
0.2378
Modularity all
0.5486
Betweenness Centrality
0.2
Modularity
0.2354

程式分割

引用 community detection 函式

```
import community

partitions = community.best_partition(g)
```

Community Detection

```
communities = {}
for k, v in partitions.items():
    communities.setdefault(v, [])
    communities[v].append(k)
```

people_name	community
people_0029	0
people_0028	0
people_0782	1
people_0783	1
people_0784	1
people_0785	1
people_0786	1
people_0787	1
people_0788	1

…

…

足夠了嗎？

Recursive Partitions

動手實作

```
There are 1000 people in this community(0), and modularity is 0.548649, split 5 communities
  There are 201 people in this community(0-0), and modularity is 0.001306
  There are 201 people in this community(0-1), and modularity is 0.001730
  There are 307 people in this community(0-2), and modularity is 0.237815, split 2 communities
    There are 154 people in this community(0-2-0), and modularity is 0.000968
    There are 153 people in this community(0-2-1), and modularity is 0.001528
  There are 151 people in this community(0-3), and modularity is 0.236836, split 2 communities
    There are 76 people in this community(0-3-0), and modularity is 0.000813
    There are 75 people in this community(0-3-1), and modularity is 0.000763
  There are 140 people in this community(0-4), and modularity is 0.235455, split 2 communities
    There are 70 people in this community(0-4-0), and modularity is 0.001771
    There are 70 people in this community(0-4-1), and modularity is 0.001321
```

這可以幫我們什麼？

Find the Association Rules

哪幾些事件總是一起發生？

PERSON-00001 QZTLUG IBBVVB RYQXHN XOQWYP PUXKTM SHUUIG ZKXXUW PSIQUL SPIJKG NMZVKU ORZILO ZIZTQJ JLGIEV GCOADL AXWEFS XFTXAY GRIRGI QAXKKF RGAIXP EXPXXC RPJXID HOJDUG CQBIKN ISVRDL GURTKN WAUYYY ZDTXXV DPEHCH XTUKGE NFBCKL IDIYUA YVVSJB YQQWJQ IZRNLZ WUUVJT UEGXNA JLLOAV CBVRMT

PERSON-00002 ORZILO EXPXXC YQQWJQ PXICYF JLGIEV YVVSJB WXQNYJ ZDTXXV XIAWTC PUXKTM UEGXNA ISVRDL JLLOAV ZKXXUW XHMQSN NMZVKU CAWELO

PERSON-00003 QZTLUG IBBVVB RYQXHN XOQWYP PUXKTM SHUUIG ZKXXUW PSIQUL DKKKBO AXWEFS NMZVKU ZIZTQJ YYWUXK JLGIEV GCOADL XFTXAY GRIRGI RPJXID CQBIKN GURTKN WAUYYY CAWELO NFBCKL IDIYUA YVVSJB YQQWJQ WUUVJT UEGXNA JLLOAV LAHAID

PERSON-00004 VGVFJB HRUFZC SCXDFP PUXKTM RYGFNG IZDAMM UQTPZQ NPOUCE BMELHY WXQNYJ WECGJK GRIRGI PXICYF AIZJYZ WBUASH ZCCLQU HOJDUG ZDTXXV ZQDVHK WAUYYY DPEHCH XHMQSN XTUKGE IQDWDJ XICHWB IZRNLZ BLIUMA UCVLBC GNLTMY JLLOAV GKMOKC

PERSON-00005 HRUFZC IBBVVB RYQXHN XOQWYP PUXKTM SHUUIG ZKXXUW DKKKBO SPIJKG NMZVKU NPOUCE ORZILO CQBIKN YYWUXK BMELHY AXWEFS XFTXAY GRIRGI QAXKKF RGAIXP YXTPUM EXPXXC RPJXID HOJDUG ZIZTQJ ZDTXXV GURTKN ISVRDL NFBCKL YVVSJB YQQWJQ LAHAID JLGIEV JLLOAV CBVRMT

…

…

…

若是客戶購買紀錄

PERSON-00001 波卡 綠色乖乖 金牛角 玉蜀黍 華元鹽酥餅...
PERSON-00002 科學麵 王子麵 辛拉麵 關廟麵 關公麵 ...
...
...

1. 推薦系統的祖先，關聯規則的找尋
2. Recommendation Algorithms
   - Content-Based
   - Collaboration Filtering
   - ALS

若是客戶基本資料

PERSON-00001 男性 40歲 有基金理財 有子女 大學學歷 會使用ATM 有聯名卡 是VIP ...
PERSON-00002 女性 21歲 沒有基金理財 無子女 研究所學歷 ...
...
...

關聯規則尋找

https://stanthecoder.me/apriori-bi-ji/

Items（最基本 itemset）

項目
Bread
Coke
Milk
Beer
Diaper
Eggs

Transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Frequent Item

項目	Support
Bread, Milk, Diaper	2

Frequent Rule

('Bread', 'Milk')

('Diaper',) 0.667

Apriori

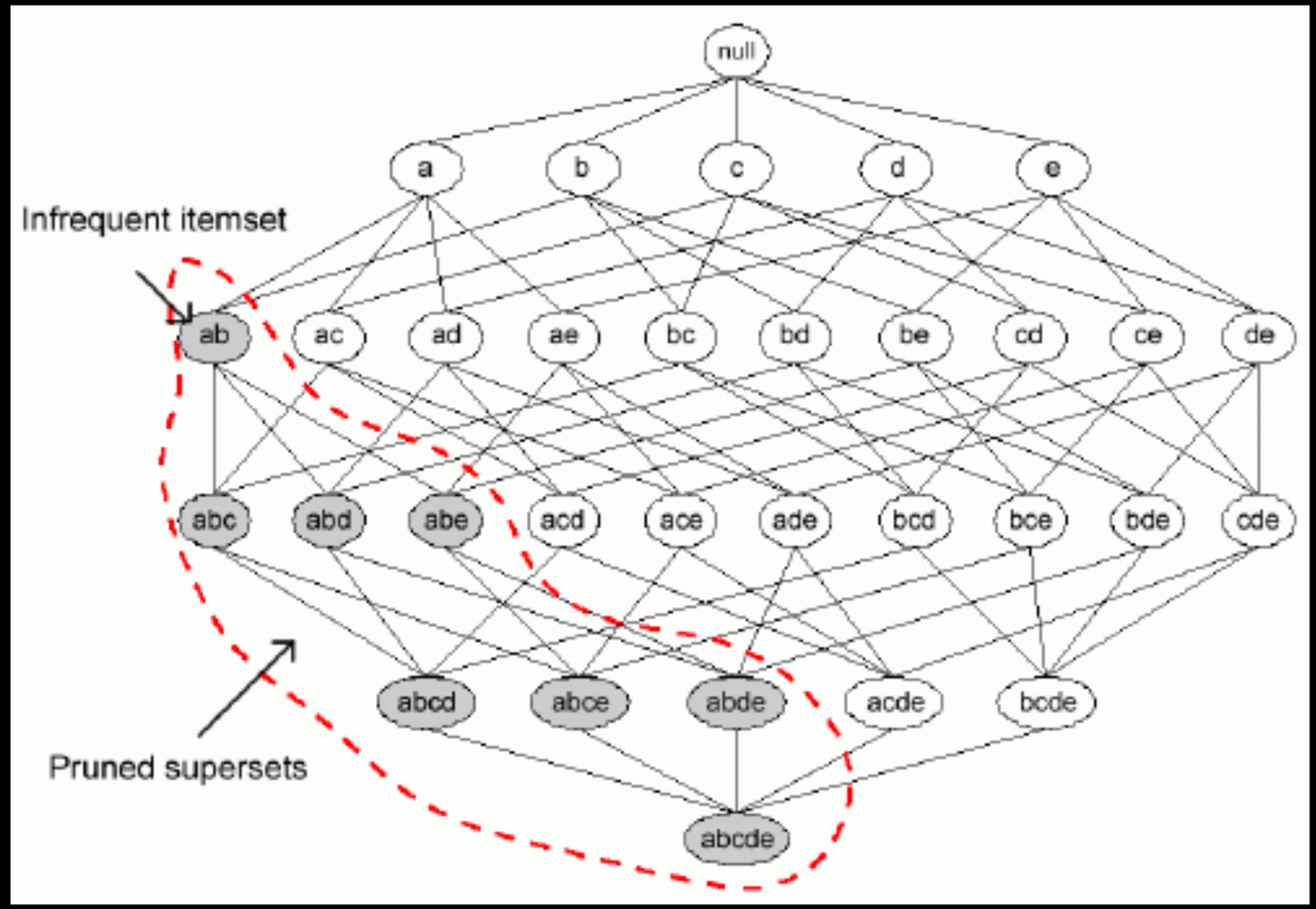
Find the Association Rule

原則一

如果一個 itemset 是高頻（高頻率出現）的，他的 subset 也一定會是高頻的。例如 ab 是高頻的，則 a b 也都會是高頻的

原則二

如果一個 itemset 不是高頻的，則他的 superset 也都不會是高頻的。例如 ab 不會是高頻的，則 abc, abd, abe 也都不會是高頻的



```
result = []
k = 1

# 篩選出第一階段的 frequent itemsets
frequent_itemsets = itemsets.filter { |itemset| itemset.support < min_support }

# 執行以下程序直到沒有任何新的 frequent itemsets
while frequent_itemsets.size > 0 do
  k = k + 1
  # 從 frequent itemsets 裡面產生長度是 k + 1 的 candidates
  candidates = generate_candidates(frequent_itemsets, length: k)

  # 用所有的 transaction 跑 for each
  Transaction.all.each do |transaction|
    # 跟每一個 itemset 出現在其中的 candidates
    candidates_of_transaction = get_subset(candidates, transaction)
    # 出現在此 transaction 的 candidates support 都加一
    candidates_of_transaction.each do |candidate|
      candidate.increase_support_count
    end
  end

  # 最後用計算完 support 的 candidates 再做一次篩選
  frequent_itemsets = candidates.filter { |itemset| itemset.support < min_support }

  把這一次篩選後的結果存進 result
  result << frequent_itemsets
end
```

Pseudo Code

還有甚麼不足的？

Dynamic Graph