**Лабораторна робота № 4.**

**Кластеризація. Метод k-mean,   
пошук оптимальних параметрів**

**1. Мета роботи**

Метою лабораторної роботи є отримання практичних навичок з виконання кластеризації даних, використовуючи метод k-mean та інші з пакету sklearn.

**2. Контекст**HELP International — це міжнародна гуманітарна неурядова організація, яка займається боротьбою з бідністю та наданням людям відсталих країн базових зручностей і допомоги під час стихійних лих і стихійних лих. Час від часу він запускає багато оперативних проектів, а також проводить адвокаційні заходи для підвищення обізнаності, а також для цілей фінансування.

Після останніх програм фінансування їм вдалося зібрати близько 10 мільйонів доларів. Тепер генеральний директор НУО має вирішити, як стратегічно та ефективно використати ці гроші. Важливі проблеми, які виникають під час прийняття цього рішення, здебільшого пов’язані з вибором країн, які найбільше потребують допомоги.

І тут ви виступаєте як аналітик даних. Ваша  робота полягає в тому, щоб класифікувати країни за певними соціально-економічними факторами та факторами охорони здоров’я, які визначають загальний розвиток країни. Потім вам потрібно запропонувати країни, на яких генеральний директор має зосередитися найбільше.

Проаналізуйте кластери та визначте ті, які гостро потребують допомоги. Ми можемо проаналізувати кластери, порівнюючи, як ці три змінні - [gdpp, child\_mort і дохід] змінюються для кожного кластера країн, щоб розпізнати та відрізнити кластери розвинутих країн від кластерів слаборозвинених країн.

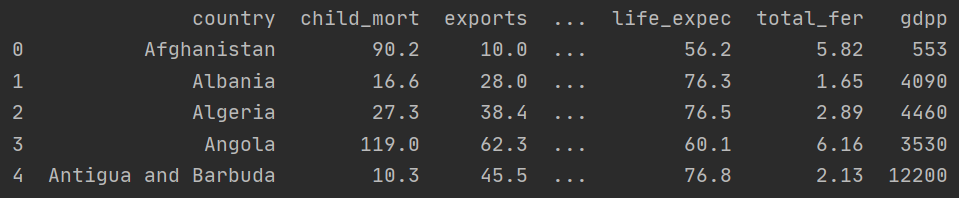
Крім того, нам потрібно виконати візуалізацію сформованих кластерів. Ми можемо зробити це, вибравши будь-які дві з трьох змінних, згаданих вище, на осях X-Y і побудувавши діаграму розсіювання для всіх країн і диференціюючи кластери. Переконайтеся, що ми створюємо візуалізацію для всіх трьох пар.

K-mean слід повідомити генеральному директору у вигляді summary. Переконайтеся, що ви повідомляєте принаймні про 5 країн, які найбільше потребують допомоги в результаті аналізу, який ви виконуєте. Включіть в саммарі необхідні графіки та текстову інформацію.

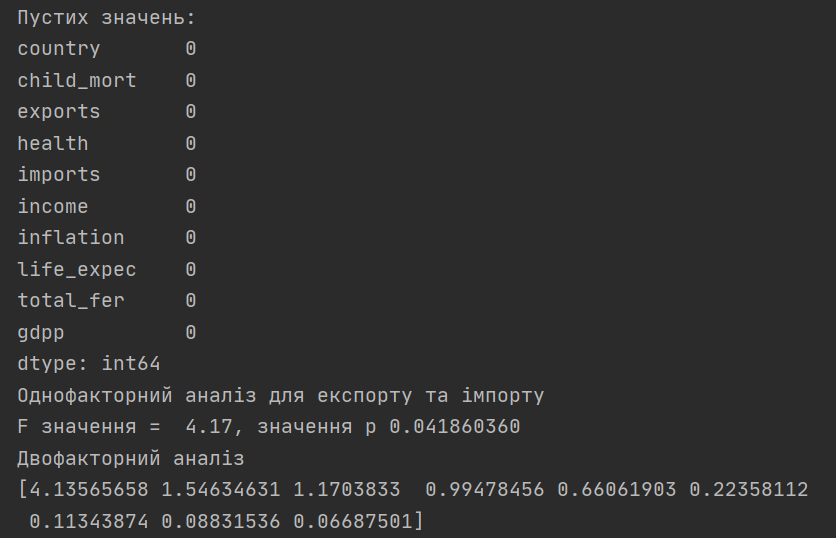
**2. Завдання до виконання**

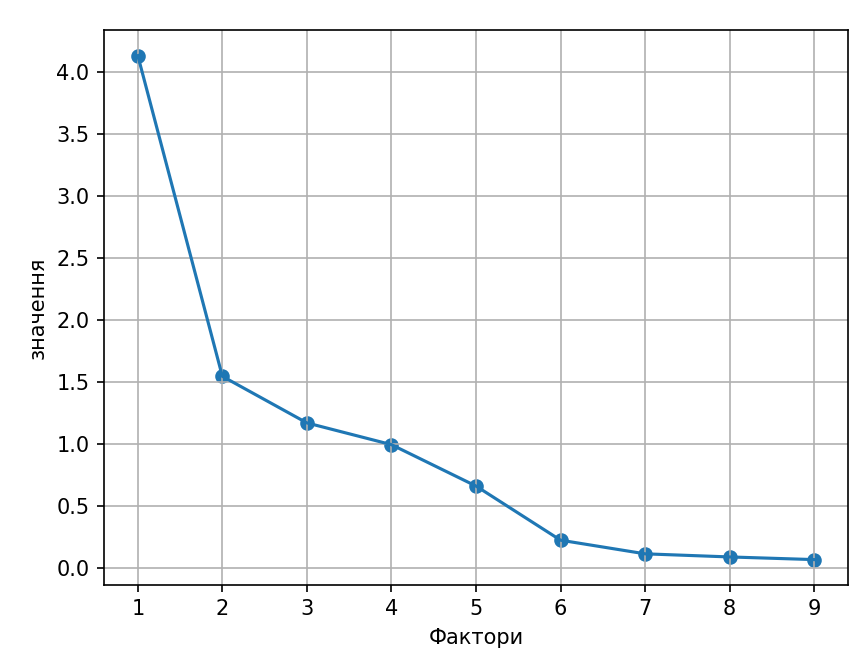
* Прочитайте надану в **Супровідних матеріалах** документацію та вивчить надані приклади:  
    
  <https://drive.google.com/drive/folders/1n9t1pQkMmfm-4oy67v75vuFLQhp_o7_h?hl=ru>
* Завантажити файл з даними у папку проекту

<https://drive.google.com/drive/folders/1dPmiiTn5JinCgEo5TrS8nnsPQW8QdFjs?hl=ru>

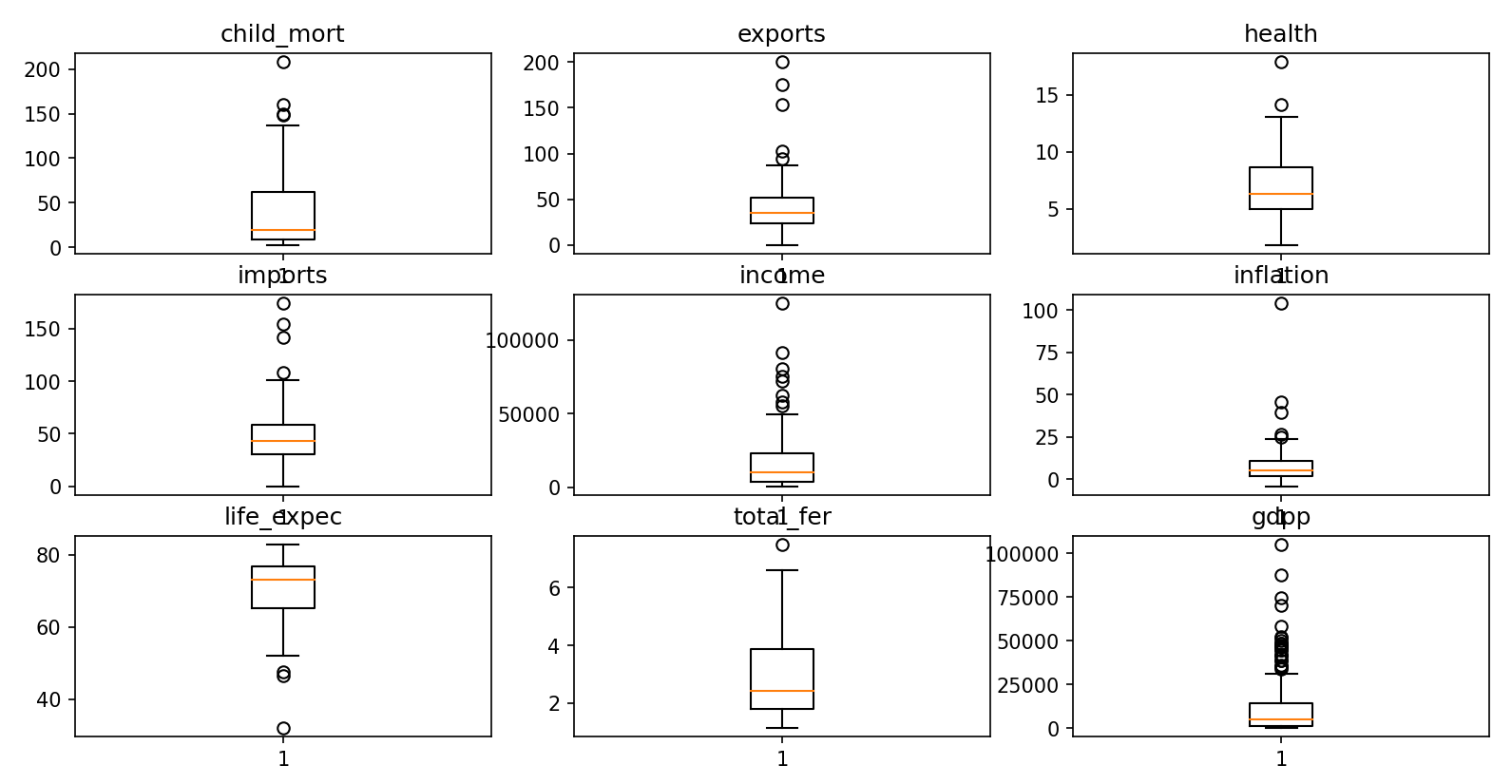


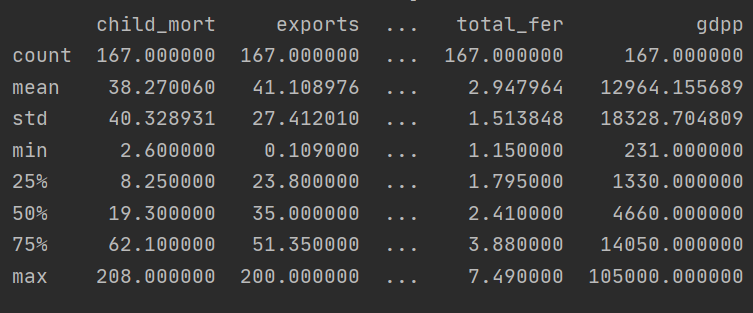
* Почніть із необхідної перевірки даних, очищення даних та EDA, зробіть однофакторний аналіз, двофакторний аналіз тощо. Візуалізуйте закономірності.





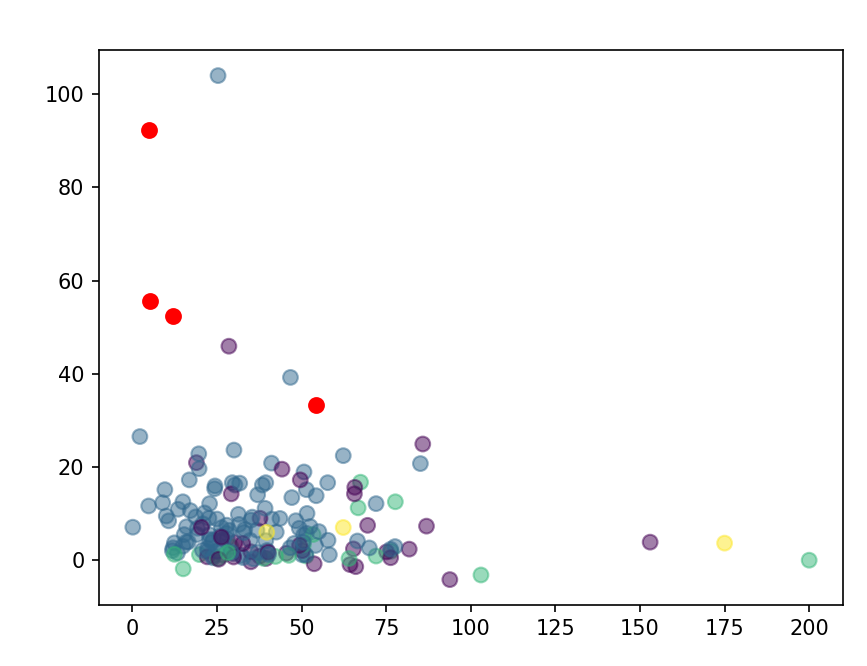
* Ви повинні виконати аналіз викидів для набору даних. Вибрати, залишити їх чи видалити залежно від отриманих результатів.

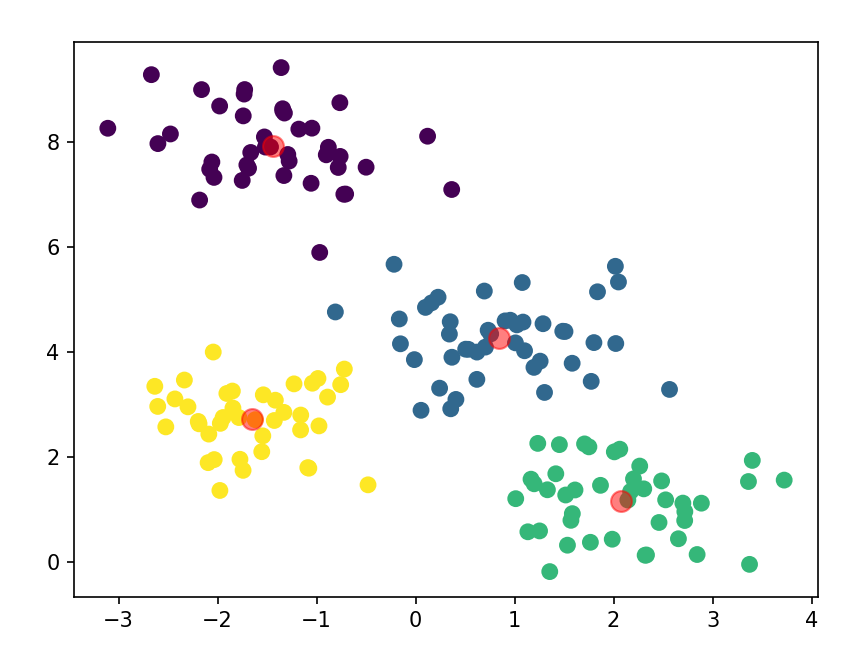




Залишаємо викиди, бо їх небагато

* Провести кластеризацію, використовуючи k-mean методологію з прикладів та документації.

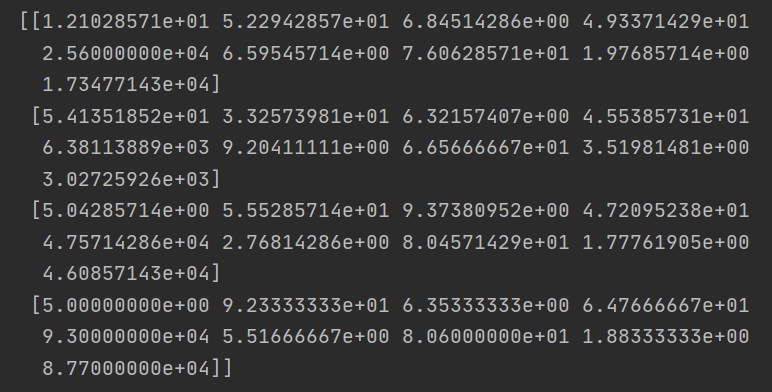


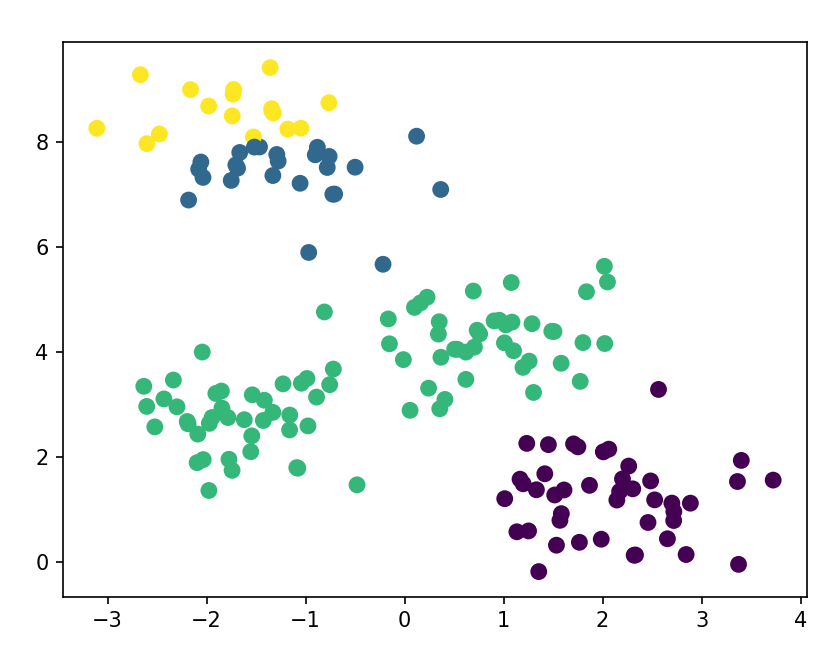


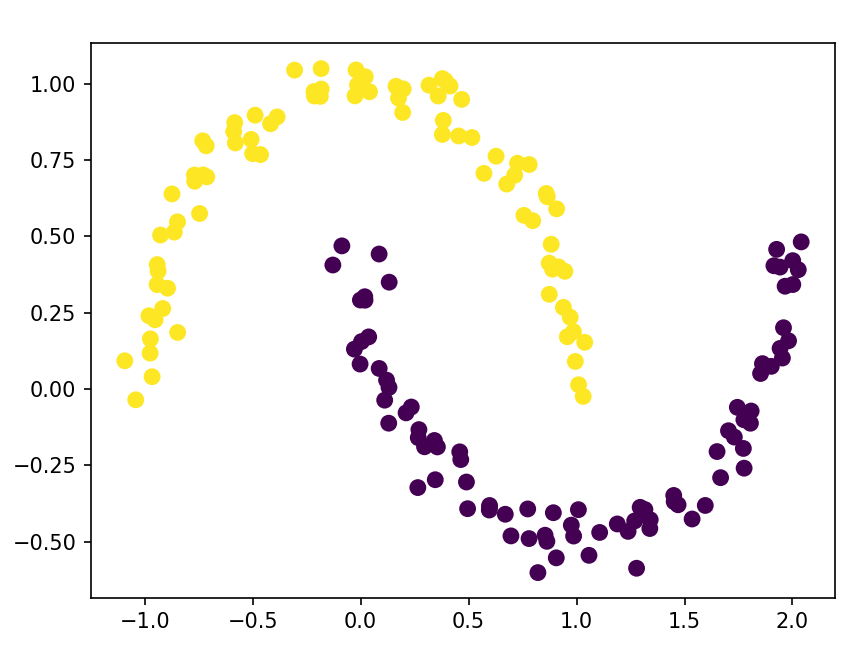
* Спробуйте оптимізувати k, використовуючи kneed із прикладу.

Було зроблено при кластерізіції

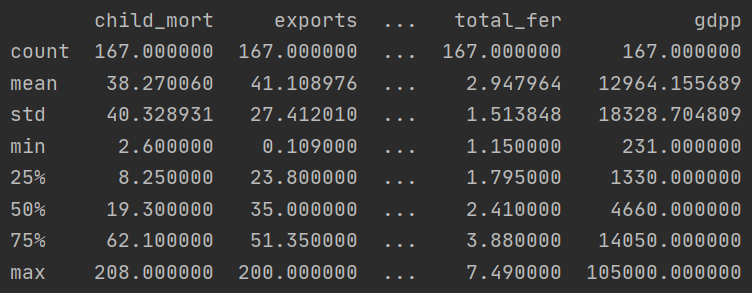
* Візуалізуйте результати кластеризації та оптимізації гіпер-параметрів.







* Підготуйте summary, використовуючи лише необхідні матеріали.



Лістинг програми:

import pandas as pd  
from factor\_analyzer import FactorAnalyzer  
import matplotlib.pyplot as plt  
from scipy import stats  
from sklearn.datasets import make\_blobs, make\_moons  
from sklearn.metrics import pairwise\_distances\_argmin  
from sklearn.cluster import SpectralClustering, KMeans  
import numpy as np  
  
  
df = pd.read\_csv('Country-data.csv')  
print(df.head())  
  
print('Пустих значень:')  
print(df.isnull().sum())  
del df['country']  
  
F, p = stats.f\_oneway(df['exports'], df['imports'])  
F\_test = stats.f.ppf((1-0.05), 4, 15)  
print("Однофакторний аналіз для експорту та імпорту")  
print('F значення = % .2F, значення p% .9f'%(F, p))  
  
print("Двофакторний аналіз")  
fa = FactorAnalyzer()  
fa.fit(df)  
  
ev, v = fa.get\_eigenvalues()  
print(ev)  
plt.scatter(range(1,df.shape[1]+1),ev)  
plt.plot(range(1,df.shape[1]+1),ev)  
plt.xlabel('Фактори')  
plt.ylabel('значення')  
plt.grid()  
plt.show()  
  
print(df.describe())  
  
plt.subplot(3,3,1)  
plt.title('child\_mort')  
plt.boxplot(df['child\_mort'])  
  
plt.subplot(3,3,2)  
plt.title('exports')  
plt.boxplot(df['exports'])  
  
plt.subplot(3,3,3)  
plt.title('health')  
plt.boxplot(df['health'])  
  
plt.subplot(3,3,4)  
plt.title('imports')  
plt.boxplot(df['imports'])  
  
plt.subplot(3,3,5)  
plt.title('income')  
plt.boxplot(df['income'])  
  
plt.subplot(3,3,6)  
plt.title('inflation')  
plt.boxplot(df['inflation'])  
  
plt.subplot(3,3,7)  
plt.title('life\_expec')  
plt.boxplot(df['life\_expec'])  
  
plt.subplot(3,3,8)  
plt.title('total\_fer')  
plt.boxplot(df['total\_fer'])  
  
plt.subplot(3,3,9)  
plt.title('gdpp')  
plt.boxplot(df['gdpp'])  
plt.show()  
  
X = pd.DataFrame(df, columns=['exports', 'inflation'])  
  
kmeans = KMeans(n\_clusters=4).fit(df)  
centroids = kmeans.cluster\_centers\_  
print(centroids)  
X.to\_csv('kmeans.csv')  
  
plt.scatter(df['exports'], df['inflation'], c= kmeans.labels\_.astype(float), s=50, alpha=0.5)  
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=50)  
plt.show()  
  
X, y\_true = make\_blobs(n\_samples=len(df['exports']), centers=4, cluster\_std=0.70, random\_state=0)  
  
kmeans = KMeans(n\_clusters=4)  
kmeans.fit(X)  
y\_kmeans = kmeans.predict(X)  
  
plt.scatter(X[:, 0], X[:, 1], c=y\_kmeans, s=50, cmap='viridis')  
  
centers = kmeans.cluster\_centers\_  
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=100, alpha=0.5)  
plt.show()  
  
  
def find\_clusters(X, n\_clusters, rseed=2):  
 # 1. Randomly choose clusters  
 rng = np.random.RandomState(rseed)  
 i = rng.permutation(X.shape[0])[:n\_clusters]  
 centers = X[i]  
  
 while True:  
 # 2a. Assign labels based on closest center  
 labels = pairwise\_distances\_argmin(X, centers)  
  
 # 2b. Find new centers from means of points  
 new\_centers = np.array([X[labels == i].mean(0)  
 for i in range(n\_clusters)])  
  
 # 2c. Check for convergence  
 if np.all(centers == new\_centers):  
 break  
 centers = new\_centers  
  
 return centers, labels  
  
  
centers, labels = find\_clusters(X, 4)  
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')  
plt.show()  
  
X, y = make\_moons(len(df['exports']), noise=.05, random\_state=0)  
model = SpectralClustering(n\_clusters=2, affinity='nearest\_neighbors',  
 assign\_labels='kmeans')  
labels = model.fit\_predict(X)  
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')  
plt.show()  
  
print(df.describe())

**3. Необхідні пакети**

pandas, numpy, seaborn, jupyter, kneed, scipy

**4. Супровідні матеріали**

<https://kneed.readthedocs.io/en/stable/index.html>

<https://scikit-learn.org/stable/modules/clustering.html#k-means>

<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

<https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html>

<https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html>