ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ

Môn học : Dữ liệu lớn

Học kỳ I (2022-2023)

ĐỀ TÀI:

ỨNG DỤNG KỸ THUẬT HỌC MÁY XỬ LÝ BÀI TOÁN TRÊN TẬP DỮ LIỆU NHỮNG CUỘC KHỦNG BỐ Ở MỸ

Sinh viên thực hiện:

Trần Nhật Tân MSSV: 19522177

Bùi Ngọc Thành MSSV: 19522220

Huỳnh Quốc Khánh MSSV: 19521677

Lê Thế Tiệm MSSV: 19522330

GVHD: ThS. Nguyễn Hồ Duy Tri

Lớp: IS405.N11.HTCL

Thành phố Hồ Chí Minh, tháng 12 năm 2022

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ

Môn học : Dữ liệu lớn

Học kỳ I (2022-2023)

ĐỀ TÀI:

ỨNG DỤNG KỸ THUẬT HỌC MÁY XỬ LÝ BÀI TOÁN TRÊN TẬP DỮ LIỆU NHỮNG CUỘC KHỦNG BỐ Ở MỸ

Sinh viên thực hiện:

Trần Nhật Tân	MSSV: 19522177
Bùi Ngọc Thành	MSSV: 19522220
Huỳnh Quốc Khánh	MSSV: 19521677
Lê Thế Tiệm	MSSV: 19522330

GVHD: ThS. Nguyễn Hồ Duy Tri

Lớp: IS405.N11.HTCL

Thành phố Hồ Chí Minh, tháng 12 năm 2022

MỤC LỤC:

LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Với lòng biết ơn sâu sắc nhất, đầu tiên em xin gửi lời cảm ơn chân thành đến tập thể quý Thầy Cô Trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM và quý Thầy Cô khoa Hệ thống thông tin đã giúp cho nhóm có những kiến thức cơ bản làm nền tảng để thực hiện đề tài này.

Đặc biệt nhóm chúng em xin gửi lời cảm ơn chân thành tới thầy Nguyễn Hồ Duy Tri – giảng viên lý thuyết và thực hành môn Dữ liệu lớn đã tận tình giúp đỡ, trực tiếp chỉ bảo, hướng dẫn em trong suốt quá trình làm đề tài. Nhờ đó, chúng em đã tiếp thu được nhiều kiến thức bổ ích trong việc vận dụng cũng như kỹ năng làm đề tài. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì em nghĩ đề tài này của nhóm rất khó có thể hoàn thiện được. Một lần nữa, em xin chân thành cảm ơn thầy.

Cuối cùng, các thành viên trong nhóm đã làm việc hết công suất để hoàn thành tốt đề tài của mình. Xin chân thành cảm ơn!

NHẬN XÉT CỦA GIẢNG VIÊN

I. TỔNG QUAN ĐỀ TÀI

1.1 Lý do chọn đề tài

Khủng bố là hoạt động phá hoại, đe dọa bằng lời nói, hình ảnh hoặc video giết người do cá nhân hoặc tổ chức thực hiện làm thiệt mạng người, đặc biệt là thường dân, hoặc gây tổn thất cho xã hội và cộng đồng để tác động vào tâm lý đối phương gây hoang mang khiếp sợ, nhằm mục đích chính trị hoặc tôn giáo. Một đặc tính thống nhất của khủng bố là việc sử dụng bừa bãi bạo lực đối với những người không có khả năng chống cự với mục đích là sự nổi tiếng cho một nhóm, một phong trào, một cá nhân hoặc gây áp lực lên đối thủ chính trị buộc họ phải chấp nhận một giải pháp chính trị có lợi cho mình. Các tổ chức khủng bố có thể khai thác nỗi sợ hãi của con người để hỗ trợ đạt được những mục tiêu này. Đối tượng bị khủng bố gây thiệt hại có thể là tính mạng, sức khỏe, danh dự nhân phẩm, tài sản (của cá nhân, tổ chức hay của nhà nước) hoặc sự vững mạnh của một chính quyền nhà nước.

Khủng bố là mối quan tâm lớn đối với các quốc gia trên toàn thế giới, vì nó thường báo hiệu mở đầu cho chiến tranh và dân thường bị mắc kẹt trong cuộc chiến. Chúng ta không thể thay đổi thực tế mà chúng ta không hiểu đầy đủ, vì vậy đây là một số dữ liệu liên quan về chủ đề này Nhận thấy tầm quan trọng ấy, nhóm tập trung nghiên cứu ứng dụng giải thuật K-Means để phân cụm những cuộc tấn công ở các thành phố trong quốc gia United States từ đó đề ra những chiến lược phù hợp giúp chống lại những cuộc khủng bố, góp phần mang lại an toàn cho người dân và xã hội.

1.2 Mô tả dữ liệu

1.2.1 Nguồn dữ liệu

Tên dữ liệu: Terrorism Data 1970to2017

Nguồn cung cấp: Kaggle.com

Link dữ liệu: Terrorism Data 1970to2017 | Kaggle

Mô tả nguồn dữ liệu:

Đây là một cơ sở dữ liệu nguồn mở bao gồm thông tin về các cuộc tấn công khủng bố trên toàn thế giới từ năm 1970 đến năm 2017. Bộ dữ liệu này bao gồm

dữ liệu có hệ thống về các vụ khủng bố trong nước cũng như quốc tế đã xảy ra trong khoảng thời gian này.

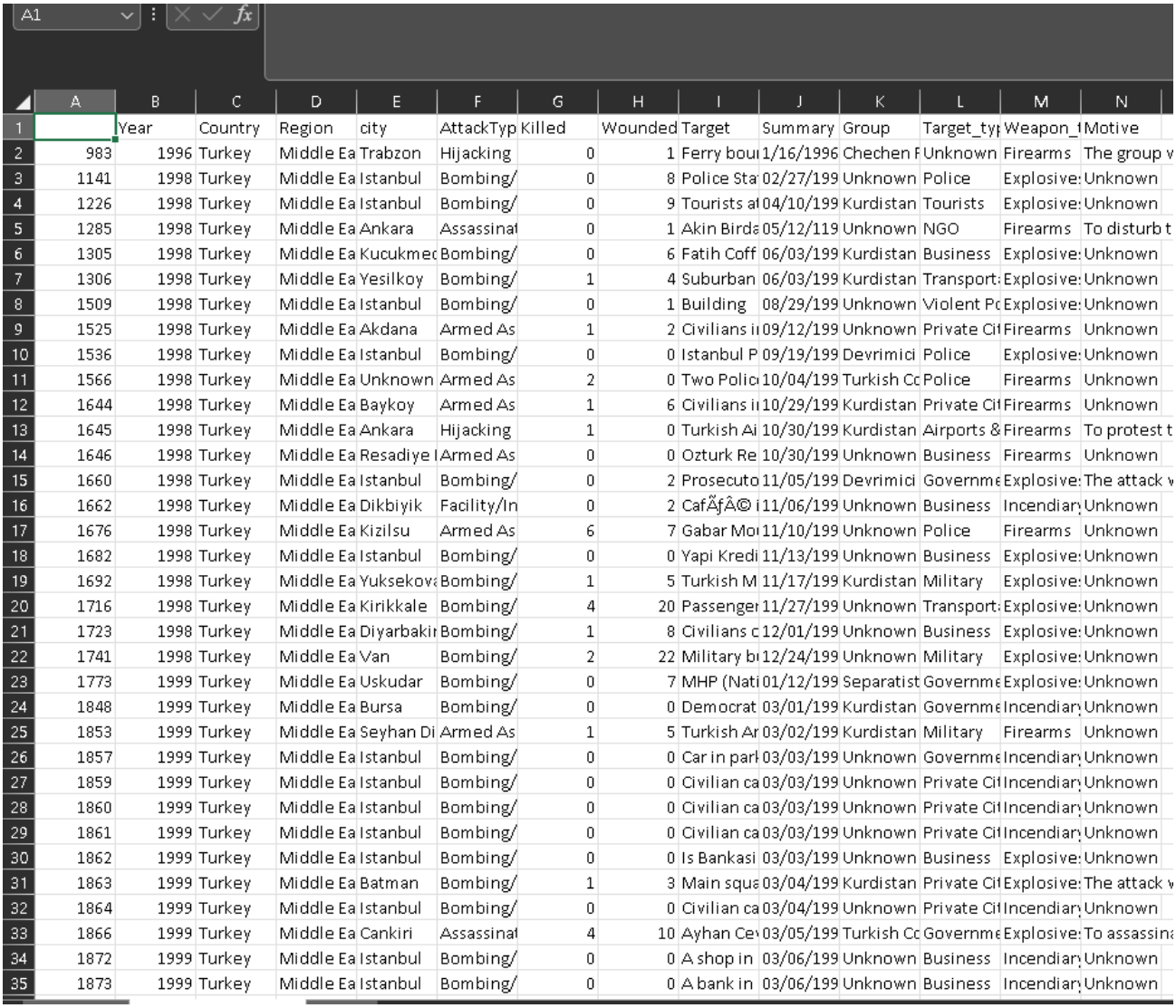
1.2.2 Các thuộc tính của tập dữ liệu

- Mỗi dòng trong tập dữ liệu là một vụ khủng bố tại Thổ Nhĩ Kỳ.
- Dữ liệu gồm 547 dòng và 17 thuộc tính.

STT	Tên cột	Kiểu dữ liệu	Ý nghĩa
1	Year	Int	Năm xảy ra vụ khủng bố
2	Month	Int	Tháng xảy ra vụ khủng bố
3	Day	Int	Ngày xảy ra vụ khủng bố
4	Country	String	Quốc gia
5	State	String	Bang của quốc gia
6	Region	String	Vùng của quốc gia
7	City	String	Thành phố xảy ra vụ khủng bố
8	Latitude	Double	Vĩ độ xảy ra cuộc khủng bố
9	Longtitude	Double	Kinh độ xảy ra cuộc khủng bố
10	AttackType	String	Kiểu tấn công
11	Killed	Int	Số người tử vong
12	Wounded	Int	Số người bị thương
13	Target	String	Mục tiêu của cuộc tấn công khủng bố
14	Summary	String	Diễn biến chi tiết của cuộc khủng bố

15	Group	String	Tổ chức khủng bố
16	Target_type	String	Loại mục tiêu của cuộc tấn công khủng bố
17	Weapon_type	String	Loại vũ khí dùng để khủng bố

- Bảng dữ liệu:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1		Year	Country	Region	city	AttackTyp	Killed	Wounded	Target	Summary	Group	Target_ty	Weapon_	Motive
2	983	1996	Turkey	Middle Ea	Trabzon	Hijacking	0	1	Ferry bou	1/16/1996	Chechen F	Unknown	Firearms	The group v
3	1141	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	8	Police Sta	02/27/199	Unknown	Police	Explosive:	Unknown
4	1226	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	9	Tourists at	04/10/199	Kurdistan	Tourists	Explosive:	Unknown
5	1285	1998	Turkey	Middle Ea	Ankara	Assassinat	0	1	Akin Birda	05/12/119	Unknown	NGO	Firearms	To disturb t
6	1305	1998	Turkey	Middle Ea	Kucukmec	Bombing/	0	6	Fatih Coff	06/03/199	Kurdistan	Business	Explosive:	Unknown
7	1306	1998	Turkey	Middle Ea	Yesilkoy	Bombing/	1	4	Suburban	06/03/199	Kurdistan	Transport:	Explosive:	Unknown
8	1509	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	1	Building	08/29/199	Unknown	Violent Pc	Explosive:	Unknown
9	1525	1998	Turkey	Middle Ea	Akdana	Armed As	1	2	Civilians i	09/12/199	Unknown	Private Cit	Firearms	Unknown
10	1536	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Istanbul P	09/19/199	Devrimici	Police	Explosive:	Unknown
11	1566	1998	Turkey	Middle Ea	Unknown	Armed As	2	0	Two Polic	10/04/199	Turkish Co	Police	Firearms	Unknown
12	1644	1998	Turkey	Middle Ea	Baykoy	Armed As	1	6	Civilians i	10/29/199	Kurdistan	Private Cit	Firearms	Unknown
13	1645	1998	Turkey	Middle Ea	Ankara	Hijacking	1	0	Turkish Ai	10/30/199	Kurdistan	Airports &	Firearms	To protest t
14	1646	1998	Turkey	Middle Ea	Resadiye I	Armed As	0	0	Ozturk Re	10/30/199	Unknown	Business	Firearms	Unknown
15	1660	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	2	Prosecuto	11/05/199	Devrimici	Governme	Explosive:	The attack v
16	1662	1998	Turkey	Middle Ea	Dikbiyik	Facility/In	0	2	CafÃƒÂ© i	11/06/199	Unknown	Business	Incendiary	Unknown
17	1676	1998	Turkey	Middle Ea	Kizilsu	Armed As	6	7	Gabar Mo	11/10/199	Unknown	Police	Firearms	Unknown
18	1682	1998	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Yapi Kredi	11/13/199	Unknown	Business	Explosive:	Unknown
19	1692	1998	Turkey	Middle Ea	Yuksekova	Bombing/	1	5	Turkish M	11/17/199	Kurdistan	Military	Explosive:	Unknown
20	1716	1998	Turkey	Middle Ea	Kirikkale	Bombing/	4	20	Passenger	11/27/199	Unknown	Transport:	Explosive:	Unknown
21	1723	1998	Turkey	Middle Ea	Diyarbakir	Bombing/	1	8	Civilians c	12/01/199	Unknown	Business	Explosive:	Unknown
22	1741	1998	Turkey	Middle Ea	Van	Bombing/	2	22	Military bi	12/24/199	Unknown	Military	Explosive:	Unknown
23	1773	1999	Turkey	Middle Ea	Uskudar	Bombing/	0	7	MHP (Nati	01/12/199	Separatist	Governme	Explosive:	Unknown
24	1848	1999	Turkey	Middle Ea	Bursa	Bombing/	0	0	Democrat	03/01/199	Kurdistan	Governme	Incendiary	Unknown
25	1853	1999	Turkey	Middle Ea	Seyhan Di	Armed As	1	5	Turkish Ar	03/02/199	Kurdistan	Military	Firearms	Unknown
26	1857	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Car in parl	03/03/199	Unknown	Governme	Incendiary	Unknown
27	1859	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Civilian ca	03/03/199	Unknown	Private Cit	Incendiary	Unknown
28	1860	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Civilian ca	03/03/199	Unknown	Private Cit	Incendiary	Unknown
29	1861	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Civilian ca	03/03/199	Unknown	Private Cit	Incendiary	Unknown
30	1862	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Is Bankasi	03/03/199	Unknown	Business	Explosive:	Unknown
31	1863	1999	Turkey	Middle Ea	Batman	Bombing/	1	3	Main squa	03/04/199	Kurdistan	Private Cit	Explosive:	The attack v
32	1864	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	Civilian ca	03/04/199	Unknown	Private Cit	Incendiary	Unknown
33	1866	1999	Turkey	Middle Ea	Cankiri	Assassinat	4	10	Ayhan Ce	03/05/199	Turkish Co	Governme	Explosive:	To assassina
34	1872	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	A shop in	03/06/199	Unknown	Business	Incendiary	Unknown
35	1873	1999	Turkey	Middle Ea	Istanbul	Bombing/	0	0	A bank in	03/06/199	Unknown	Business	Incendiary	Unknown

1.3 Mô tả bài toán

Mục tiêu của bài toán này là xây dựng một mô hình học máy để phân cụm cuộc khủng bố tại Mỹ để từ đó đưa ra các chiến lược phù hợp nhằm giúp chống lại những cuộc khủng bố, góp phần mang lại an toàn cho người dân và xã hội.

Đây là project học máy kĩ thuật gom cụm với học không giám sát.

1.4 Kỹ thuật tiền xử lý dữ liệu được lựa chọn

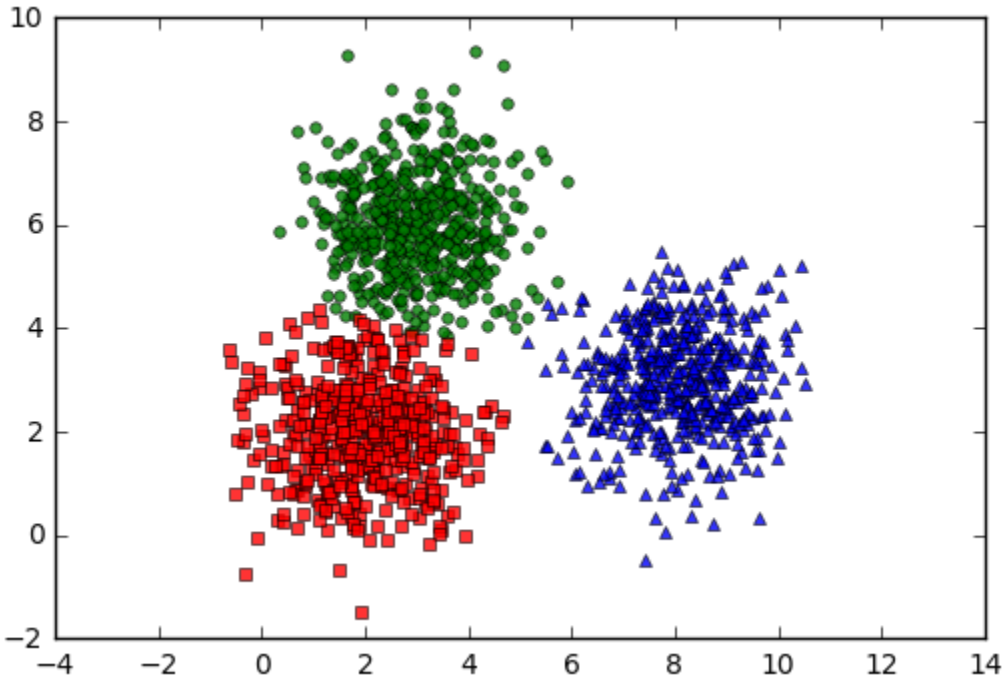
- Tìm cột không cần thiết.
- Tìm cột thiếu dữ liệu (missing values) .
- Tìm những cột chỉ có 1 giá trị.
- Chuyển các thuộc tính Object thành Numeric.
- Mô tả các thành phần của dữ liệu (count,min/max, mean, std)

1.5 Thuật toán khai thác dữ liệu được lựa chọn

1.5.1 Thuật toán K-Means

Thuật toán phân cụm K-means là một thuật toán phân cụm đơn giản thuộc loại học không giám sát (tức là dữ liệu không có nhãn) và được sử dụng để giải quyết bài toán phân cụm. Và là một phương pháp được sử dụng trong phân tích tính chất cụm của dữ liệu. Nó đặc biệt được sử dụng nhiều trong khai phá dữ liệu và thống kê. K-Means phân vùng dữ liệu thành k cụm khác nhau. Giải thuật này giúp xác định được dữ liệu thuộc về cụm (Cluster) nào.

Trong đó số lượng cụm được cho trước là k. Công việc phân cụm được xác lập dựa trên nguyên lý: Các điểm dữ liệu trong cùng 1 cụm thì phải có cùng 1 số tính chất nhất định. Tức là giữa các điểm trong cùng 1 cụm phải có sự liên quan lẫn nhau. Đối với máy tính thì các điểm trong 1 cụm đó sẽ là các điểm dữ liệu gần nhau.



Hình 1. Kết quả phân cụm bằng thuật toán k-means

- Ý tưởng của thuật toán :
  - Bước 1: Chọn ngẫu nhiên K tâm (centroid) cho K cụm (cluster). Mỗi cụm được đại diện bằng các tâm của cụm.
  - Bước 2: Tính khoảng cách giữa các đối tượng (objects) đến K tâm (thường dùng khoảng cách Euclidean).
  - Bước 3: Nhóm các đối tượng vào nhóm gần nhất.
  - Bước 4: Xác định lại tâm mới cho các nhóm.
  - Bước 5: Thực hiện lại bước 2 cho đến khi không có sự thay đổi nhóm nào của các đối tượng.
- Ưu điểm :
  - Dễ dàng thực hiện.
  - Chia dữ liệu thành những tập lớn và đảm bảo sự hội tụ của các phần tử

trong cụm.

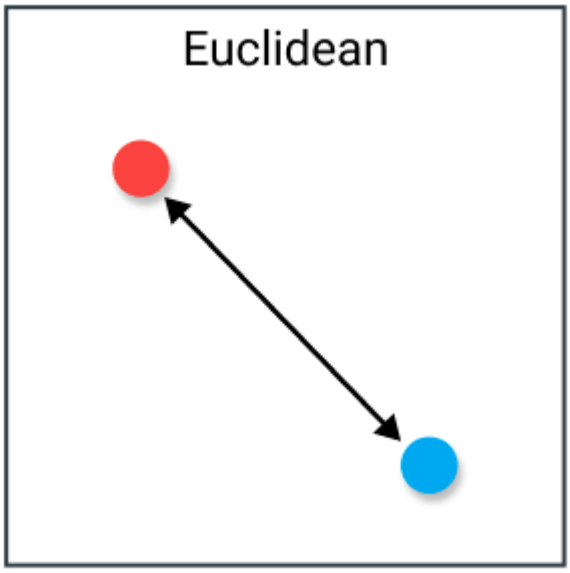
- Với một số lượng lớn các biến, K-Means có thể nhanh hơn về mặt tính toán so với phân cụm phân cấp (nếu K nhỏ).
- K-means có thể tạo ra nhiều cụm hơn so với phân cụm phân cấp.
- K-Means tạo ra các cụm chặt chẽ hơn so với phân cụm phân cấp, đặc biệt nếu các cụm là hình cầu.

- Nhược điểm :

  - Người dùng phải chỉ định k (số lượng cụm) khi thực hiện.
  - k-means chỉ có thể xử lý dữ liệu số.
  - Bị phụ thuộc vào các giá trị ban đầu.
  - Khi xử lý các cụm lớn, nó có thể sẽ không hoạt động tốt.
  - Các phân vùng ban đầu khác nhau có thể dẫn đến các cụm cuối cùng khác nhau.
  - Không hoạt động tốt với các cụm (trong dữ liệu gốc) có kích thước khác nhau và mật độ khác nhau.

1.6 Khoảng cách Euclidean

Khoảng cách Euclid giữa hai điểm trong không gian Euclid là độ dài của đoạn thẳng nối hai điểm đó. Là thước đo khoảng cách tốt nhất có thể được giải thích là độ dài của một đoạn nối hai điểm.



Hình 2. Hình ảnh khoảng cách Euclidean

- Công thức :

$$D(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2)}$$

- Trường hợp sử dụng :

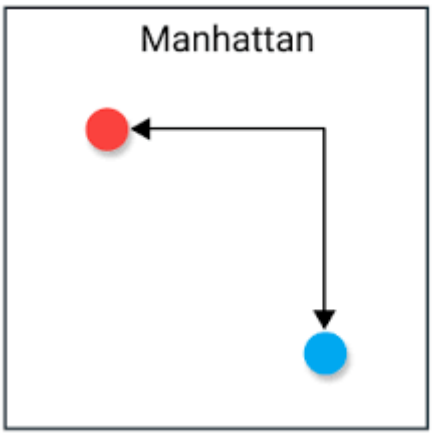
  - Khoảng cách Euclid hoạt động hiệu quả khi bạn có dữ liệu chiều thấp và độ lớn của vector là điều quan trọng cần được đo.
  - Các thuật toán như K-nn và HDBSCAN cho kết quả tốt nếu sử dụng khoảng cách Euclid trên dữ liệu chiều thấp.
  - Mặc dù nhiều biện pháp khác đã được phát triển để giải quyết những nhược điểm của khoảng cách Euclide, nó vẫn là một trong những thước đo khoảng cách được sử dụng nhiều nhất vì những lý do chính đáng. [4]
  - Nó cực kỳ trực quan để sử dụng, đơn giản để thực hiện và cho thấy kết quả

tuyệt vời trong nhiều trường hợp sử dụng.

- Ưu điểm :
  - Được sử dụng phổ biến, dễ hiểu, dễ thực hiện.
  - Cho kết quả tốt trong nhiều usecase.
  - Đặc biệt hiệu quả trên các tập dữ liệu ít chiều.
- Nhược điểm :
  - Mặc dù nó là một thước đo khoảng cách phổ biến, khoảng cách Euclide không phải là biến thể theo tỷ lệ, có nghĩa là khoảng cách được tính toán có thể bị sai lệch tùy thuộc vào đơn vị của các đối tượng địa lý.
  - Cần phải chuẩn hóa dữ liệu trước khi sử dụng thước đo khoảng cách này.
  - Euclide distance có thể bị ảnh hưởng bởi đơn vị của feature. Chính vì vậy cần phải thực hiện normalize trước khi tính toán.
  - Khi số chiều vector space tăng lên, Euclide Distance trở nên kém hiệu quả. Một phần nguyên nhân do dữ liệu thực tế thường không chỉ nằm trong Euclide Metric Space.

1.7 Khoảng cách Manhattan

- Khoảng cách Manhattan
  - Khoảng cách Manhattan tính toán sự khác biệt tuyệt đối giữa các tọa độ của các cặp đối tượng. Nó còn được gọi là khoảng cách L1 (hay Taxicab/City Block distance).



Hình 3 Khoảng cách Manhattan

- Công thức:

$$D(x,y) = \sum_{i=1}^{k} |x_i - y_i|$$

- Trường hợp sử dụng :
  - Khi tập dữ liệu của bạn có các thuộc tính rời rạc và / hoặc nhị phân, Manhattan dường như hoạt động khá tốt vì nó tính đến các đường dẫn mà thực tế có thể được thực hiện trong các giá trị của các thuộc tính đó.
- Ưu điểm :
  - Trong một số trường hợp sẽ hiệu quả khi thực hiện trên các tập dữ liệu nhiều chiều.
- Nhược điểm :
  - Mặc dù khoảng cách Manhattan có vẻ phù hợp với dữ liệu nhiều chiều , nhưng nó là một phép đo có phần kém trực quan hơn so với khoảng cách euclide, đặc biệt là khi sử dụng trong dữ liệu nhiều chiều.
  - Hơn nữa, nó có nhiều khả năng cho một giá trị khoảng cách cao hơn khoảng cách euclide vì nó không phải là đường đi ngắn nhất có thể. Điều này không nhất thiết đưa ra các vấn đề nhưng là điều người dùng nên tính đến.
- Cost :

- Hàm cost dùng để tính tổng bình phương khoảng cách từ các điểm đến tâm tương ứng của chúng. Hàm này được dùng với mục đích tối thiểu hoá chi phí tính toán vì bước này liên quan đến việc gán một điểm dữ liệu cho cụm gần nhất có thể.
- Công thức:

$$J(c^{(1)}, c^{(2)}, \cdots, c^{(m)}, \mu_1, \mu_2, \cdots, \mu_K) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

Trong đó:

- c(i) là số thứ tự của cụm.
- μk là toạ độ tâm cụm K.
- $\mu_c^{(i)}$ là toạ độ tâm cụm của mẫu x

1.8 K-Means Mapreduce

- Map:
  - Cho một điểm và tập hợp các tâm cụm.
  - Tính khoảng cách từ điểm đó đến từng tâm cụm.s
  - Phát ra điểm đó và tâm cụm gần nó nhất
- Reduce:
  - Cho tâm cụm và các điểm thuộc cụm đó.
  - Tính tâm cụm mới cho từng cụm bằng cách lấy trung bình cộng của tất các các điểm dữ liệu đã được gán vào cluster đó.
  - Phát ra tâm cụm mới.

II. TIỀN XỬ LÝ DỮ LIỆU

2.1 Các thư viện được sử dụng

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import networkx as nx
import ndlib.models.ModelConfig as mc
import ndlib.models.epidemics as ep


from networkx.algorithms import bipartite
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.cm as cm
import matplotlib
from community import community_louvain
import seaborn as sns
from termcolor import colored
from networkx.algorithms.community import label_propagation_communities
from sklearn import preprocessing
from sklearn.cluster import KMeans
```

Hình 4. import các thư viện

2.2 Đọc dữ liệu đầu vào

Sử dụng lệnh spark.read.csv để đọc file dữ liệu "terrorism-in-US.csv".

```
# đọc dữ liệu vào dataframe
spark = SparkSession.builder.appName('ml-terro').getOrCreate()
dataframe = spark.read.csv("khungbo/terrorism-in-US.csv", header = True, inferSchema = True)
dataframe.printSchema()
```

Hình 5. đọc file csv

Sử dụng hàm describe.show để quan sát thống kê của bảng dữ liệu.

```
# Xem thống kê tập dữ liệu
dataframe.describe().show()
```

```
+-------+------------------+-----------------+------------------+-------------+------+-----------+--------+------------------+------------------+-----------+------------------+------------------+
|summary|              Year|            Month|               Day|      Country| State|     Region|    City|          Latitude|         Longitude| AttackType|            Killed|           Wounded|
+-------+------------------+-----------------+------------------+-------------+------+-----------+--------+------------------+------------------+-----------+------------------+------------------+
|  count|              2836|             2836|              2836|         2836|  2832|       2836|    2836|              2835|              2835|       2836|              2763|              2743|
|   mean| 1983.984837799718|6.2224964739069115|15.316995768688294|         null|  null|       null|    null|36.675561514991195|-91.81962188606785|       null|1.3648208469055374|7.547211082756107|
| stddev|14.193825172407253|3.3924183923107827|  9.144808394152761|         null|  null|       null|    null| 7.355989143847745|21.768840099791674|       null| 37.56433178774492|222.57220829483366|
|    min|              1970|                1|                 0|United States|Alabama|North America|   Afton|         17.966072|       -157.858333|Armed Assault|                 0|                 0|
|    max|              2017|               12|                31|United States|Wyoming|North America|Yuba City|         64.837778|       105.270546|    Unknown|              1384|              8191|
+-------+------------------+-----------------+------------------+-------------+------+-----------+--------+------------------+------------------+-----------+------------------+------------------+
```

Hình 6. kết quả dataframe của dữ liệu

2.3 Xóa các cột bị thiếu hoặc dư thừa

- Sử dụng hàm drop để loại bỏ các cột dữ liệu dư thừa
  - Cột Latitude và Longtitude làm mã định vị của cuộc khủng bố, không mang ý nghĩa trong tập dữ liệu.
  - Cột Year, Month, Day mang ý nghĩa ngày, tháng, năm diễn ra cuộc khủng bố, không được chọn để phân tích nên có thể xóa.
  - Cột Target, Summary, Group, Motive là những cuộc có kiểu dữl liệu string mô tả, diễn giải cuộc tấn công, không phù hợp để phân tích.

```
#Chúng ta có thể thấy rằng có nhiều cột dư thừa trong tập dữ liệu. Chúng ta nên bỏ chúng đi trước khi tiếp tục.
#Bỏ các cột thừa
df= dataframe.drop('Target','Year','Month','Day','Latitude','Longitude','State','Summary','Group','Motive','Weapon_type','AttachType')
```

Hình 7. Loại bỏ dữ liệu trống

- Tìm **cột có dữ liệu thiếu**
  - Thiếu dữ liệu là một sự xuất hiện phổ biến và có ảnh hưởng đáng kể đến các kết luận được rút ra từ dữ liệu. Để có thể kết luận chính xác hơn thì loại bỏ các dòng dữ liệu là cần thiết.

```
[ ] from pyspark.sql.functions import isnan, isnull, when, count, col
    df.select([count(when(isnan(c), c)).alias(c) for c in df.columns]).show()
    #print(np.round(df.select([count(when(isnull(c), c)).alias(c) for c in df.columns]).mean(), 4),  ' % missing values')
```

```
+-------+------+----+----------+------+-------+-----------+
|Country|Region|City|AttackType|Killed|Wounded|Target_type|
+-------+------+----+----------+------+-------+-----------+
|      0|     0|   0|         0|     0|      0|          0|
+-------+------+----+----------+------+-------+-----------+
```

Hình 8. Kiểm tra giá trị N/a của các cột dữ liệu

```
[ ] from pyspark.sql.functions import isnan, when, count, col
    df.select([count(when(isnull(c), c)).alias(c) for c in df.columns]).show()
    #print(np.round(df.select([count(when(isnull(c), c)).alias(c) for c in df.columns]).mean(), 4),  ' % missing values')
```

```
+-------+------+----+----------+------+-------+-----------+
|Country|Region|City|AttackType|Killed|Wounded|Target_type|
+-------+------+----+----------+------+-------+-----------+
|      0|     0|   0|         0|    73|     93|          0|
+-------+------+----+----------+------+-------+-----------+
```

Hình 9. Kiểm tra giá trị Null của các cột dữ liệu

⇨ Có thể thấy rằng cột dữ liệu Killed và Wouded là 2 cột bị thiếu dữ liệu.

- Tìm **cột chỉ có một giá trị**

```
[ ] df.select([F.countDistinct(c).alias(c) for c in df.columns]).show()
```

```
+-------+------+----+----------+------+-------+-----------+
|Country|Region|City|AttackType|Killed|Wounded|Target_type|
+-------+------+----+----------+------+-------+-----------+
|      1|     1| 727|         9|    22|     38|        126|
+-------+------+----+----------+------+-------+-----------+
```

Hình 10. tìm cột chỉ có một giá trị

⇨ Cột Country và Region đều chỉ có 1 giá trị.

- Tiến hành loại bỏ những dòng giá trị thiếu.

```
[ ] # Tiến hành xóa luôn các cột không chọn để phân tích
    columns_to_drop = ['Region', 'Country','City',]
    df= df.drop(*columns_to_drop)
```

```
[ ] df.toPandas().info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2836 entries, 0 to 2835
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   AttackType   2836 non-null   object
 1   Killed       2763 non-null   float64
 2   Wounded      2743 non-null   float64
 3   Target_type  2836 non-null   object
dtypes: float64(2), object(2)
memory usage: 88.8+ KB
```

Hình 11. Xóa các côt không chọn

2.4 Chuyển các thuộc tính Object thành Numeric

Vì dữ liệu của cột AttackType và Target_type là dạng chuỗi nên sẽ không phục vụ được cho việc Mining dữ liệu, nhóm sẽ chuyển đổi dữ liệu sang dạng số

```
[ ] from pyspark.ml.feature import StringIndexer

    #Encode categories attributes
    inputs = ["AttackType", "Target_type"]
    outputs = ["AttackType_n", "Target_type_n"]
    dfn = StringIndexer(inputCols=inputs, outputCols=outputs, stringOrderType='alphabetAsc')
    dfn = dfn.fit(df).transform(df)
```

Hình 11. Tiến hành chuyển đổi kiểu dữ liệu

```
[ ] #Calculate mean of balance column
    indexedTypedf = dfn.withColumn("KilledDouble", dfn["Killed"].cast("double"))
    indexedTypedf.groupBy().mean('KilledDouble').show()

    #Replace "N/A" value with 1362
    from pyspark.sql.functions import regexp_replace
    #dfn = dfn.withColumn('Wounded', regexp_replace('Wounded', 'NaN', '1.0'))
    #dfn = dfn.withColumn('Wounded', regexp_replace('Wounded', 'N/A', '1.0'))
    dfn = dfn.fillna({'Killed':'1'})
    dfn.show()
```

```
+------------------+
| avg(KilledDouble)|
+------------------+
|1.3648208469055374|
+------------------+

+------+-------+------------+-------------+
|Killed|Wounded|AttackType_n|Target_type_n|
+------+-------+------------+-------------+
|     0|      0|         0.0|        115.0|
|     0|      0|         2.0|        123.0|
|     0|      0|         3.0|        112.0|
|     0|      0|         3.0|        107.0|
|     0|      0|         2.0|        112.0|
|     0|      0|         3.0|        112.0|
|     0|      0|         3.0|        107.0|
|     0|      0|         3.0|        102.0|
|     0|      0|         2.0|        104.0|
|     0|      0|         2.0|        102.0|
|     0|      0|         3.0|        102.0|
|     0|      1|         3.0|        115.0|
|     0|      0|         2.0|        104.0|
```

Hình 12 kết quả sau khi chuyển đổi kiểu dữ liệu

2.5 Mô tả số thành phần của dữ liệu

Tiến hành mô tả các thành phần của dữ liệu như:

- Số lượng dòng dữ liệu(count)
- Giá trị cao nhất, thấp nhất(max,min)

- Giá trị trung bình, trung vị(mean)
- Độ lệch chuẩn(stddev)

- Sử dụng hàm describe.show để xem thông tin của tập dữ liệu sau khi xóa các cột.

```
# xem tóm tắt của tập dữ liệu
# Nhóm sẽ lựa chọn các col : num_reactions,num_comments,num_shares để tiến hành phân cụm
# Chuyển dữ liệu thành kiểu int để có thể xem describe chính xác hơn
dfn = dfn.withColumn("AttackType_n", col("AttackType_n").cast("int")) \
    .withColumn("Killed", col("Killed").cast("int")) \
    .withColumn("Wounded", col("Wounded").cast("int")) \
    .withColumn("Target_type_n", col("Target_type_n").cast("int"))
dfn.describe().show()
```

```
+-------+------------------+-----------------+------------------+------------------+
|summary|            Killed|          Wounded|      AttackType_n|     Target_type_n|
+-------+------------------+-----------------+------------------+------------------+
|  count|              2836|             2836|              2836|              2836|
|   mean|1.3554301833568405|7.529266572637518|2.2954866008462624|104.84026798307475|
| stddev|  37.0775902520346|218.8911341202021|1.2969074909306515|14.611592621631132|
|    min|                 0|                0|                 0|                 0|
|    max|              1384|             8191|                 8|               125|
+-------+------------------+-----------------+------------------+------------------+
```

Hình 13. Kết quả các thành phần của dữ liệu

- Sử dụng hàm toPandas.info để kiểm tra lại các cột của tập dữ liệu

```
[ ]  # kiểm tra lại các cột của data
     dfn.toPandas().info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2836 entries, 0 to 2835
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Killed         2836 non-null   int32
 1   Wounded        2836 non-null   int32
 2   AttackType_n   2836 non-null   int32
 3   Target_type_n  2836 non-null   int32
dtypes: int32(4)
memory usage: 44.4 KB
```

Hình 14. kết quả các thành phần của dữ liệu

III. THUẬT TOÁN KHAI THÁC DỮ LIỆU

3.1 Thuật toán K-Means

- Tiến hành xuất dữ liệu sang file .csv. Sau đó đọc nhiều tệp dữ liệu vào một file RDD với số lượng phân mảnh nhỏ nhất (minPartitions) là 20.

```
In [27]: # Xuất ra file đã xử lý, bỏ header
         dfn.toPandas().to_csv(r'TerrorismUS_Clean.csv', index = False, header = False)

In [28]: # đọc dữ liệu vào RDD
         fileRDD = sc.textFile("TerrorismUS_Clean.csv",minPartitions=20)
```

Hình 15. Xuất dữ liệu ra .csv và đọc các tệp vào file RDD

- Thực hiện gắn số dòng cho từng dòng trong data, chuyển dữ liệu sang int*, thêm vào array. Với key là số dòng, bắt đầu từ 0, giá trị là giá trị chuyển sang int* trong array.

```
In [29]: # gắn số dòng cho từng dòng trong data, chuyển dữ liệu sang int *, thêm vào array
         # key là số dòng, bắt đầu từ 0, giá trị là giá trị chuyển sang int * trong array
         fileRDD=fileRDD.zipWithIndex().map(lambda x:(x[1],np.array([int(y) for y in x[0].split(',')])))
```

Hình 16. Xử lý trên từng dòng trong data

- Viết hàm **clust_assn** có chức năng gắn cụm (cluster) gần nhất cho từng điểm dữ liệu (datapoint) **. Đây là bước tìm tâm cụm (Centroid).** Hàm nhận tham số đầu vào là từng dòng trong file RDD và trả về giá trị id của cụm, khoảng cách từ datapoint đến cluster gần nhất, số lượng quan sát và tọa độ.

  Trong đó :

  - X là mỗi dòng của file RDD.
  - Trong function return:
    - Temp là id của cụm.
    - dist: khoảng cách từ datapoint tới cụm (cluster) gần nhất.
    - x[0] là số quan sát.

- x[1] là tọa độ (coordinate).

In [30]:
```
# function: gắn cluster gần nhất cho từng dataPoint
# Đây là bước tìm tâm cụm cho việc khởi tạo
# x là mỗi dòng của RDD

def clust_assn(x):
  # Temp là id của cụm
  temp=0
  # dis: khoảng cách từ datapoint tới cluster gần nhất
  dist=np.inf
  # lặp qua từng tâm cụm và gắn tâm cụm gần nhất cho dataPoint
  for i in clus_bd.value:
    # Tính khoảng cách
    d = np.square(np.sqrt(sum(np.square(np.subtract(x[1], i[1])))))
    # So sánh lấy khoảng cách ngắn hơn
    if d < dist:
      # Gắn lại các giá trị tâm cụm
      dist = d
      temp=i[0]
  return (temp,(dist,x[0],x[1]))
```

Hình 17. Hàm clust_assn

- Viết hàm **generate_initial_centroids** để lấy tâm cụm đầu tiên, xác định tọa độ của tâm và in thực hiện Broadcast trên tâm đầu tiên và tìm các điểm dữ liệu (datapoint) ở xa nhất**.** Hàm nhận tham số đầu vào là số cụm k.

In [31]:
```
# hàm khởi tạo k tâm cụm
def generate_initial_centroids(k):
  # Lấy tâm cụm đầu tiên
  # Lấy ngẫu nhiên một dòng dữ liệu để tạo thành tâm đầu tiên
  first_centroid=random.sample(range(fileRDD.count()),1)
  # Lấy giá trị tọa độ
  first_centroid_cord=fileRDD.filter(lambda x:x[0] in first_centroid)
  # Khai báo và thông báo biến ra ngoài
  global clus_bd
  clus_bd=sc.broadcast(first_centroid_cord.collect())
  for i in range(k-1):
    l = []
    bla=fileRDD.map(lambda x:clust_assn(x))
    next_full = bla.max(key=lambda x: x[1][0]) # tìm điểm ở xa nhất
    next = (next_full[1][1], next_full[1][2])
    # Ghi các giá trị tâm cụm, giải phóng tài nguyên và thông báo ra toàn cụm máy
    clus_bd.value.append(next)
    l = clus_bd.value
    clus_bd.unpersist()
    clus_bd = sc.broadcast(l)
    first_centroid.append(next[0])
```

Hình 18. Hàm generate_initial_centroids

- Viết hàm **euc_distance_assign_cluster** để tính khoảng cách **Euclid** và gắn các điểm dữ liệu (dataPoint) vào cụm (cluster). Hàm nhận tham số đầu vào là mỗi dòng của file RDD và phạm vi n. Kết quả trả về là giá trị id của cụm, số lượng quan sát, tọa độ và chi phí khoảng cách Euclid.

In [32]:
```
# Tính khoảng cách Euclid và gắn dataPoint vào cluster
def euc_distance_assign_cluster(x,n):
  temp=0
  dist=np.inf
  # Duyệt qua tất cả n tâm
  for i in range(n):
    # Tính khoảng cách Euclid d
    d = np.sqrt(sum(np.square(np.subtract(x[1], centroid_centres_bd.value[i][1]))))
    # So sánh lấy khoảng cách ngắn nhất
    if d < dist:
      # Gắn Datapoint vào Cluster
      dist = d
      temp=centroid_centres_bd.value[i][0]
      # Tính chi phí cost
      costf=dist**2
  return (temp,(x[0],x[1],costf))
```

Hình 19. Hàm euc_distance_assign_cluster

- Viết hàm **man_distance_assign_cluster** để tính khoảng cách **Manhattan** và gắn các điểm dữ liệu (dataPoint) vào cụm (cluster). Hàm nhận tham số đầu vào là mỗi dòng của file RDD và phạm vi n. Kết quả trả về là giá trị id của cụm, số lượng quan sát, tọa độ và chi phí khoảng cách Manhattan.

In [33]:
```
# Tính khoảng cách Mahattan và gắn dataPoing vào cluster
def man_distance_assign_cluster(x,n):
  temp=0
  dist=np.inf
  # Duyệt qua tất cả n tâm
  for i in range(n):
    # Tính khoảng cách Mahattan d
    d = np.sqrt(sum(np.absolute(np.subtract(x[1], centroid_centres_bd.value[i][1]))))
    # So sánh lấy khoảng cách ngắn nhất
    if d < dist:
      # Gắn Datapoint vào Cluster
      dist = d
      temp=centroid_centres_bd.value[i][0]
      # Tính chi phí cost
      cost=dist**2
  return (temp,(x[0],x[1],cost))
```

Hình 20. Hàm man_distance_assign_cluster

- Viết hàm **kmeans** để tiến hành chạy thuật toán phân cụm k-means. Hàm nhận tham số đầu vào là method, số cụm, số lượng phần tử bên trong và trả về giá trị cost khi chạy trên hai khoảng cách.

In [34]:
```
def kmeans(method, n_clus, n_iter):
  # khởi tạo các tâm cụm
  generate_initial_centroids(n_clus)
  cf=[]
  # chuyển method thành chữ viết thường (không hoa)
  method = method.lower()
  # khởi tạo và gắn các giá trị global sẽ được broadcast
  global centroid_centres_bd
  global label
  centroid_centres_bd=sc.broadcast(clus_bd.value)
  # Chạy các vòng lặp lại thuật toán
  for i in range(n_iter):
    if method == 'e':
      # tính khoảng cách và gắn datapoin vào cluster sử dụng Euclid
      assign_cluster=fileRDD.map(lambda x: euc_distance_assign_cluster(x,n_clus))
    elif method == 'm':
      # tính khoảng cách và gắn datapoin vào cluster Mahattan
      assign_cluster=fileRDD.map(lambda x: man_distance_assign_cluster(x,n_clus))
    # tính và nạp giá trị cost_func
    cost_func=round(assign_cluster.map(lambda x:x[1][2]).sum(),2)
    cf.append(cost_func)

    assign_cluster=assign_cluster.map(lambda x:(x[0],x[1][1]))
    # Đưa giá trị label ra ngoài
    label=sc.broadcast(assign_cluster.collect())
    # Tính toán các giá trị tâm các cụm
    new_clusters=assign_cluster.mapValues(lambda v: (v, 1)) \
      .reduceByKey(lambda a,b: (a[0]+b[0], a[1]+b[1])) \
      .mapValues(lambda v: v[0]/v[1])
    new_clusters=new_clusters.map(lambda x:x[1]).zipWithIndex().map(lambda x:(x[1],x[0]))
    # giải phóng tài nguyên và thông báo ra ngoài giá trị tâm cụm
    centroid_centres_bd.unpersist()
    centroid_centres_bd=sc.broadcast(new_clusters.collect())
  return cf
```

Hình 21. Hàm kmeans

- Tiến hành chạy thuật toán k-means sử dụng khoảng cách **Manhattan.**

```
  return cf
```

In [35]:
```
# chạy thuật toán kmeans sử dụng khoảng cách Mahattan
costs_man = kmeans('m', 5, 10)
kmpp_centroids_man=[x[1] for x in centroid_centres_bd.value]
```

Hình 22. Chạy giải thuật k-means dùng Mahattan Distance

- Tiến hành chạy thuật toán k-means sử dụng khoảng cách **Euclid.**

```
# chạy thuật toán kmeans sử dụng khoảng cách Euclid
costs_eu = kmeans('e', 5, 10)
kmpp_centroids_eu=[x[1] for x in centroid_centres_bd.value]
```

Hình 23. Chạy giải thuật k-means dùng Euclid Distance

- In các nhãn cụm (cluster) của dữ liệu ra và tính toán chi phí thuật toán sử dụng trên hai khoảng cách **Euclid** và **Manhattan**.

In [36]:
```
# in các nhãn cụm của dữ liệu ra
with open('label.csv', "w") as output:
    output.write('label')
    for x in label.value:
        output.write('\n'+str(x[0]))
```

Hình 24. In các nhãn cụm của dữ liệu

- Tính toán chi phí thuật toán sử dụng trên hai khoảng cách **Euclid** và **Manhattan**.

In [37]:
```
# chi phí thuật toán sử dụng Euclid
costs_eu[9]
```

Out[37]: 725133.69

In [38]:
```
# Chi phí thuật toán sử dụng Mahattan
costs_man[9]
```

Out[38]: 32109.32

Hình 25. Tính toán chi phí thuật toán sử dụng trên hai khoảng cách

⇨ Quan sát được chi phí thuật toán với khoảng cách Euclid cao hơn nhiều so với chi phí trên khoảng cách Mahattan.

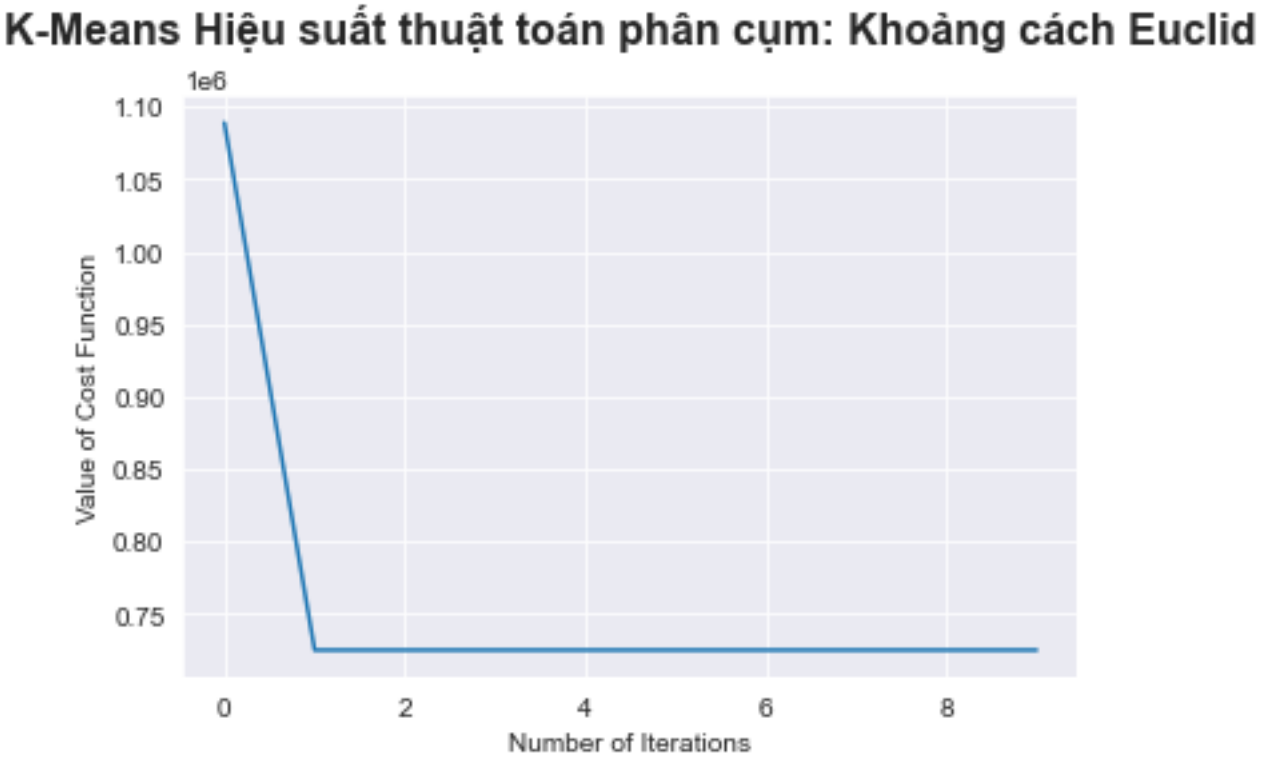
- In ra tâm các cụm có k-means sử dụng khoảng cách **Euclid** và khoảng cách **Manhattan**.

```
In [39]: # In ra tâm các cụm có kmeans sử dụng khoảng cách Euclid
         with open('output_eu.txt', "w") as output:
             output.write(str(kmpp_centroids_eu))
```

```
In [40]: # In ra tâm các cụm có kmeans sử dụng khoảng cách Mahattan
         with open('output_tman.txt', "w") as output:
             output.write(str(kmpp_centroids_man))
```

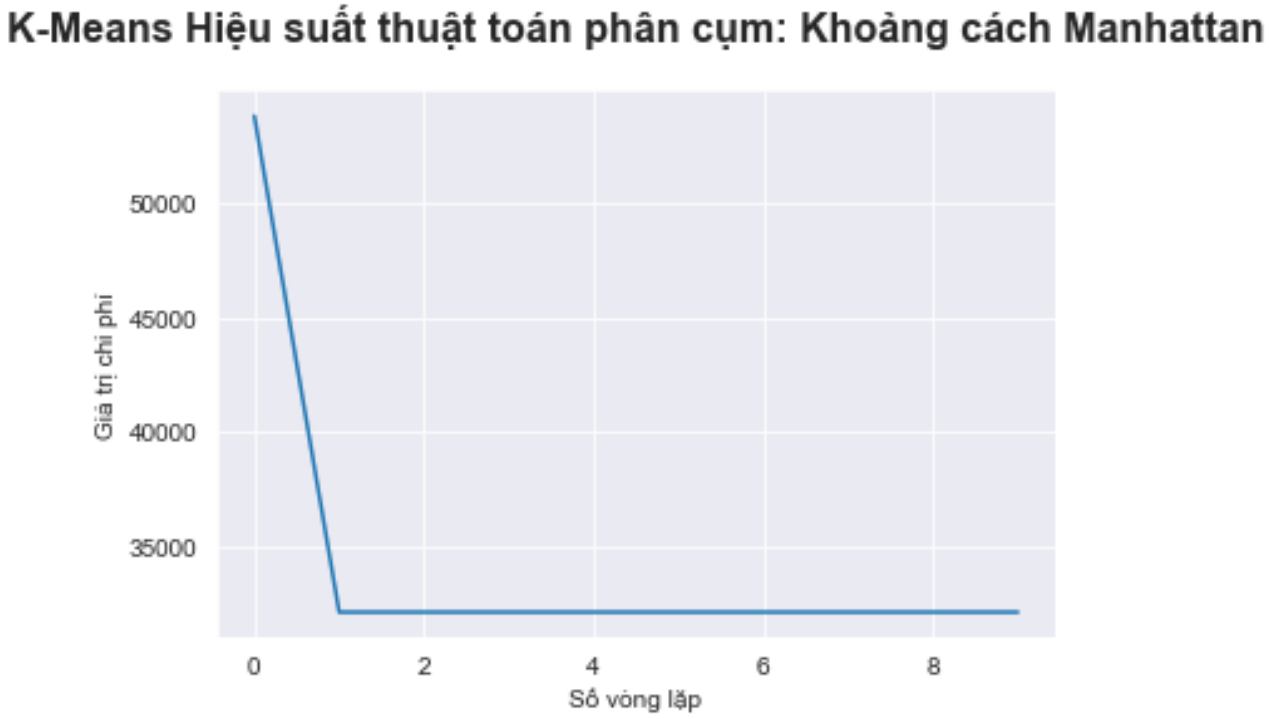
Hình 26. In các tâm cụm có kmeans sử dụng 2 khoảng cách

- Hiệu suất thuật toán phân cụm đạt được khi thực hiện trên khoảng cách **Euclid**.



Hình 27. Kết quả hiệu suất giải thuật với khoảng cách Euclid

- Hiệu suất thuật toán phân cụm đạt được khi thực hiện trên khoảng cách **Manhattan**.



Hình 28. Kết quả hiệu suất giải thuật với khoảng cách Mahattan

- Tính toán chi phí giảm thiểu khi sử dụng khoảng cách **Euclid** sau khi thực hiện qua 20 vòng lặp.

```
In [44]: per_chan_e=round(((costs_eu[0]-costs_eu[9])/costs_eu[0])*100,2)
         print("Chi phí giảm thiểu khi sử dụng khoảng cách Euclid sau 10 vòng lặp: % " + str(per_chan_e))

         Chi phí giảm thiểu khi sử dụng khoảng cách Euclid sau 20 vòng lặp: % 33.4
```

Hình 29. Kết quả chi phí giảm thiểu thực hiện trên Euclid Distance

⇨ Chi phí ước tính giảm thiểu khoảng 33.4%.

- Tính toán chi phí giảm thiểu khi sử dụng khoảng cách **Manhattan** sau khi thực hiện qua 20 vòng lặp.

```
In [45]: per_chan_m=round(((costs_man[0]-costs_man[9])/costs_man[0])*100,2)
         print("Chi phí giảm thiểu khi sử dụng khoảng cách Mahattan sau 10 vòng lặp: % " + str(per_chan_m))

         Chi phí giảm thiểu khi sử dụng khoảng cách Mahattan sau 10 vòng lặp: % 40.34
```

Hình 30. Kết quả chi phí giảm thiểu thực hiện trên Manhattan Distance

⇨ Chi phí ước tính giảm thiểu khoảng 40.34%.

- Import các thư viện để xử lý dữ liệu trước khi thực hiện thuật toán để tìm đặc trưng cụm.

```
In [53]: from pyspark.sql.functions import monotonically_increasing_id, row_number
         from pyspark.sql.window import Window

         # Đọc lại file dữ liệu gốc cùng file gắn nhãn và gộp cả hai lại
         df2 = spark.read.option("header",True) \
              .csv("khungbo/terrorism-in-US.csv")
         df3 = spark.read.option("header",True) \
              .csv("label.csv")
         # Sử dụng row_index để thay thế cho các id dòng
         df2=df2.withColumn('row_index', row_number().over(Window.orderBy(monotonically_increasing_id())))
         df3=df3.withColumn('row_index', row_number().over(Window.orderBy(monotonically_increasing_id())))
         # Join dữ liệu dựa trên Row_index sau đó xóa Row_index
         df2 = df2.join(df3, on=["row_index"]).drop("row_index")

         # Cắt bỏ những cột dữ liệu không dùng tới
         columns_to_drop = ['Target','Latitude','Longitude','State','Summary','Group','Motive','Country','Region']
         newdf= df2.drop(*columns_to_drop)

         newdf.show(n=5)
```

```
+----+-----+---+-------+--------------------+------+-------+--------------------+-----------+-----+
|Year|Month|Day|   City|          AttackType|Killed|Wounded|         Target_type|Weapon_type|label|
+----+-----+---+-------+--------------------+------+-------+--------------------+-----------+-----+
|1970|    1|  1|  Cairo|       Armed Assault|     0|      0|              Police|   Firearms|    2|
|1970|    1|  2|Oakland|   Bombing/Explosion|     0|      0|           Utilities| Explosives|    2|
|1970|    1|  2|Madison|Facility/Infrastr...|     0|      0|            Military| Incendiary|    2|
|1970|    1|  3|Madison|Facility/Infrastr...|     0|      0|Government (General)| Incendiary|    2|
|1970|    1|  1|Baraboo|   Bombing/Explosion|     0|      0|            Military| Explosives|    2|
+----+-----+---+-------+--------------------+------+-------+--------------------+-----------+-----+
only showing top 5 rows
```

Hình 31. Kết quả kiểm tra một vài dòng dữ liệu đầu sau khi xử lý

- **Cụm 0 :**

In [58]:
```
# Cụm 0
df_filtered=newdf.filter(newdf.label == '0')
#columns_to_drop = ['status_id','status_published']
df_filtered= df_filtered.drop(*columns_to_drop)
# In ra 10 dòng dữ liệu của cụm
df_filtered.show(n=10)

# In ra các thống kê dữ liệu trong cụm
df_filtered.describe().show()

# Lọc các dữ liệu theo năm
xx = df_filtered.groupBy('Year').count()
arr={}

# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}
```

```
# in ra các giá trị loại tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('Weapon_type').count()
arr={}

# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

- Hiển thị kết quả

-

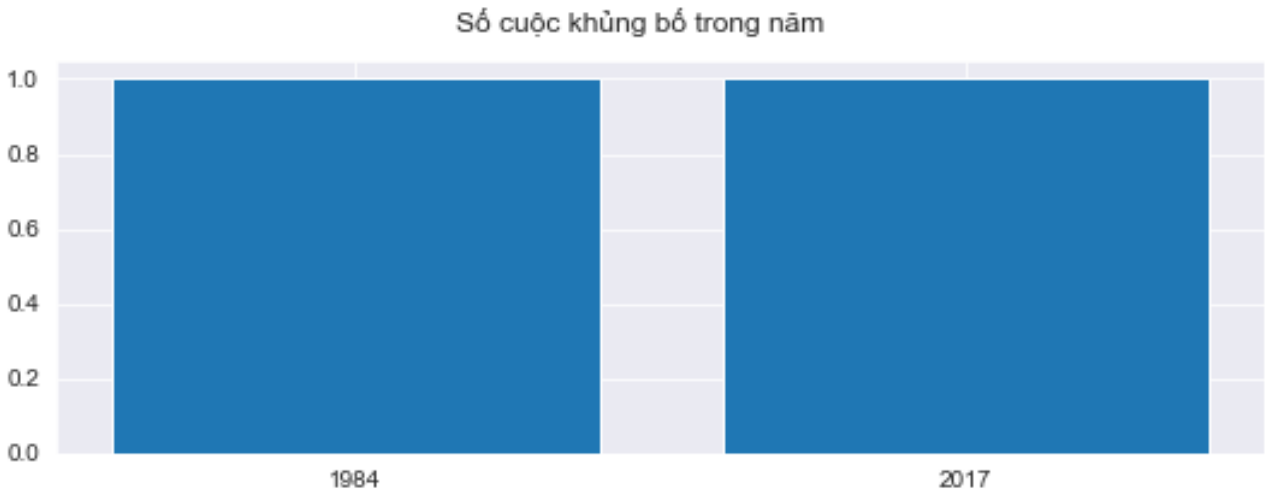
```
+----+-----+---+----------+---------------+------+-------+-----------+-----------+-----+
|Year|Month|Day|      City|     AttackType|Killed|Wounded|Target_type|Weapon_type|label|
+----+-----+---+----------+---------------+------+-------+-----------+-----------+-----+
|1984|    9| 20|The Dalles|Unarmed Assault|     0|    751|   Business| Biological|    0|
|2017|   10|  1| Las Vegas|  Armed Assault|    59|    851|   Business|   Firearms|    0|
+----+-----+---+----------+---------------+------+-------+-----------+-----------+-----+
```

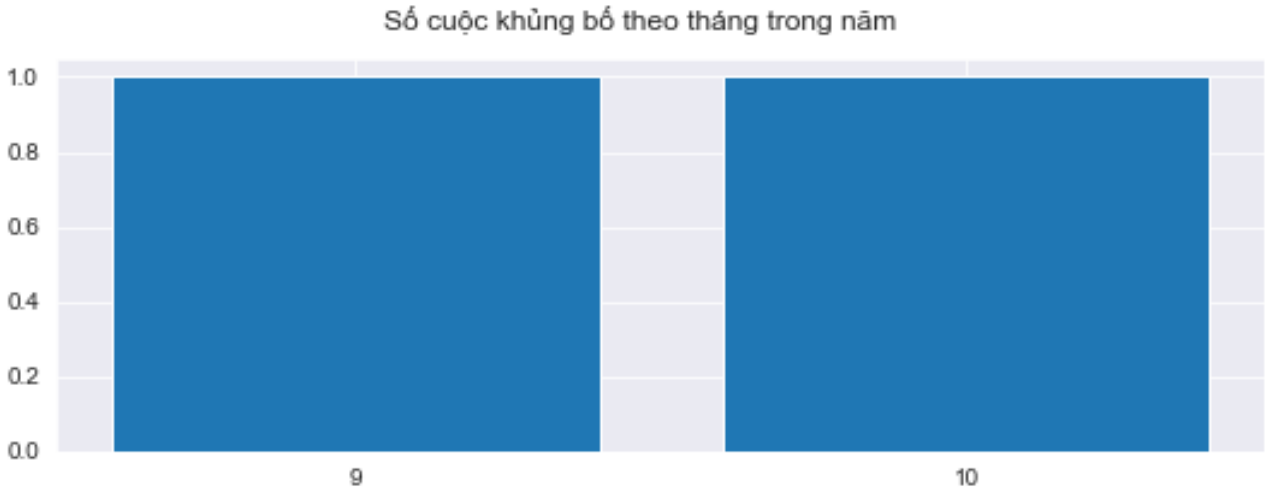
- Hiển thị các thành phần của dữ liệu.

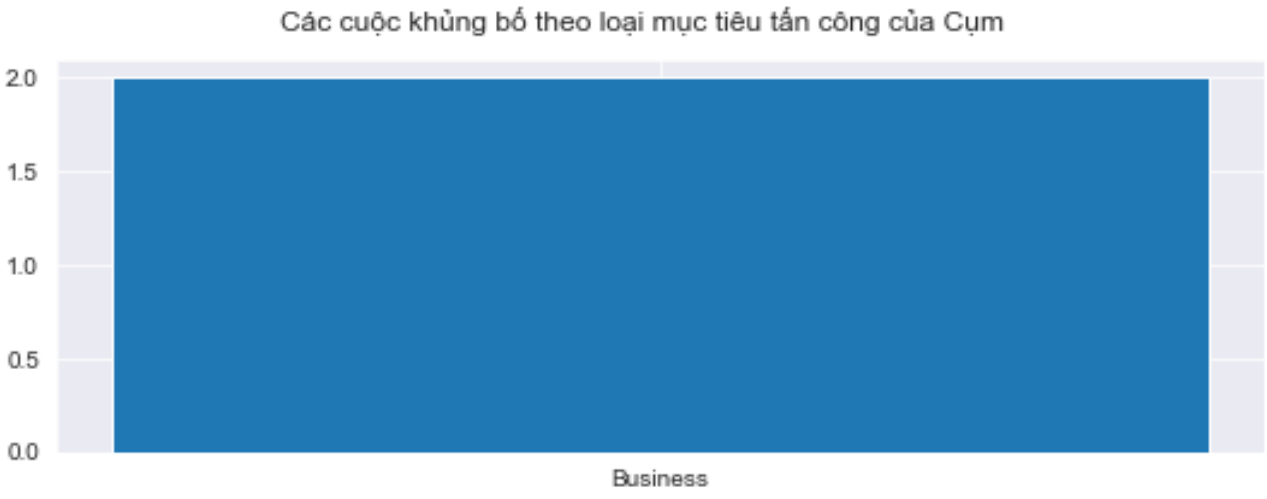
-

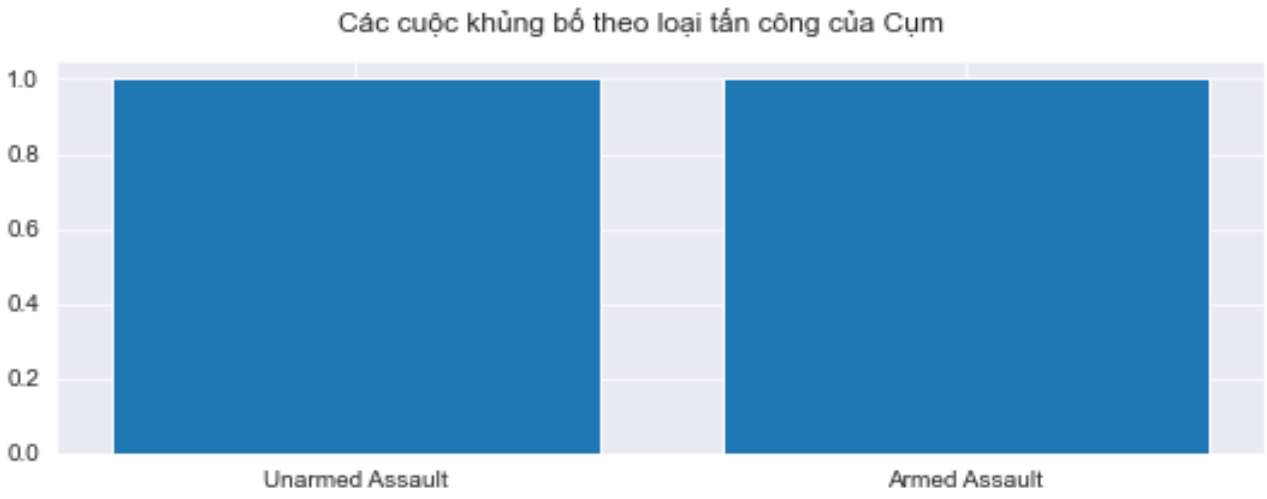
```
+-------+-----------------+-----------------+-----------------+----------+--------------+-----------------+-----------------+-----------
+-----------+-----------+-----+
|summary|             Year|            Month|              Day|      City|     AttackType|           Killed|          Wounded
|Target_type|Weapon_type|label|
+-------+-----------------+-----------------+-----------------+----------+--------------+-----------------+-----------------+-----------
+-----------+-----------+-----+
|  count|                2|                2|                2|         2|             2|                2|                2
|          2|          2|    2|
|   mean|           2000.5|              9.5|             10.5|      null|          null|             29.5|            801.0
|       null|       null|  0.0|
| stddev|23.33452377915607|0.7071067811865476|13.435028842544403|      null|          null|41.71930009000631|70.71067811865476
|       null|       null|  0.0|
|    min|             1984|               10|                1| Las Vegas|  Armed Assault|                0|              751
|   Business| Biological|    0|
|    max|             2017|                9|               20|The Dalles|Unarmed Assault|               59|              851
|   Business|   Firearms|    0|
+-------+-----------------+-----------------+-----------------+----------+--------------+-----------------+-----------------+-----------
+-----------+-----------+-----+

1984: 1
2017: 1
9: 1
10: 1
Business: 2
Unarmed Assault: 1
Armed Assault: 1
Biological: 1
Firearms: 1
```

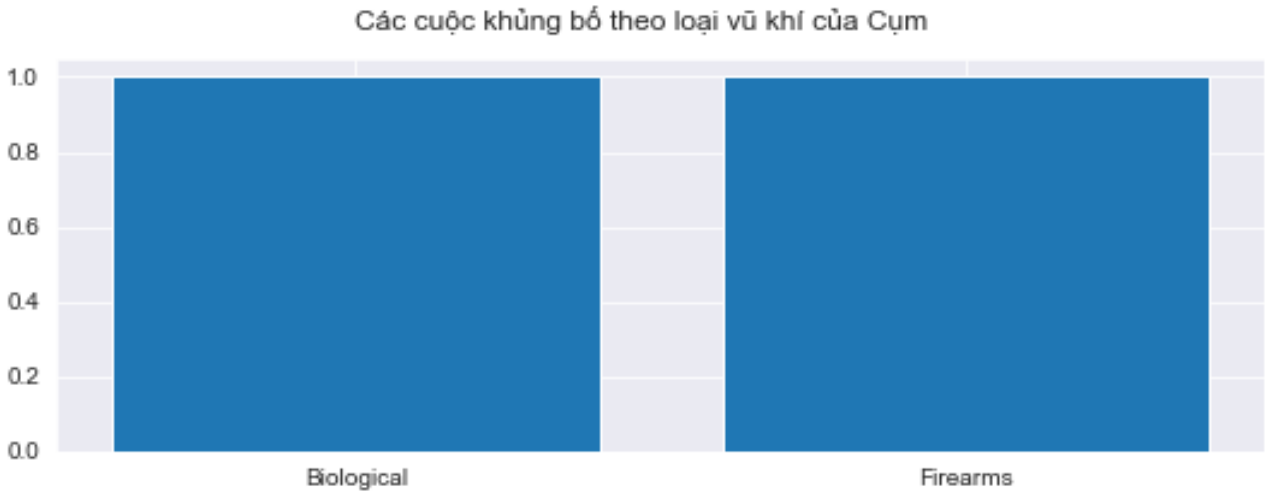
- Kết quả cụm 0


Số cuộc khủng bố trong năm
1.0
0.8
0.6
0.4
0.2
0.0
1984
2017


Số cuộc khủng bố theo tháng trong năm
1.0
0.8
0.6
0.4
0.2
0.0
9
10


Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm
2.0
1.5
1.0
0.5
0.0
Business


Các cuộc khủng bố theo loại tấn công của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Unarmed Assault
Armed Assault


Các cuộc khủng bố theo loại vũ khí của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Biological
Firearms

- **Cụm 1 :**

In [59]:
```
# Cụm 1
df_filtered=newdf.filter(newdf.label == '1')
#columns_to_drop = ['status_id','status_published']
df_filtered= df_filtered.drop(*columns_to_drop)
# In ra 10 dòng dữ liệu của cụm
df_filtered.show(n=10)

# In ra các thống kê dữ liệu trong cụm
df_filtered.describe().show()

# Lọc các dữ liệu theo năm
xx = df_filtered.groupBy('Year').count()
arr={}

# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}
```

```
# in ra các giá trị tháng trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị về thời gian cuộc khủng bố theo tháng trong năm
fig.suptitle('Số cuộc khủng bố theo tháng trong năm')

# Lọc các dữ liệu theo loại mục tiêu tấn công
xx = df_filtered.groupBy('Target_type').count()
arr={}

# in ra các giá trị loại mục tiêu tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('AttackType').count()
arr={}
```

```
# in ra các giá trị loại tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('Weapon_type').count()
arr={}

# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

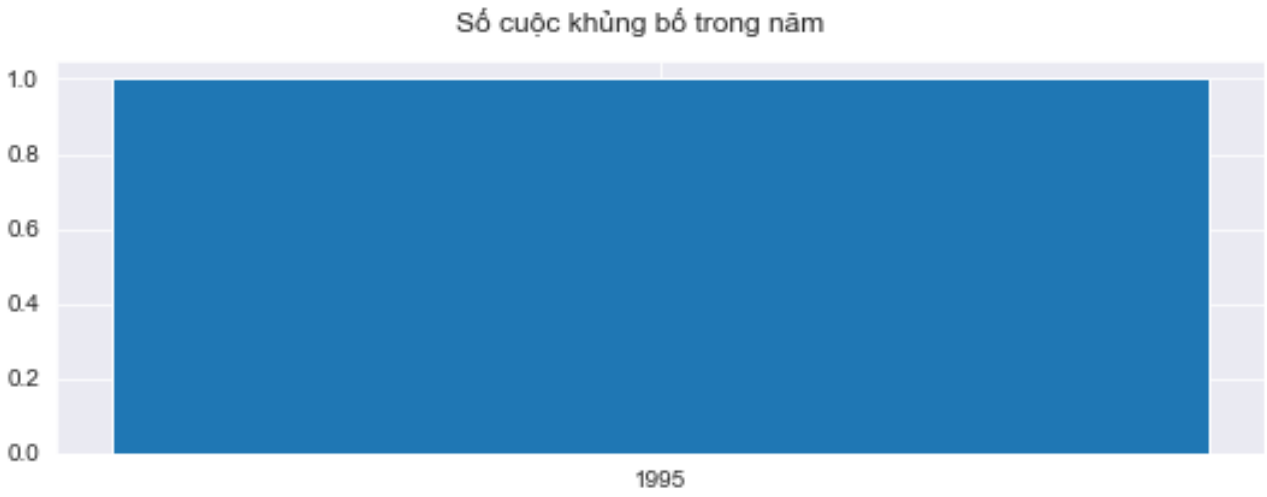
- Hiển thị kết quả

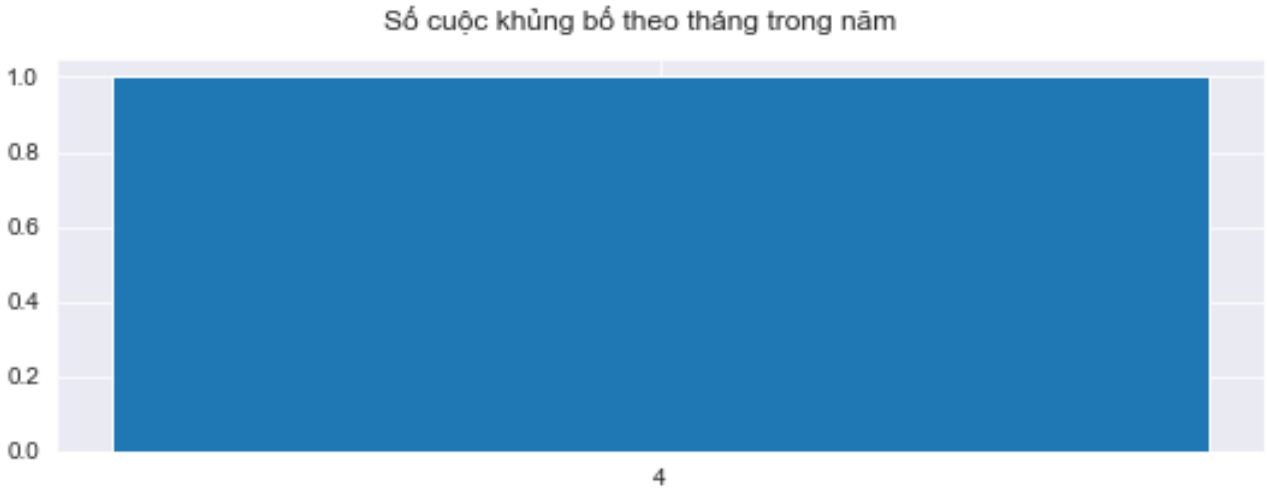
```
+----+-----+---+-------------+-----------------+------+-------+--------------------+-----------+-----+
|Year|Month|Day|         City|       AttackType|Killed|Wounded|         Target_type|Weapon_type|label|
+----+-----+---+-------------+-----------------+------+-------+--------------------+-----------+-----+
|1995|    4| 19|Oklahoma City|Bombing/Explosion|   168|    650|Government (General)| Explosives|    1|
+----+-----+---+-------------+-----------------+------+-------+--------------------+-----------+-----+
```

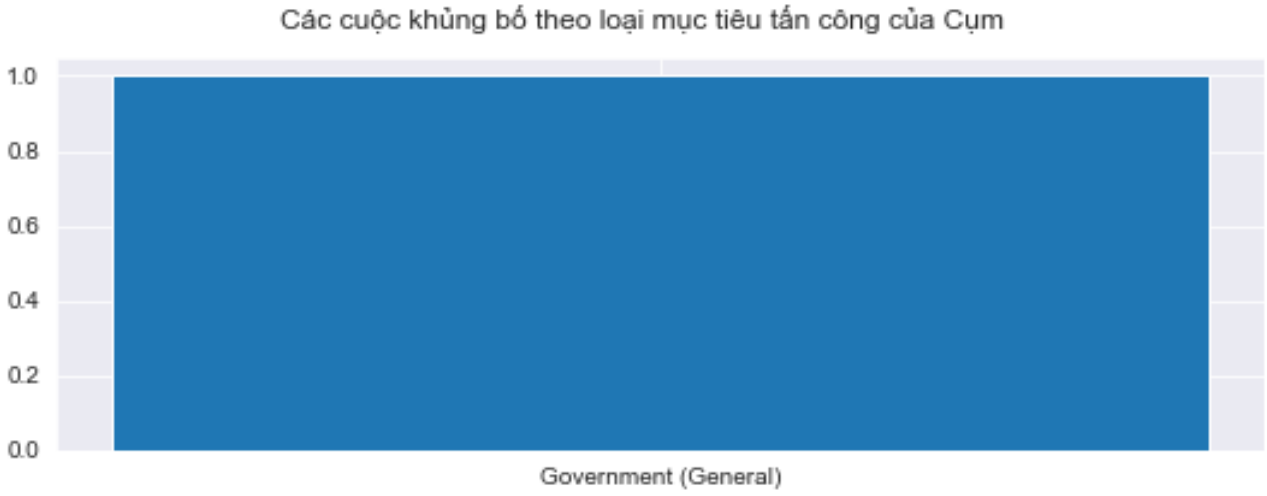
- Hiển thị các thành phần của dữ liệu
-

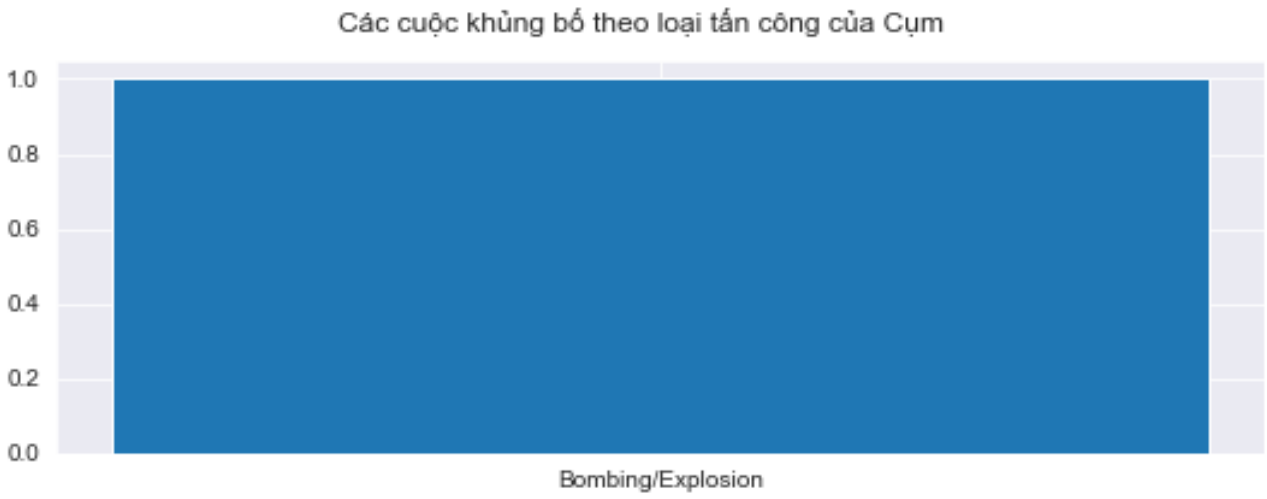
```
+-------+------+-----+----+-------------+-----------------+------+-------+--------------------+-----------+-----+
|summary|  Year|Month| Day|         City|       AttackType|Killed|Wounded|         Target_type|Weapon_type|label|
+-------+------+-----+----+-------------+-----------------+------+-------+--------------------+-----------+-----+
|  count|     1|    1|   1|            1|                1|     1|      1|                   1|          1|    1|
|   mean|1995.0|  4.0|19.0|         null|             null| 168.0|  650.0|                null|       null|  1.0|
| stddev|  null| null|null|         null|             null|  null|   null|                null|       null| null|
|    min|  1995|    4|  19|Oklahoma City|Bombing/Explosion|   168|    650|Government (General)| Explosives|    1|
|    max|  1995|    4|  19|Oklahoma City|Bombing/Explosion|   168|    650|Government (General)| Explosives|    1|
+-------+------+-----+----+-------------+-----------------+------+-------+--------------------+-----------+-----+

1995: 1
4: 1
Government (General): 1
Bombing/Explosion: 1
Explosives: 1
```

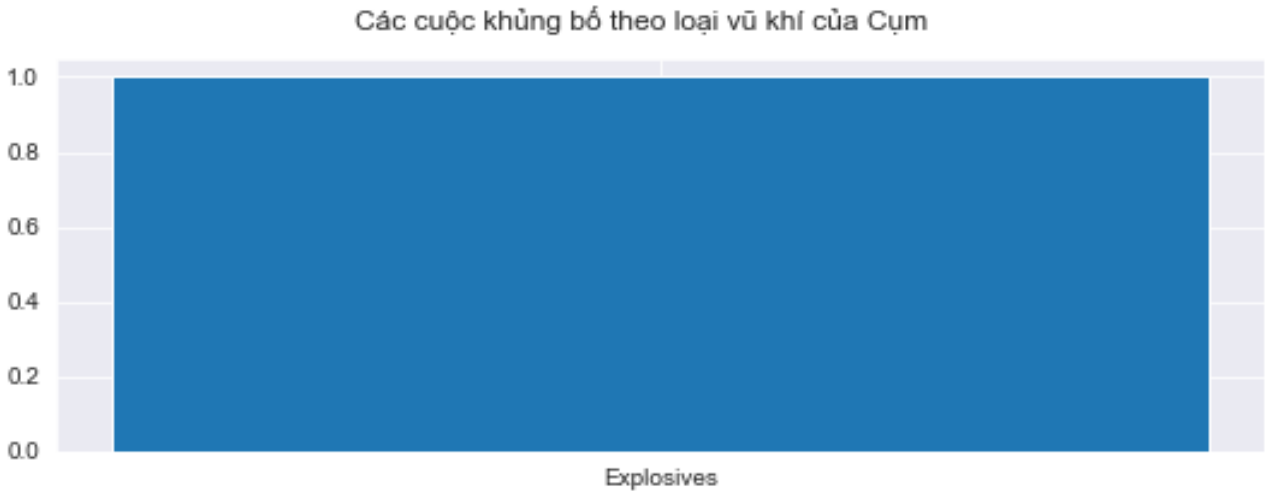
- Kết quả cụm 1


Số cuộc khủng bố trong năm
1.0
0.8
0.6
0.4
0.2
0.0
1995


Số cuộc khủng bố theo tháng trong năm
1.0
0.8
0.6
0.4
0.2
0.0
4


Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Government (General)


Các cuộc khủng bố theo loại tấn công của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Bombing/Explosion


Các cuộc khủng bố theo loại vũ khí của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Explosives

- **Cụm 2 :**

In [60]:
```
# Cụm 2
df_filtered=newdf.filter(newdf.label == '2')
#columns_to_drop = ['status_id','status_published']
df_filtered= df_filtered.drop(*columns_to_drop)
# In ra 10 dòng dữ liệu của cụm
df_filtered.show(n=10)

# In ra các thống kê dữ liệu trong cụm
df_filtered.describe().show()

# Lọc các dữ liệu theo năm
xx = df_filtered.groupBy('Year').count()
arr={}

# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}

# in ra các giá trị tháng trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
```

```
# In ra đồ thị về thời gian cuộc khủng bố theo tháng trong năm
fig.suptitle('Số cuộc khủng bố theo tháng trong năm')

# Lọc các dữ liệu theo loại mục tiêu tấn công
xx = df_filtered.groupBy('Target_type').count()
arr={}

# in ra các giá trị loại mục tiêu tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('AttackType').count()
arr={}

# in ra các giá trị loại tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')
```

```
# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

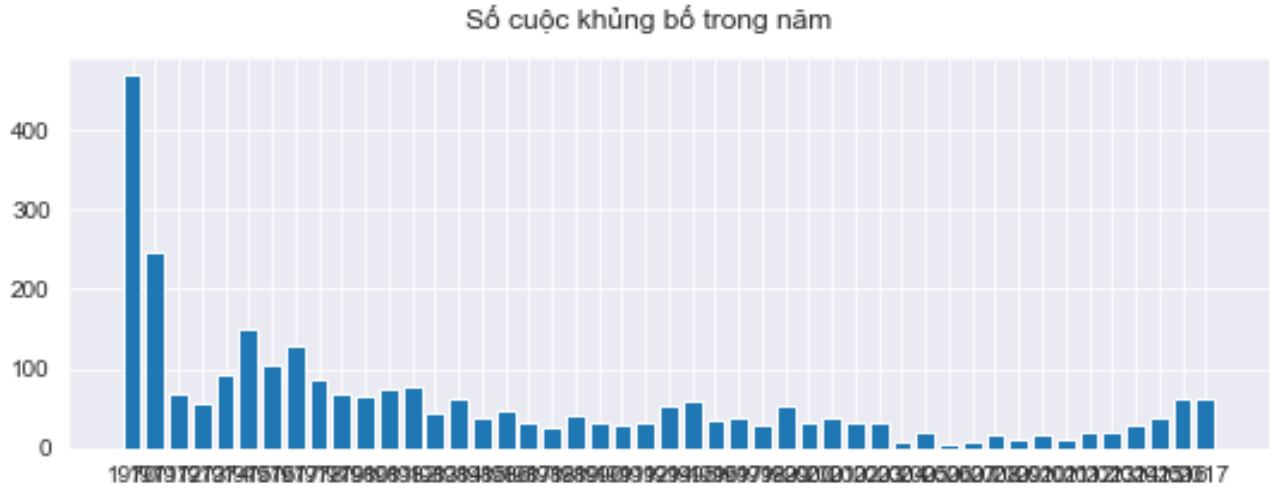
- Hiển thị kết quả 10 dòng đầu

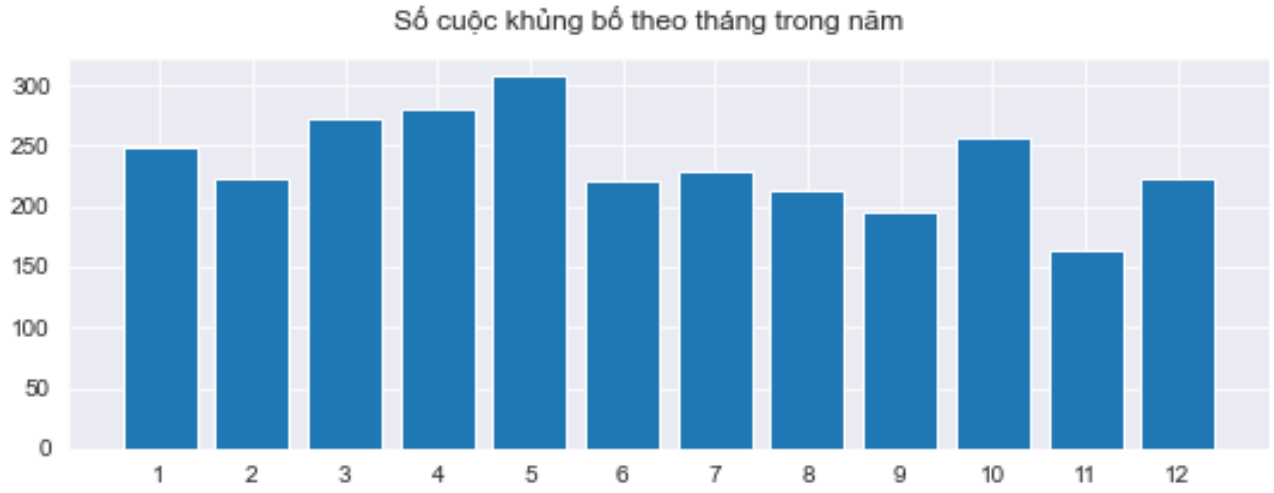
```
+----+-----+---+-------------+--------------------+------+-------+--------------------+-----------+-----+
|Year|Month|Day|         City|          AttackType|Killed|Wounded|         Target_type|Weapon_type|label|
+----+-----+---+-------------+--------------------+------+-------+--------------------+-----------+-----+
|1970|    1|  1|        Cairo|       Armed Assault|     0|      0|              Police|   Firearms|    2|
|1970|    1|  2|      Oakland|   Bombing/Explosion|     0|      0|           Utilities| Explosives|    2|
|1970|    1|  2|      Madison|Facility/Infrastr...|     0|      0|            Military| Incendiary|    2|
|1970|    1|  3|      Madison|Facility/Infrastr...|     0|      0|Government (General)| Incendiary|    2|
|1970|    1|  1|      Baraboo|   Bombing/Explosion|     0|      0|            Military| Explosives|    2|
|1970|    1|  6|       Denver|Facility/Infrastr...|     0|      0|            Military| Incendiary|    2|
|1970|    1|  9|      Detroit|Facility/Infrastr...|     0|      0|Government (General)| Incendiary|    2|
|1970|    1|  9|  Rio Piedras|Facility/Infrastr...|     0|      0|            Business| Incendiary|    2|
|1970|    1| 12|New York City|   Bombing/Explosion|     0|      0|Educational Insti...| Explosives|    2|
|1970|    1| 12|   Rio Grande|   Bombing/Explosion|     0|      0|            Business| Explosives|    2|
+----+-----+---+-------------+--------------------+------+-------+--------------------+-----------+-----+
only showing top 10 rows
```

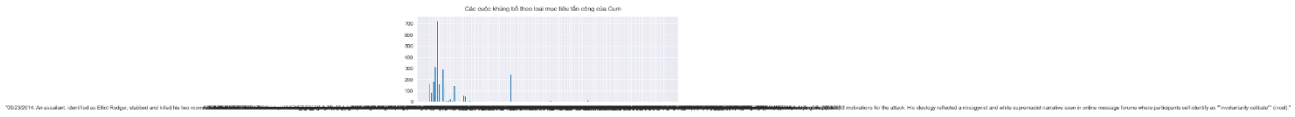
- Hiển thị các thành phần của dữ liệu

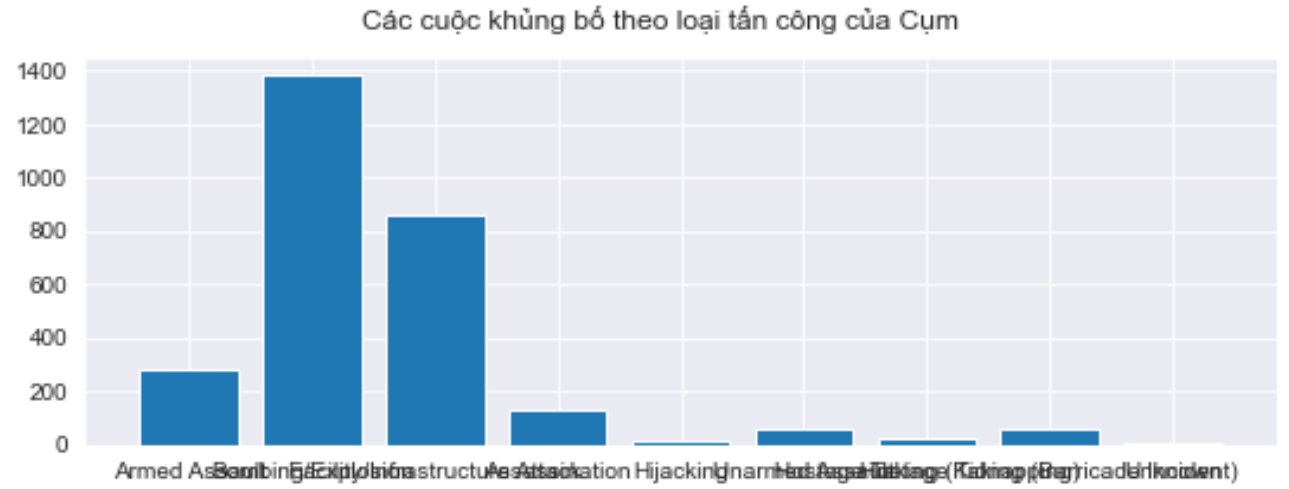
```
+-------+------------------+------------------+------------------+---------+------------+------------------+--------------
---+--------------------+--------------------+-----+
|summary|              Year|             Month|               Day|     City|  AttackType|            Killed|          Woun
ded|         Target_type|        Weapon_type|label|
+-------+------------------+------------------+------------------+---------+------------+------------------+--------------
---+--------------------+--------------------+-----+
|  count|              2830|              2830|              2830|     2830|        2830|              2757|             2
737|                2830|                2830| 2830|
|   mean|1983.9512367491166|6.2180212014134275|15.323674911660778|     null|        null|0.21291258614435982|0.7172086225794
666|                null|                 0.0|  2.0|
| stddev|14.182934381415324|3.3934043337527937| 9.148774539120696|     null|        null| 1.5324288839766855| 6.002409344284
871|                null|                null|  0.0|
|    min|              1970|                 1|                 0|   Afton|Armed Assault|                 0|
0| ""The action was...| California in th...|    2|
|    max|              2017|                 9|                 9|Yuba City|     Unknown|                 9|
9|    White extremists|       Zebra killers|    2|
+-------+------------------+------------------+------------------+---------+------------+------------------+--------------
---+--------------------+--------------------+-----+
```

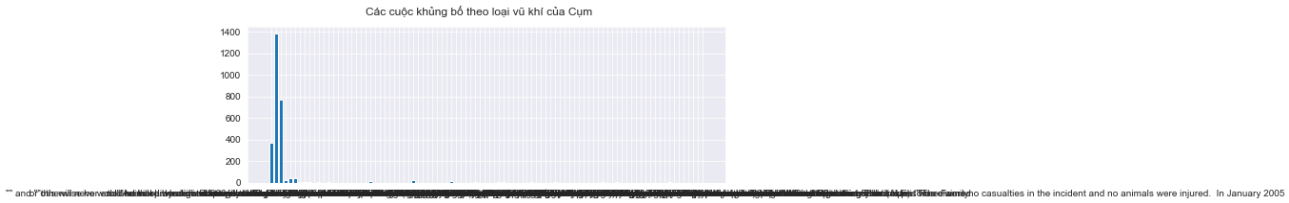
- Kết quả cụm 2








Các cuộc khủng bố theo loại tấn công của Cụm
1400
1200
1000
800
600
400
200
0


Các cuộc khủng bố theo loại vũ khí của Cụm
1400
1200
1000
800
600
400
200
0

- **Cụm 3** :

In [61]:
```
# Cụm 3
df_filtered=newdf.filter(newdf.label == '3')
#columns_to_drop = ['status_id','status_published']
df_filtered= df_filtered.drop(*columns_to_drop)
# In ra 10 dòng dữ liệu của cụm
df_filtered.show(n=10)

# In ra các thống kê dữ liệu trong cụm
df_filtered.describe().show()

# Lọc các dữ liệu theo năm
xx = df_filtered.groupBy('Year').count()
arr={}

# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}
```

```
# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}

# in ra các giá trị tháng trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị về thời gian cuộc khủng bố theo tháng trong năm
fig.suptitle('Số cuộc khủng bố theo tháng trong năm')

# Lọc các dữ liệu theo loại mục tiêu tấn công
xx = df_filtered.groupBy('Target_type').count()
arr={}
```

```
# in ra các giá trị loại mục tiêu tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('AttackType').count()
arr={}

# in ra các giá trị loại tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('Weapon_type').count()
arr={}
```

```
arr={}

# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

```
arr={}

# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

- Hiển thị kết quả

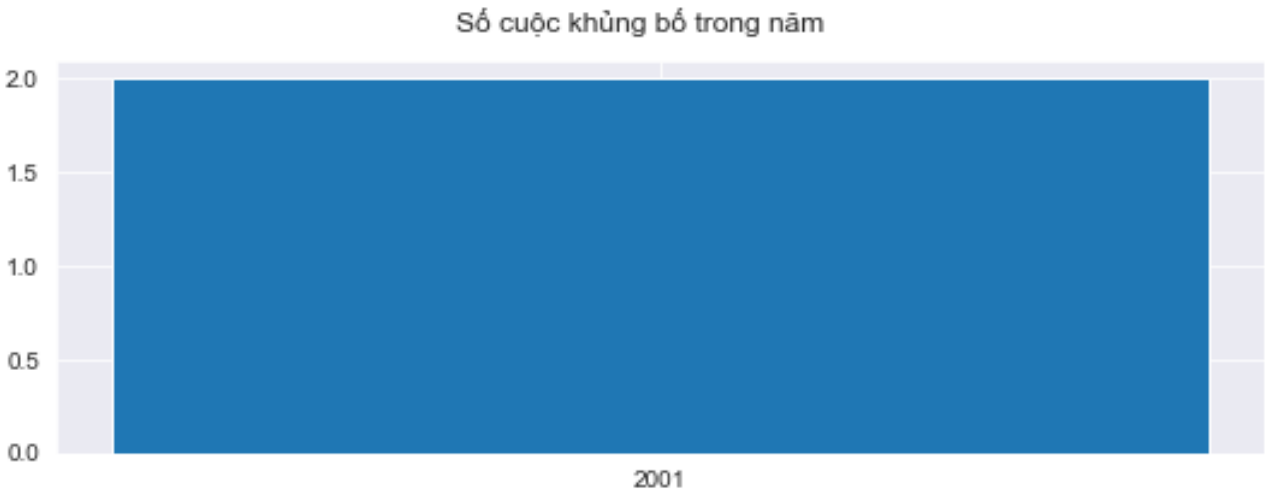
```
+----+-----+---+-------------+----------+------+-------+--------------------+--------------------+-----+
|Year|Month|Day|         City|AttackType|Killed|Wounded|         Target_type|        Weapon_type|label|
+----+-----+---+-------------+----------+------+-------+--------------------+--------------------+-----+
|2001|    9| 11|New York City| Hijacking|  1384|   8190|Private Citizens ...|Vehicle (not to i...|    3|
|2001|    9| 11|New York City| Hijacking|  1383|   8191|Private Citizens ...|Vehicle (not to i...|    3|
+----+-----+---+-------------+----------+------+-------+--------------------+--------------------+-----+
```

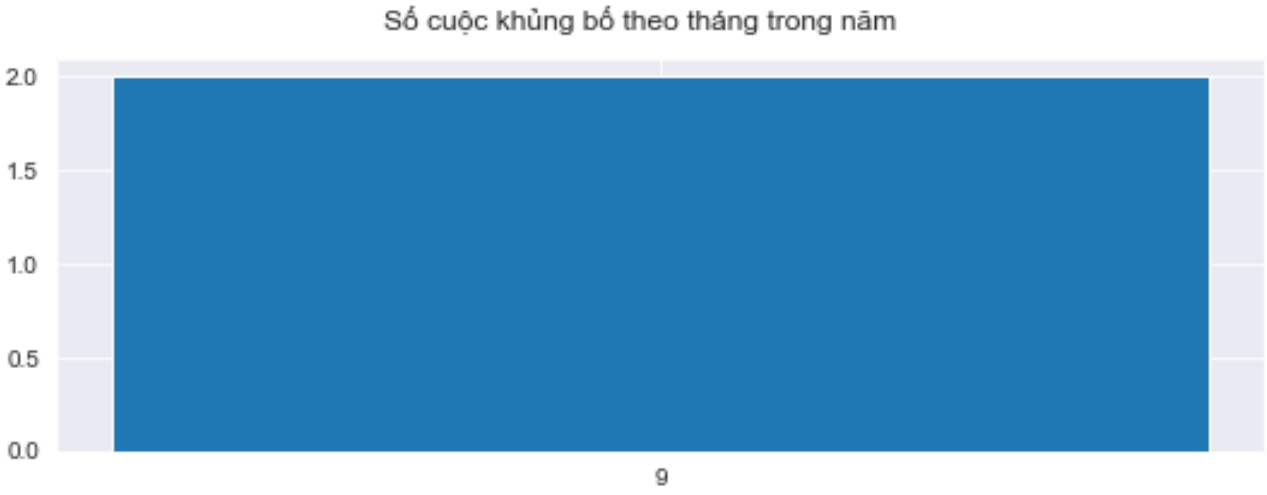
- Hiển thị các thành phần của dữ liệu

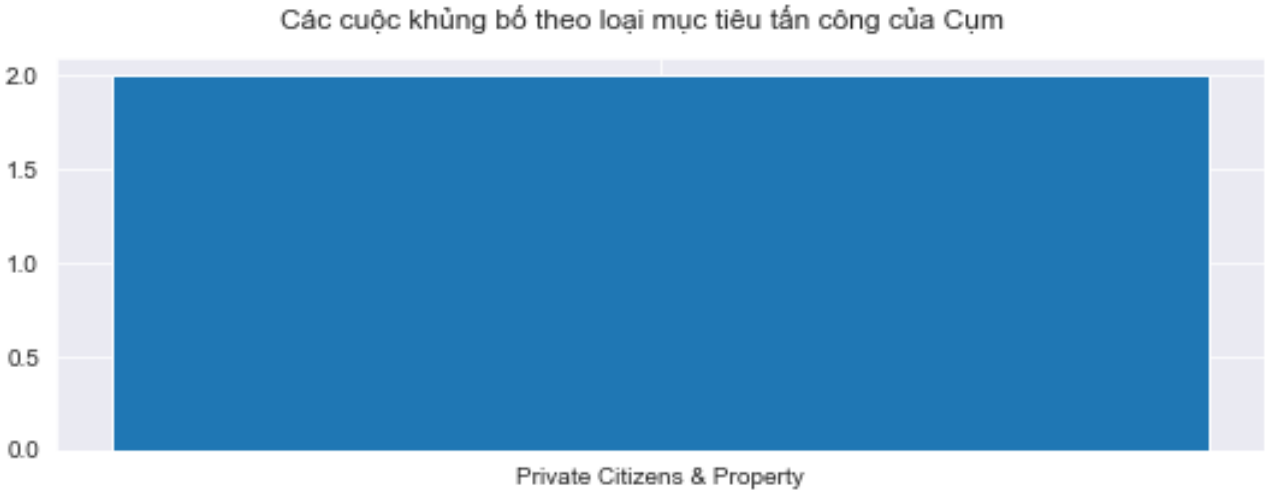
```
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
|summary|  Year|Month| Day|         City|AttackType|            Killed|           Wounded|         Target_type|         Weapon_
type|label|
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
|  count|     2|    2|   2|            2|         2|                 2|                 2|                   2|
2|    2|
|   mean|2001.0|  9.0|11.0|         null|      null|            1383.5|            8190.5|                null|
null|  3.0|
| stddev|   0.0|  0.0| 0.0|         null|      null|0.7071067811865476|0.7071067811865476|                null|
null|  0.0|
|    min|  2001|    9|  11|New York City| Hijacking|              1383|              8190|Private Citizens ...|Vehicle (not to
i...|    3|
|    max|  2001|    9|  11|New York City| Hijacking|              1384|              8191|Private Citizens ...|Vehicle (not to
i...|    3|
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
```

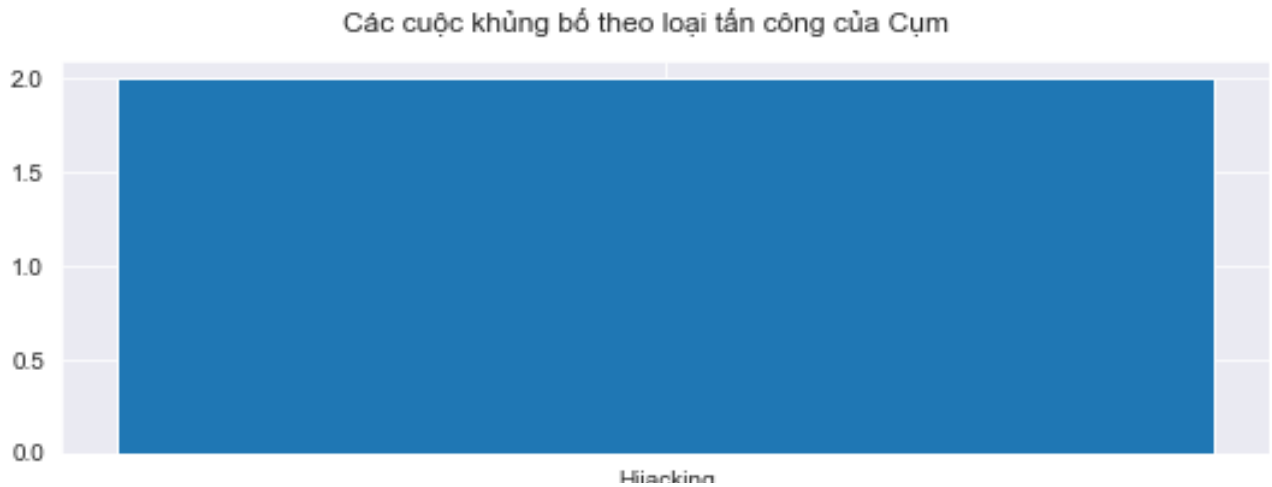
```
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
|summary|  Year|Month| Day|         City|AttackType|            Killed|           Wounded|         Target_type|         Weapon_
type|label|
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
|  count|     2|    2|   2|            2|         2|                 2|                 2|                   2|
2|    2|
|   mean|2001.0|  9.0|11.0|         null|      null|            1383.5|            8190.5|                null|
null|  3.0|
| stddev|   0.0|  0.0| 0.0|         null|      null|0.7071067811865476|0.7071067811865476|                null|
null|  0.0|
|    min|  2001|    9|  11|New York City| Hijacking|              1383|              8190|Private Citizens ...|Vehicle (not to
i...|    3|
|    max|  2001|    9|  11|New York City| Hijacking|              1384|              8191|Private Citizens ...|Vehicle (not to
i...|    3|
+-------+------+-----+----+-------------+----------+------------------+------------------+--------------------+----------
----+-----+
```

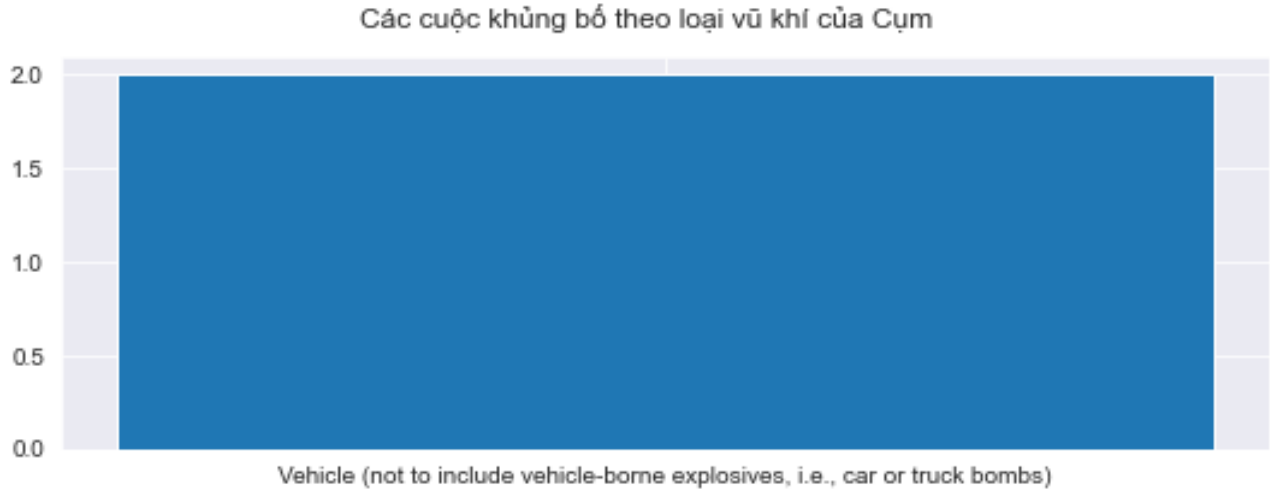
- Kết quả cụm 3


Số cuộc khủng bố trong năm
2.0
1.5
1.0
0.5
0.0
2001


Số cuộc khủng bố theo tháng trong năm
2.0
1.5
1.0
0.5
0.0
9


Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm
2.0
1.5
1.0
0.5
0.0
Private Citizens & Property


Các cuộc khủng bố theo loại tấn công của Cụm
2.0
1.5
1.0
0.5
0.0
Hijacking


Các cuộc khủng bố theo loại vũ khí của Cụm
2.0
1.5
1.0
0.5
0.0
Vehicle (not to include vehicle-borne explosives, i.e., car or truck bombs)

- **Cụm 4 :**

In [62]:
```
# Cụm 4
df_filtered=newdf.filter(newdf.label == '4')
#columns_to_drop = ['status_id','status_published']
df_filtered= df_filtered.drop(*columns_to_drop)
# In ra 10 dòng dữ liệu của cụm
df_filtered.show(n=10)

# In ra các thống kê dữ liệu trong cụm
df_filtered.describe().show()

# Lọc các dữ liệu theo năm
xx = df_filtered.groupBy('Year').count()
arr={}

# in ra các giá trị trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Số cuộc khủng bố trong năm')

# Lọc các dữ liệu theo tháng trong năm
xx = df_filtered.groupBy('Month').count()
arr={}
```

```
# in ra các giá trị tháng trong năm và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị về thời gian cuộc khủng bố theo tháng trong năm
fig.suptitle('Số cuộc khủng bố theo tháng trong năm')

# Lọc các dữ liệu theo loại mục tiêu tấn công
xx = df_filtered.groupBy('Target_type').count()
arr={}

# in ra các giá trị loại mục tiêu tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('AttackType').count()
arr={}
```

```
# in ra các giá trị loại mục tiêu tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('AttackType').count()
arr={}

# in ra các giá trị loại tấn công và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')
```

```
fig.suptitle('Các cuộc khủng bố theo loại tấn công của Cụm')

# Lọc các dữ liệu theo loại tấn công
xx = df_filtered.groupBy('Weapon_type').count()
arr={}

# in ra các giá trị loại vũ khí và số lượng đếm được
for x in list(xx.collect()):
    arr[''+str(x[0])] = x[1]
    print(x[0],end=': ')
    print(x[1])

names = list(arr.keys())
values = list(arr.values())

fig, axs = plt.subplots(1,1, figsize=(9, 3), sharey=True)
axs.bar(names, values)
# In ra đồ thị Số cuộc khủng bố trong năm
fig.suptitle('Các cuộc khủng bố theo loại vũ khí của Cụm')
```

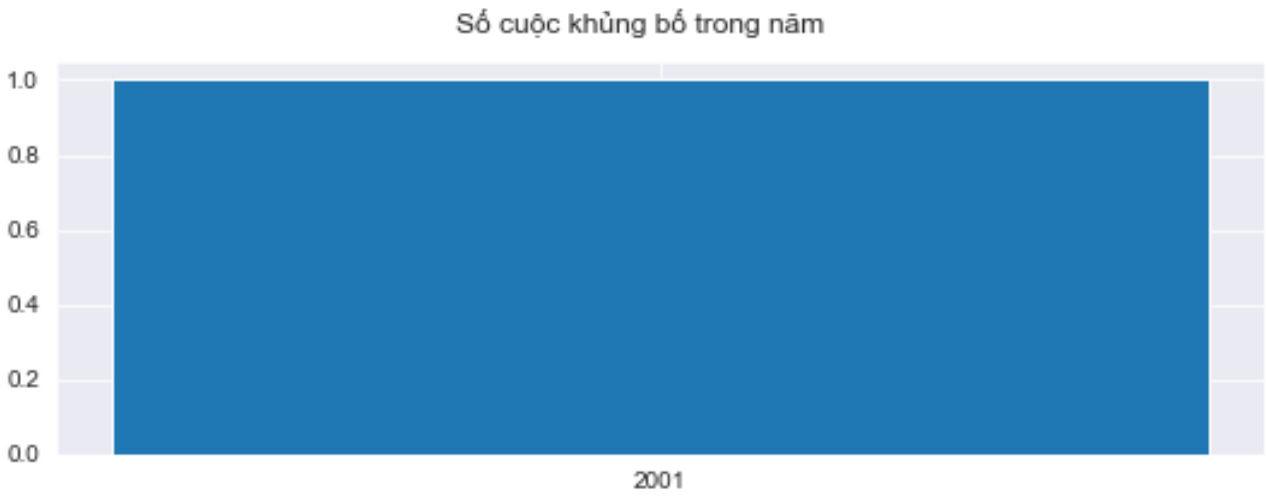
- Hiển thị kết quả

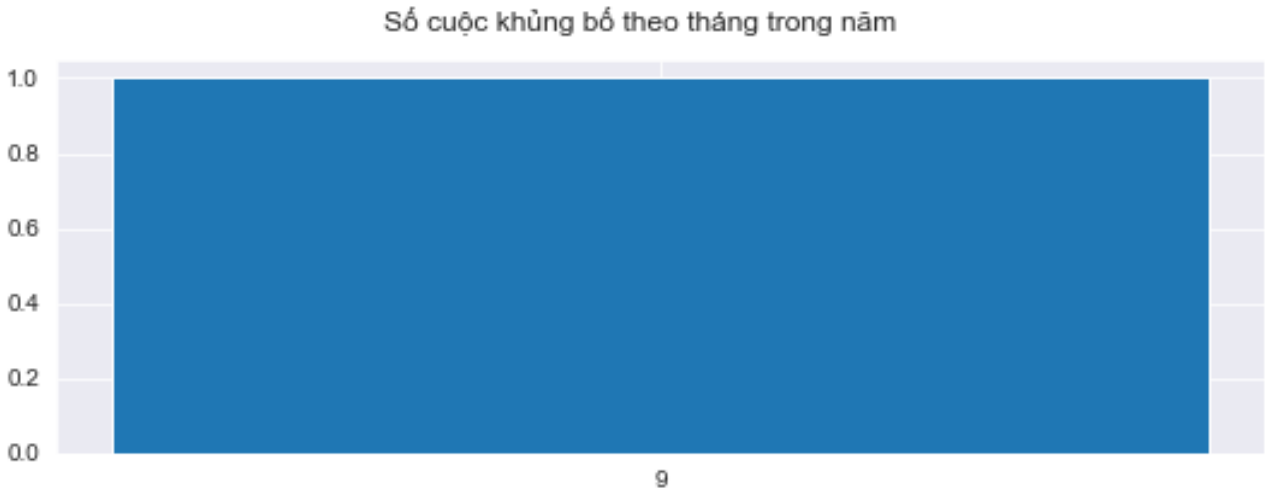
```
+----+-----+---+---------+----------+------+-------+--------------------+--------------------+-----+
|Year|Month|Day|     City|AttackType|Killed|Wounded|         Target_type|         Weapon_type|label|
+----+-----+---+---------+----------+------+-------+--------------------+--------------------+-----+
|2001|    9| 11|Arlington| Hijacking|   190|    106|Government (General)|Vehicle (not to i...|    4|
+----+-----+---+---------+----------+------+-------+--------------------+--------------------+-----+
```

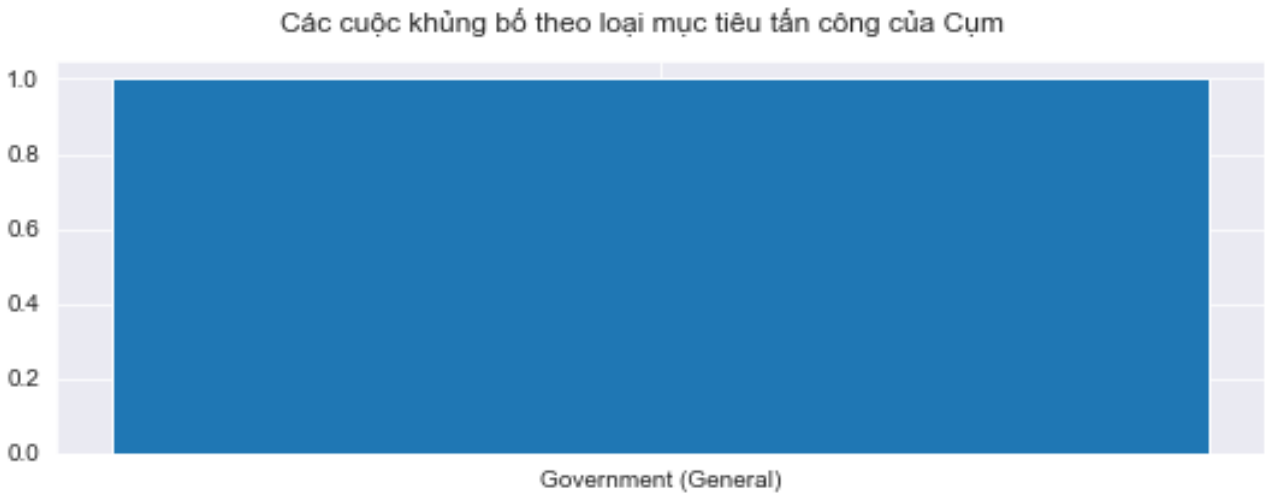
- Hiển thị các thành phần của dữ liệu

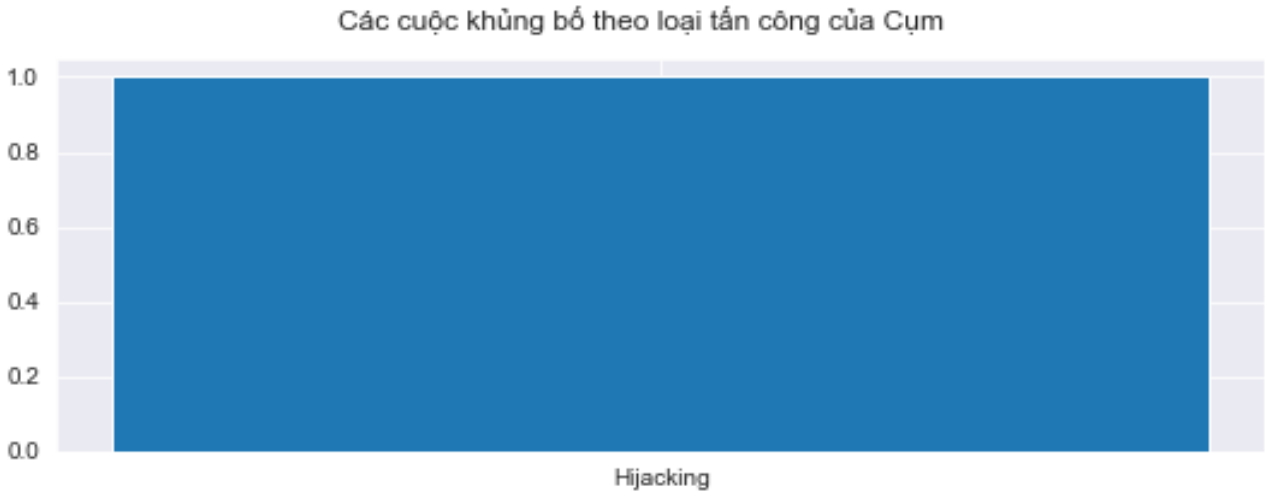
summary	Year	Month	Day	City	AttackType	Killed	Wounded	Target_type	Weapon_type	label
count	1	1	1	1	1	1	1	1	1	1
mean	2001.0	9.0	11.0	null	null	190.0	106.0	null	null	4.0
stddev	null	null	null	null	null	null	null	null	null	null
min	2001	9	11	Arlington	Hijacking	190	106	Government (General)	Vehicle (not to i...	4
max	2001	9	11	Arlington	Hijacking	190	106	Government (General)	Vehicle (not to i...	4

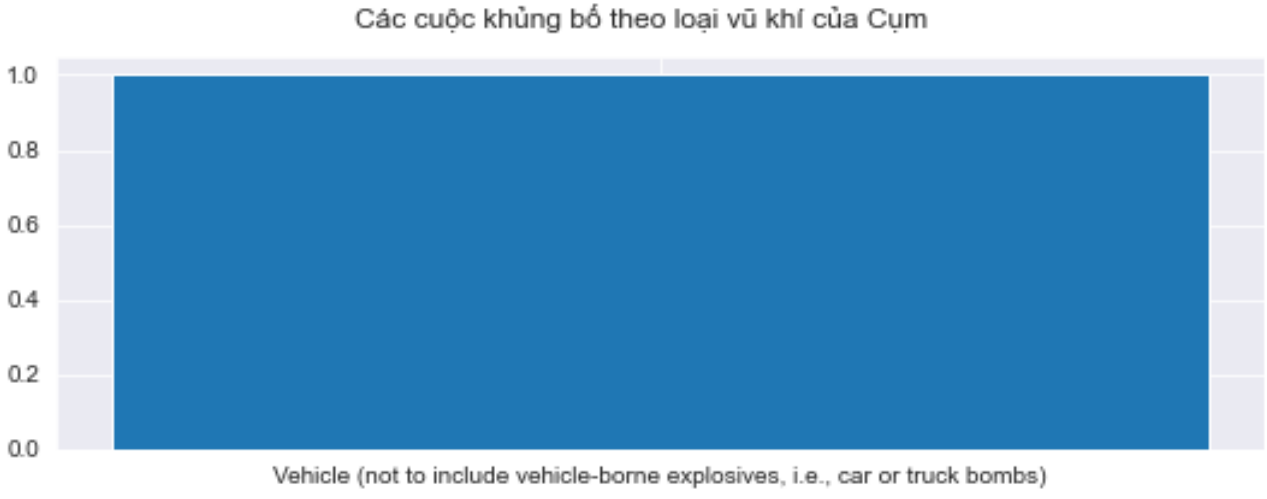
- Kết quả cụm 4






Các cuộc khủng bố theo loại mục tiêu tấn công của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Government (General)


Các cuộc khủng bố theo loại tấn công của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Hijacking


Các cuộc khủng bố theo loại vũ khí của Cụm
1.0
0.8
0.6
0.4
0.2
0.0
Vehicle (not to include vehicle-borne explosives, i.e., car or truck bombs)

IV.Kết quả đạt được

4.1 Phát biểu kết quả

Cụm 0:

- Ý nghĩa phân cụm: những cuộc khủng bố ở cụm 0 chủ yếu là khủng bố theo loại "Assault" với chỉ có hai cuộc khủng bố nhưng có số người chết và bị thương rất cao.
- Phân tích:

  Hai cuộc khủng bố xảy ra vào tháng 9 năm 1984 và tháng 10 năm 2017 đều nhắm vào "Business".

  Cuộc khủng bố ở Las Vegas năm 2017 có lượng người chết và tai nạn cao hơn ở The Dalles năm 1984. Mặc dù ở thành phố The Dalles không có thương vong nhưng số người bị thương cũng rất cao

Year	Month	Day	City	AttackType	Killed	Wounded	Target_type	Weapon_type	label
1984	9	20	The Dalles	Unarmed Assault	0	751	Business	Biological	0
2017	10	1	Las Vegas	Armed Assault	59	851	Business	Firearms	0

Cụm 1:

- Ý nghĩa: Mặc dù năm 1995 chỉ xảy ra 1 cuộc khủng bố nhưng hậu quả để lại là cnhiều người chết và bị thương.
- Phân tích: cuộc khủng bố diễn ra vào tháng 4 năm 1995 và mục tiêu bị nhắm tới là "Goverment". Bọn khủng bố đã sử dụng cách tấn công là "Bombing/Exploision" khiến 168 người thiệt mạng và 650 người bị thương.

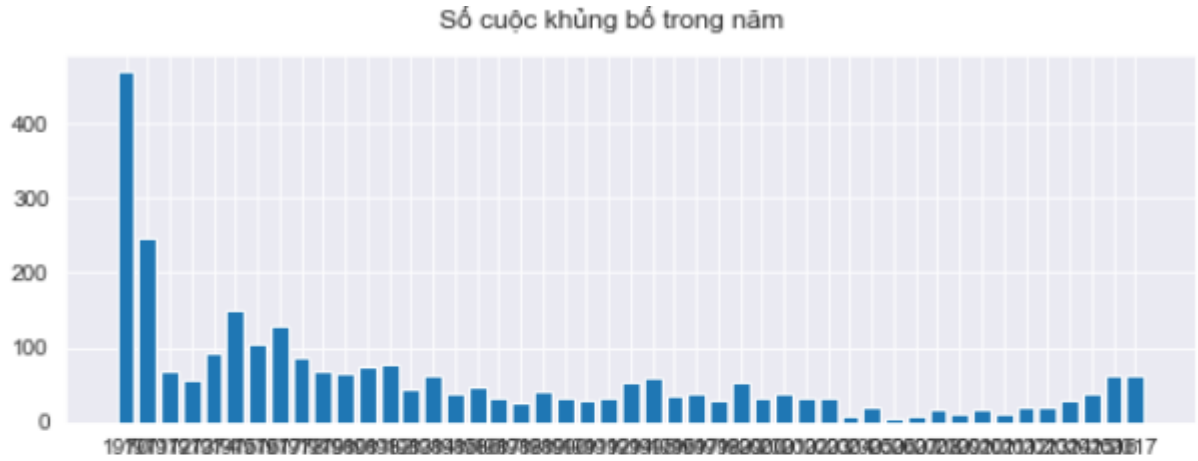
Year	Month	Day	City	AttackType	Killed	Wounded	Target_type	Weapon_type	label
1995	4	19	Oklahoma City	Bombing/Explosion	168	650	Government (General)	Explosives	1

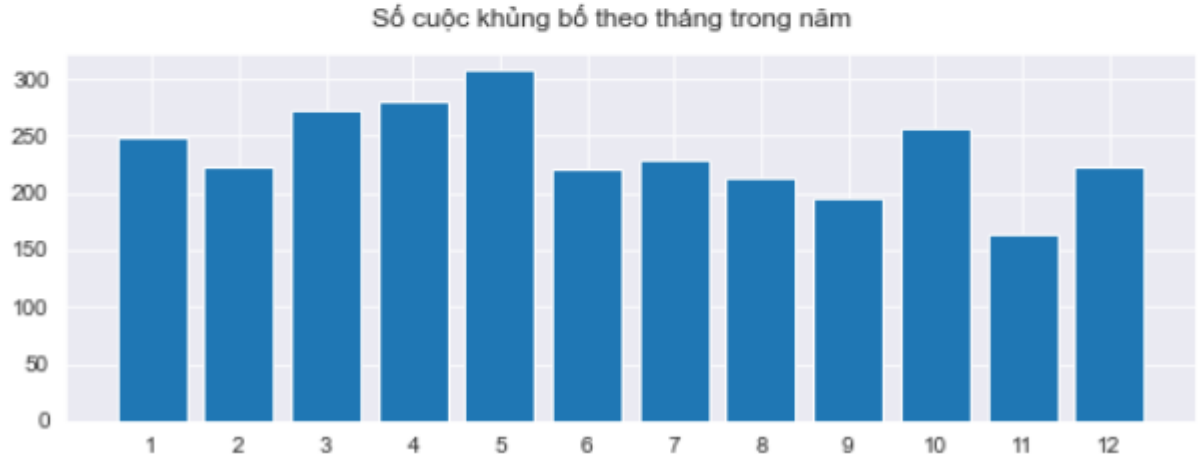
Cụm 2:

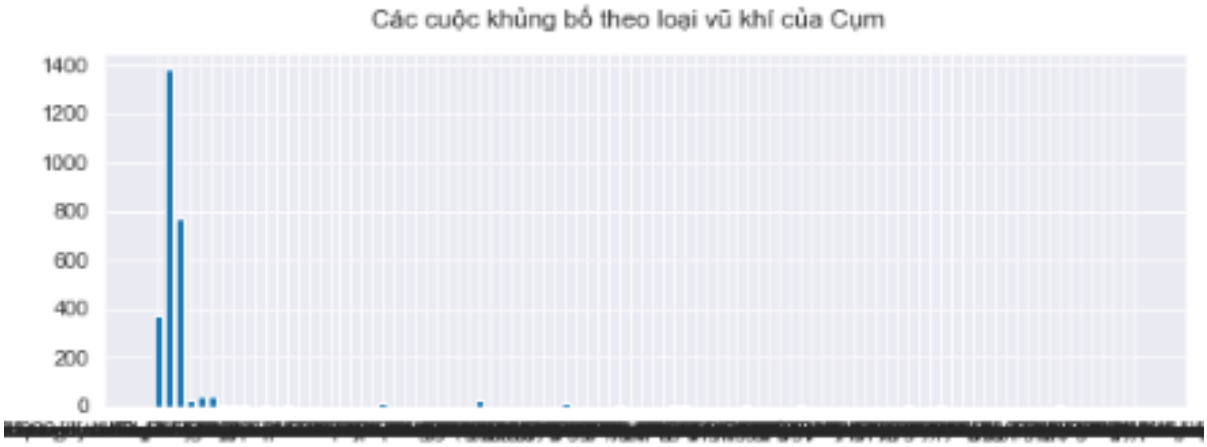
-Ý nghĩa: Đây là cụm có nhiều cuộc khủng bố nhất kéo dài từ năm 1967 đến năm 2017. Cách tấn công chủ yếu là “Bombing/Exploision” .

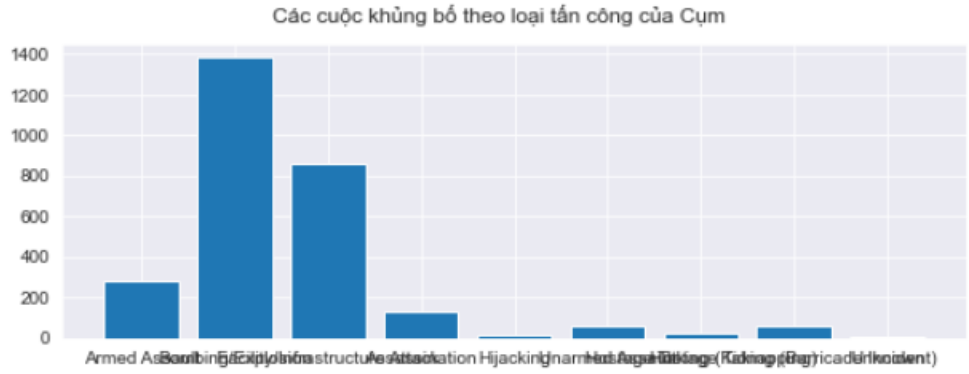
- Phân tích:

  Năm 1970 và 1971 có nhiều cuộc khủng nhất trong tất cả các năm(468 và 247) và các cuộc khủng bố xảy ra đều giữa các tháng.









Cụm 3:

- Ý nghĩa cụm: 2 cuộc khủng bố ở cụm 3 đều có loại tấn công là “Hijacking”. Cả 2 đã gây ra nhiều thương vong và làm rất nhiều người bị thương.
- Phân tích: 2 cuộc khủng bố xảy ra vào tháng 9 năm 2021 và mục tiêu được nhắm đến là các cư dân và tài sản riêng của họ. Cả 2 cuộc khủng bố không có sự chênh lệch về số người tử vong và bị thương.

Year	Month	Day	City	AttackType	Killed	Wounded	Target_type	Weapon_type	label
2001	9	11	New York City	Hijacking	1384	8190	Private Citizens ...	Vehicle (not to i...	3
2001	9	11	New York City	Hijacking	1383	8191	Private Citizens ...	Vehicle (not to i...	3

Cụm 4:

- Ý nghĩa: cuộc khủng bố thuộc cụm 4 có loại tấn công là “Hijacking”. Mặc dù chỉ có một cuộc tấn công nhưng cũng khiến nhiều người tử vong và bị thương.

- Phân tích: cuộc khủng bố xảy ra vào tháng 9 năm 2021 và mục tiêu bị nhắm đến là "Goverment" ở thành phố Arlington.

Year	Month	Day	City	AttackType	Killed	Wounded	Target_type	Weapon_type	label
2001	9	11	Arlington	Hijacking	190	106	Government (General)	Vehicle (not to i...	4

4.2 So sánh và đánh giá

Để so sánh hai thuật toán này, tỷ lệ chi phí so với số ô lặp lại được quan sát cho cả khoảng cách euclidean và manhattan.

Đối với kmeans, điểm bắt đầu của chi phí thấp hơn kmeans ++. Tuy nhiên, giá trị điểm cuối của hàm chi phí cao hơn kmeans ++.

Điều này cho thấy đối với kmeans ++ mặc dù nó bắt đầu ở hàm chi phí cao hơn, nhưng sự hội tụ sẽ sớm đạt được với giá trị nhỏ nhất của hàm chi phí so với kmeans.

Điều này chứng minh rằng kmeans ++ có thể là thuật toán phân cụm hiệu quả nhất trong số hai thuật toán này.

Euclid Distance	Manhattan Distance
Giá trị chi phí nhiều hơn	Giá trị chi phí thấp hơn
Chi phí giảm thiểu sau sau khi thực hiện ít hơn (33,4%)	Chi phí giảm thiểu sau sau khi thực hiện ít hơn (40.34%)

Nhận xét:

- Thực hiện chạy thuật toán với khoảng cách Manhattan giảm thiểu được nhiều chi phí thực hiện hơn và cho hiệu quả cao hơn.

V.Kết luận

5.1 Ưu điểm

- Ứng dụng được bài toán phân cụm và giải quyết được vấn đề đặt ra.
- Áp dụng được các phương pháp đánh giá và so sánh mô hình.

5.2 Khuyết điểm

Vì thời gian để nghiên cứu và hiện thực đề tài còn giới hạn do đó còn tồn tại một số hạn chế, như sau:

- Chưa ứng dụng thêm các Distance Measure khác vào xử lý bài toán.
- Chưa nghiên cứu tìm thêm một số thuật toán khác tối ưu hơn.

5.3 Hướng phát triển

- Ứng dụng kết quả đề tài vào các dự án theo hướng phân tích mạng xã hội.
- Sử dụng nhiều tập dataset để đánh giá hiệu năng của mô hình.
- Nghiên cứu sử dụng thêmnhiều thuật toán khác để đánh giá và so sánh từ đó tối ưu kết quả đạt được.
- Tiếp tục tối ưu các thuật toán sử dụng để đưa ra kết quả chính xác và thời gian nhanh hơn.
- Kết hợp các thuật toán lại thành một mô hình hoàn chỉnh.

VI.Tài liệu tham khảo

1. Link Dataset “Terrorism Data 1970to2017” - Terrorism Data 1970to2017 | Kaggle

2. “K-Means Clustering” - https://machinelearningcoban.com/2017/01/01/kmeans/

3. “K-means clustering - Wikipedia” - https://en.wikipedia.org/wiki/K-means_clustering

4. “9 phép đo khoảng cách trong khoa học dữ liệu” - https://ichi.pro/vi/9-phep-do-khoang-cach-trong-khoa-hoc-du-lieu-159983401462266

5. “Distance Measure trong Machine Learning” - https://viblo.asia/p/distance-measure-trong-machine-learning-ByEZkopYZQ0

HẾT.