НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря Сікорського»
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

ЗВІТ

про виконання комп'ютерного практикуму № 4

з дисципліни «Інтелектуальний аналіз даних»

Виконав:

Студент III курсу

Групи КА-13

Приймак Є.О.

Варіант № 25

Перевірила:

Недашківська Н. І.

Київ – 2023 рік

Тема: Побудова та оцінювання якості моделей кластеризації в Scikit-Learn Python

Завдання: **25**. Агломеративний алгоритм AgglomerativeClustering. Дослідити методи розрахунку відстані між кластерами: ward, single, average, complete.

Метрики якості: Estimated number of clusters, Adjusted Rand Index, V-measure. Побудувати матриці відстаней між кластерами, використовуючи metrics.pairwise_distances.

Чи є розбиття стабільним після вилучення окремих об’єктів?

Початкові дані:

(а) sklearn.datasets.make_moons

(б) sklearn.datasets.load_iris

Виконання:

1. Представити початкові дані графічно.

2. Побудувати модель кластеризації згідно з варіантом.

3. Виконати кластеризацію даних на основі моделі.

4. Представити розбиття на кластери графічно, наприклад, різними кольорами.

5. Розрахувати час кластеризації. Оцінити швидкодію методу на надвеликих наборах даних (наприклад, при збільшенні кількості точок даних до ста тисяч і більше).

6. Побудувати кілька альтернативних моделей:

• використати різні функції відстані в алгоритмах, де це можна зробити,

7. Для кожної альтернативної моделі розрахувати метрики якості кластеризації, що реалізовані в модулі sklearn.metrics:

• Estimated Number of Clusters.

• Adjusted Rand Index.

• V-measure.

8. Виконати аналіз результатів кластеризації одним з неформальних методів (тільки методом згідно з варіантом):

• чи є розбиття стабільним після видалення окремих об’єктів,

9. Вищенаведені пункти виконати для заданих двох наборів даних різної форми.

10. Зробити висновки про якість роботи моделей на досліджених даних та про швидкодію методу.

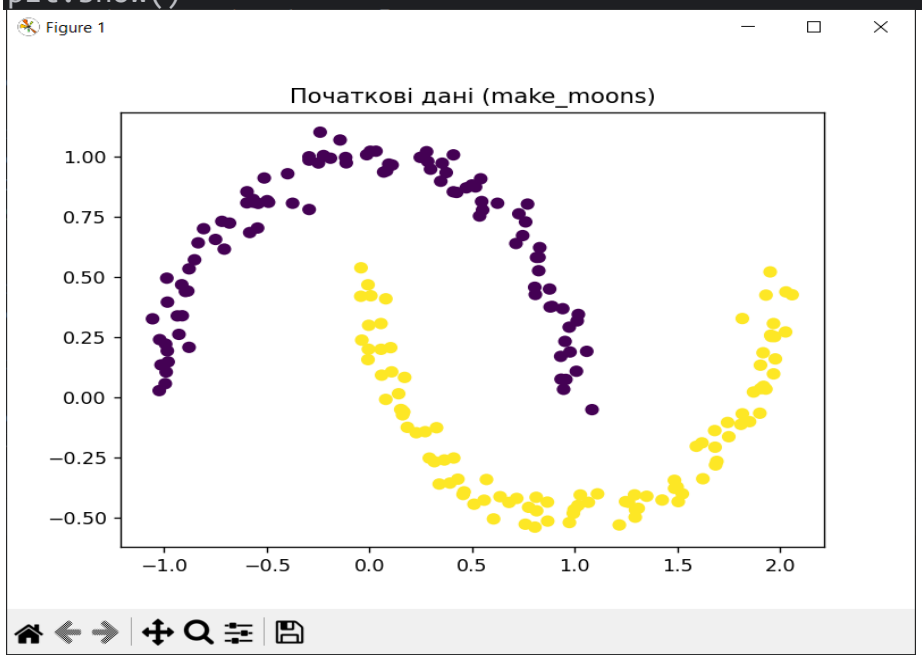
11. Оцінити результати кластеризації на основі метрик якості та на основі неформальних методів. У кожному варіанті задано два набори даних.Спробувати підібрати найкращу модель кластеризації для кожного набору даних.

Хід роботи:

Набір (а) (make_moons):

1.Представимо початкові дані графічно:

```
from sklearn import datasets
from sklearn.cluster import AgglomerativeClustering
from sklearn.metrics import adjusted_rand_score, v_measure_score
from sklearn.metrics.pairwise import pairwise_distances
import time
import numpy as np
import matplotlib.pyplot as plt

# Завдання (1а): Завантаження та візуалізація даних
X, y = datasets.make_moons(n_samples=200, noise=0.05)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis')
plt.title("Початкові дані (make_moons)")
plt.show()
```



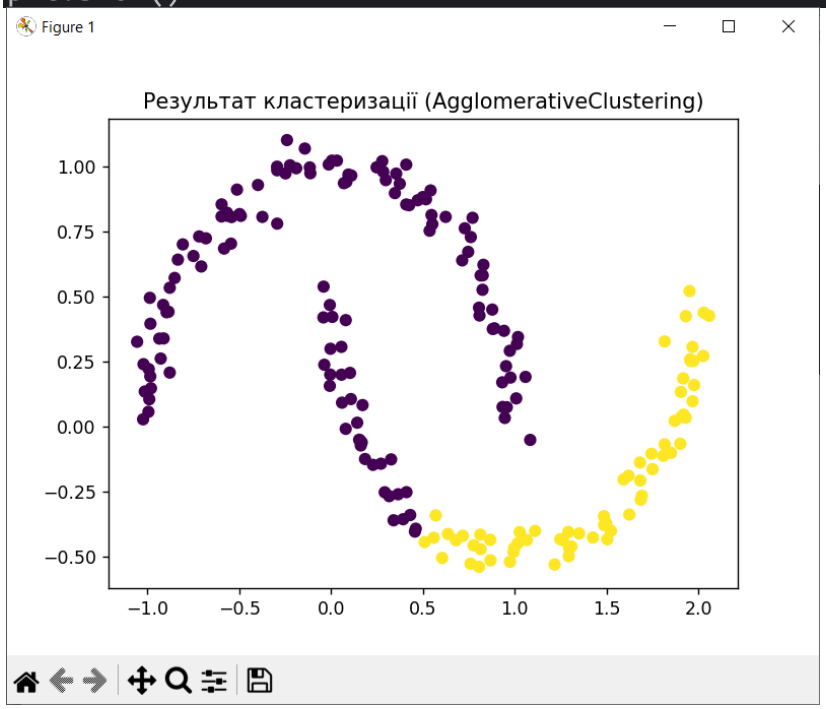
2+3. Побудуємо модель кластеризації, зафіксуємо час роботи моделі:

```
# Завдання (2а): Створення моделі AgglomerativeClustering
model = AgglomerativeClustering(n_clusters=2)
# Завдання (3а): Кластеризація даних
start_time = time.time()
y_pred = model.fit_predict(X)
elapsed_time = time.time() - start_time
print(f"Час кластеризації: {elapsed_time:.2f} сек")
```

```
Час кластеризації: 0.02 сек
```

4.Візуалізуємо отриману кластеризацію (за замовчанням linkage = ‘ward’):

```
# Завдання (4а): Візуалізація розбиття на кластери
plt.scatter(X[:, 0], X[:, 1], c=y_pred, cmap='viridis')
plt.title("Результат кластеризації (AgglomerativeClustering)")
plt.show()
```

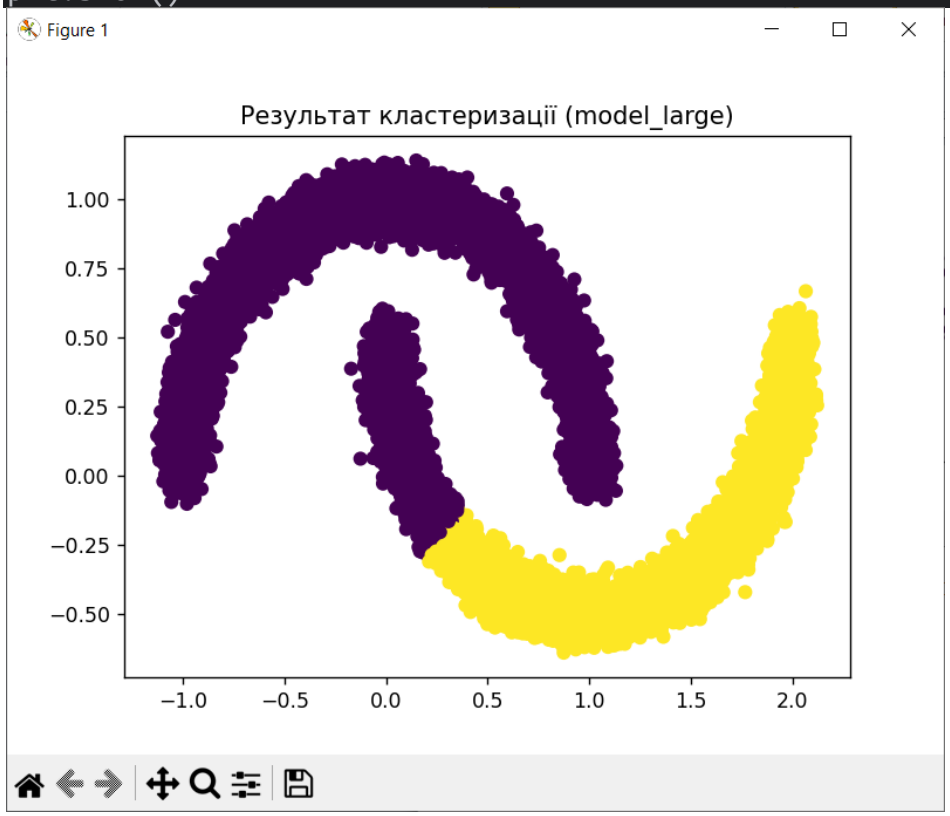


5. Оцінимо швидкодію методу на надвеликих наборах даних, збільшимо кількість об’єктів до 20000:

```
# Завдання (5а): Оцінка швидкості та якості для великого набору даних
make_moons
X_large, y_large = datasets.make_moons(n_samples=20000, noise=0.05)
model_large = AgglomerativeClustering(n_clusters=2)
start_time = time.time()
y_pred_large = model_large.fit_predict(X_large)
elapsed_time_large = time.time() - start_time
print(f"Час кластеризації для великого набору даних:
{elapsed_time_large:.2f} сек")
```

```
Час кластеризації для великого набору даних: 35.46 сек
```

```
# Візуалізація результатів для model_large на наборі даних make_moons
plt.scatter(X_large[:, 0], X_large[:, 1], c=y_pred_large,
cmap='viridis')
plt.title("Результат кластеризації (model_large)")
plt.show()
```



6+7. В циклі побудуємо альтернативні моделі кластеризації (використовуючи різні функції відстані), візуалізуємо їх, розрахуємо задані метрики якості моделей.

Нам знадобляться функції:

```
def estimated_num_clusters(model, X):
    return model.n_clusters_

# Функція для отримання масиву кластерів
def get_clusters(X, labels):
    unique_labels = np.unique(labels)
    clusters = []
    for label in unique_labels:
        cluster = X[labels == label]
        clusters.append(cluster)
    return clusters


# Функція для обчислення матриці відстаней між кластерами
```

```
def cluster_distance_matrix(clusters):
    n_clusters = len(clusters)
    dist_matrix = np.zeros((n_clusters, n_clusters))
    for i in range(n_clusters):
        for j in range(i+1, n_clusters):
            dist_matrix[i, j] = pairwise_distances(clusters[i],
clusters[j]).mean()
            dist_matrix[j, i] = dist_matrix[i, j]
    return dist_matrix
```

Альтернативи кластеризації, їх візуалізація та метрики:

```
# Завдання (6+7 (а)): Побудова альтернативних моделей для make_moons
models_alternative = []
linkage_methods = ['ward', 'single', 'average', 'complete']

for method in linkage_methods:
    model_alternative = AgglomerativeClustering(n_clusters=2,
linkage=method)
    models_alternative.append(model_alternative)

# Візуалізація альтернативних моделей та метрик + матриця
pairwise_distances для кластерів
for i, model in enumerate(models_alternative):
    y_pred_alternative = model.fit_predict(X)
    clusters = get_clusters(X, y_pred_alternative)
    plt.scatter(X[:, 0], X[:, 1], c=y_pred_alternative,
cmap='viridis')
    plt.title(f"Результат кластеризації {linkage_methods[i]}
(make_moons)")
    plt.show()
    estimated_clusters = estimated_num_clusters(model, X)
    ari = adjusted_rand_score(y, y_pred_alternative)
    v_measure = v_measure_score(y, y_pred_alternative)
    pairwise_distances_matrix = cluster_distance_matrix(clusters)
    print(f"\nEstimated number of clusters {linkage_methods[i]}
(make_moons): {estimated_clusters}")
    print(f"Adjusted Rand Index {linkage_methods[i]} (make_moons):
{ari:.2f}")
    print(f"V-measure {linkage_methods[i]} (make_moons):
{v_measure:.2f}")
    print(f"\nМатриця відстаней кластеризації {linkage_methods[i]}
(make_moons):\n")
    print(pairwise_distances_matrix)
```

Отримуємо:

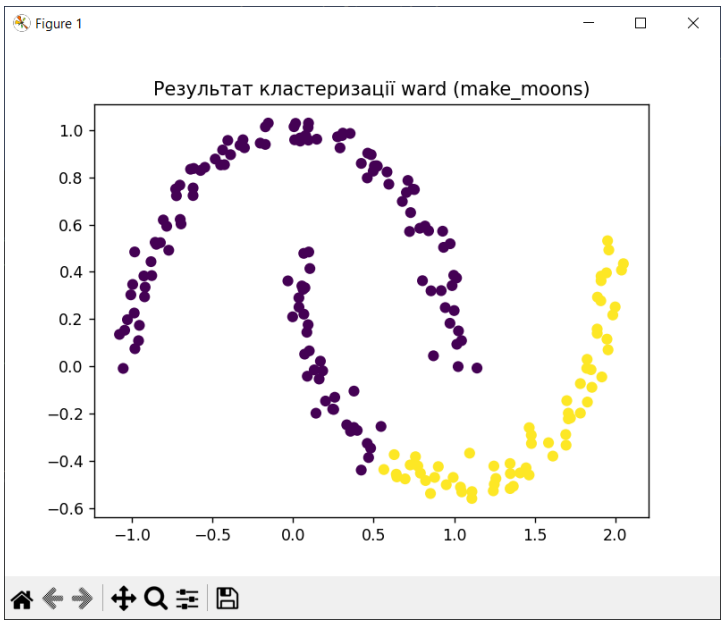


Figure 1
Результат кластеризації ward (make_moons)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

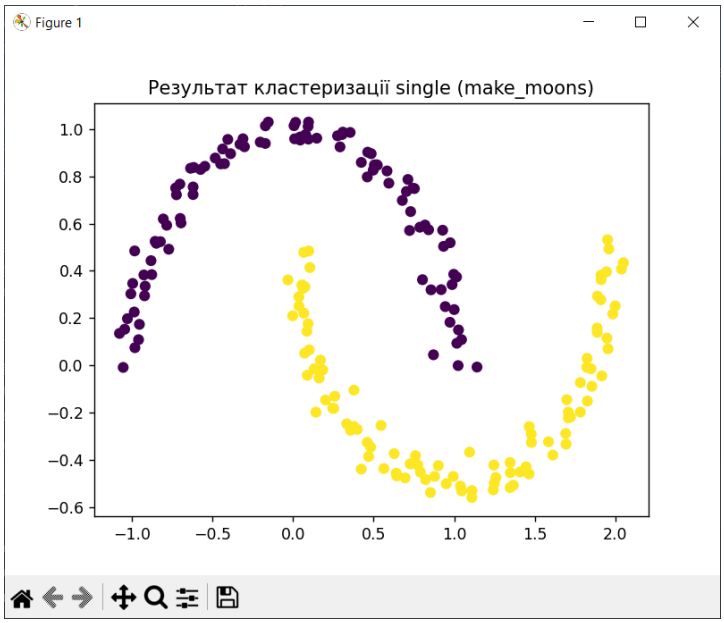


Figure 1
Результат кластеризації single (make_moons)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

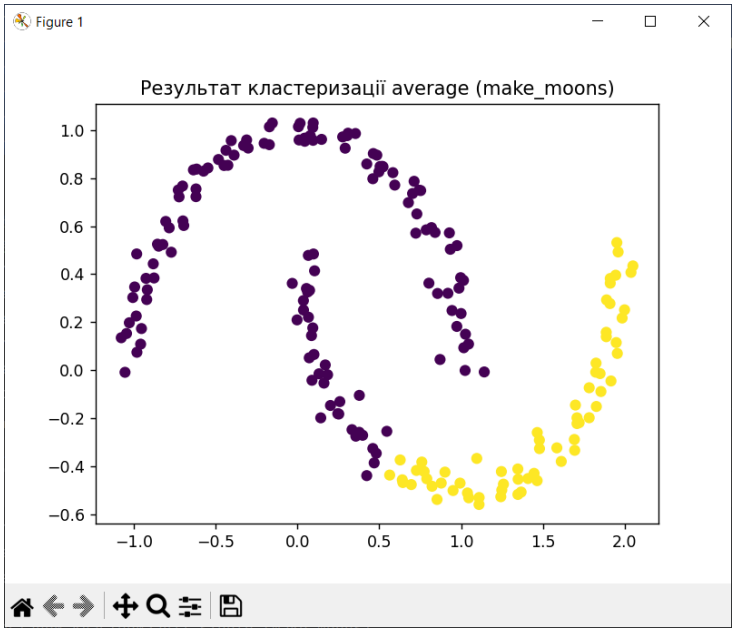
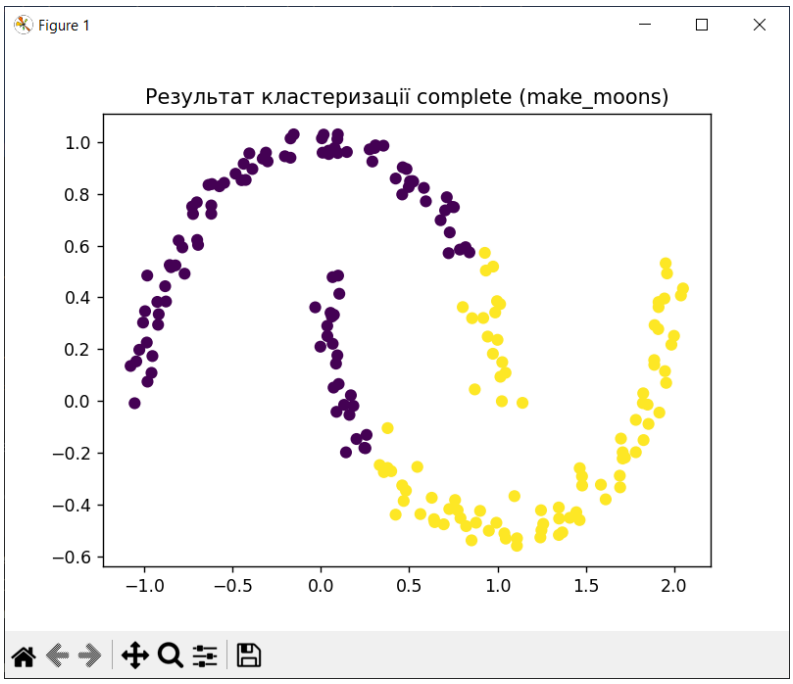
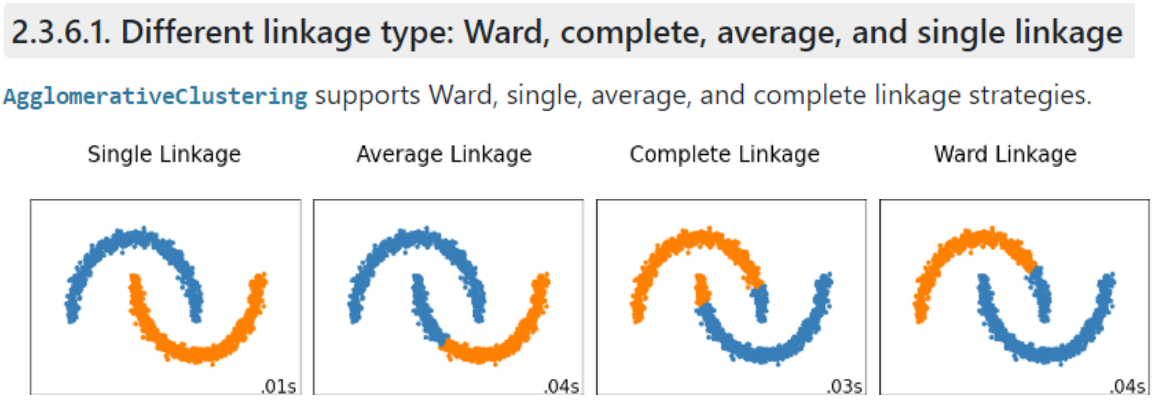


Figure 1
Результат кластеризації average (make_moons)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0



Можна порівняти з еталонними результатами методів для схожого набору з офіційної документації sklearn.AgglomerativeClustering:



Отримані метрики та матриці відстаней між кластерами (за допомогою metrics.pairwise_distances)

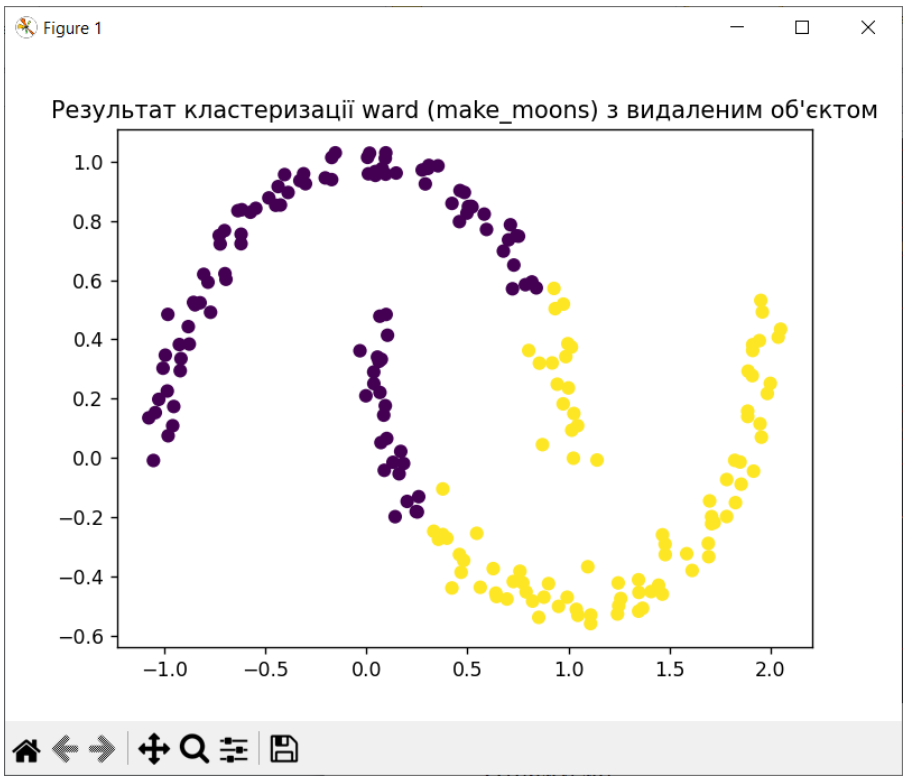
```
Estimated number of clusters ward (make_moons): 2
Adjusted Rand Index ward (make_moons): 0.42
V-measure ward (make_moons): 0.46

Матриця відстаней кластеризації ward (make_moons):

[[0.         1.68409377]
 [1.68409377 0.        ]]

Estimated number of clusters single (make_moons): 2
Adjusted Rand Index single (make_moons): 1.00
V-measure single (make_moons): 1.00

Матриця відстаней кластеризації single (make_moons):

[[0.         1.53565894]
 [1.53565894 0.        ]]
```

```
Estimated number of clusters average (make_moons): 2
Adjusted Rand Index average (make_moons): 0.42
V-measure average (make_moons): 0.46

Матриця відстаней кластеризації average (make_moons):

[[0.         1.68409377]
 [1.68409377 0.        ]]

Estimated number of clusters complete (make_moons): 2
Adjusted Rand Index complete (make_moons): 0.32
V-measure complete (make_moons): 0.25

Матриця відстаней кластеризації complete (make_moons):

[[0.         1.67853726]
 [1.67853726 0.        ]]
```

8. Виконаємо аналіз результатів кластеризації одним з неформальних методів за варіантом (видаливши окремий елемент):

```
# Завдання (8а): Аналіз стабільності розбиття
# Видалення одного об'єкта та повторна кластеризація
idx_to_remove = np.random.randint(0, len(X))
X_removed = np.delete(X, idx_to_remove, axis=0)
y_removed = np.delete(y, idx_to_remove)
y_pred_removed = model.fit_predict(X_removed)
clusters_r = get_clusters(X_removed, y_pred_removed)
plt.scatter(X_removed[:, 0], X_removed[:, 1], c=y_pred_removed,
cmap='viridis')
plt.title("Результат кластеризації ward (make_moons) з видаленим
об'єктом")
plt.show()
pairwise_distances_matrix_r = cluster_distance_matrix(clusters_r)
estimated_clusters_r = estimated_num_clusters(model, X_removed)
ari_r = adjusted_rand_score(y_removed, y_pred_removed)
v_measure_r = v_measure_score(y_removed, y_pred_removed)
print(f"\nEstimated number of clusters після видалення об'єкта
(make_moons): {estimated_clusters_r}")
print(f"Adjusted Rand Index після видалення об'єкта (make_moons):
{ari_r:.2f}")
print(f"V-measure після видалення об'єкта (make_moons):
{v_measure_r:.2f}")
print(f"\nМатриця відстаней кластеризації 'ward' після видалення
об'єкта (make_moons):\n")
print(pairwise_distances_matrix_r)
```

Отримуємо:




```
Estimated number of clusters після видалення об'єкта (make_moons): 2
Adjusted Rand Index після видалення об'єкта (make_moons): 0.32
V-measure після видалення об'єкта (make_moons): 0.25

Матриця відстаней кластеризації 'ward' після видалення об'єкта (make_moons):

[[0.         1.67422612]
 [1.67422612 0.        ]]
```

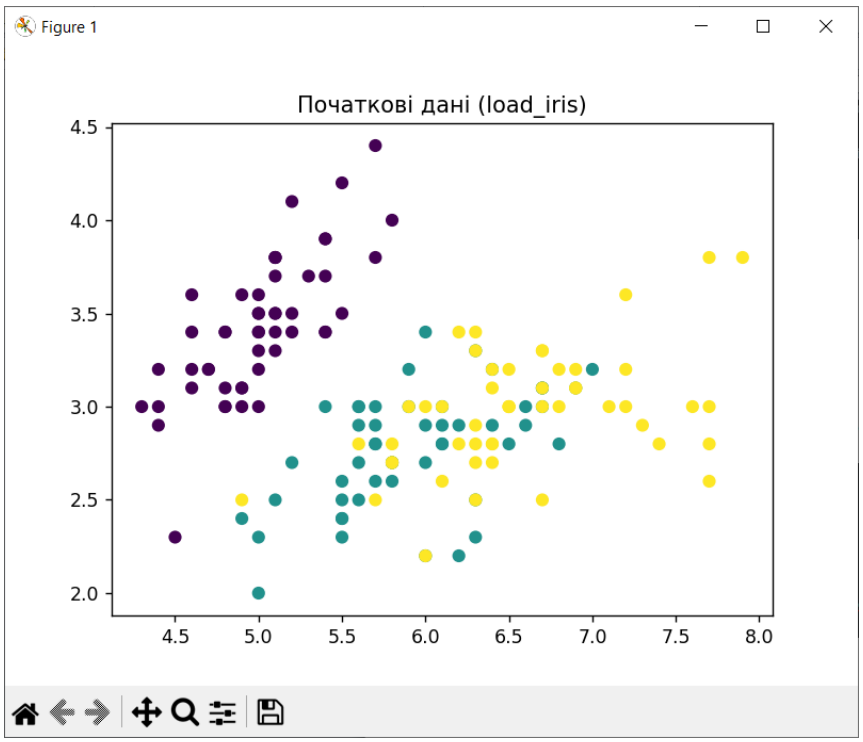
Розбиття зберігає стабільність

Набір (б) load_iris

Хід роботи аналогічний:

1.Представляємо набір графічно:

```
# Завдання (1б): Завантаження та візуалізація даних
iris = datasets.load_iris()
X = iris.data
y = iris.target
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis')
plt.title("Початкові дані (load_iris)")
plt.show()
```



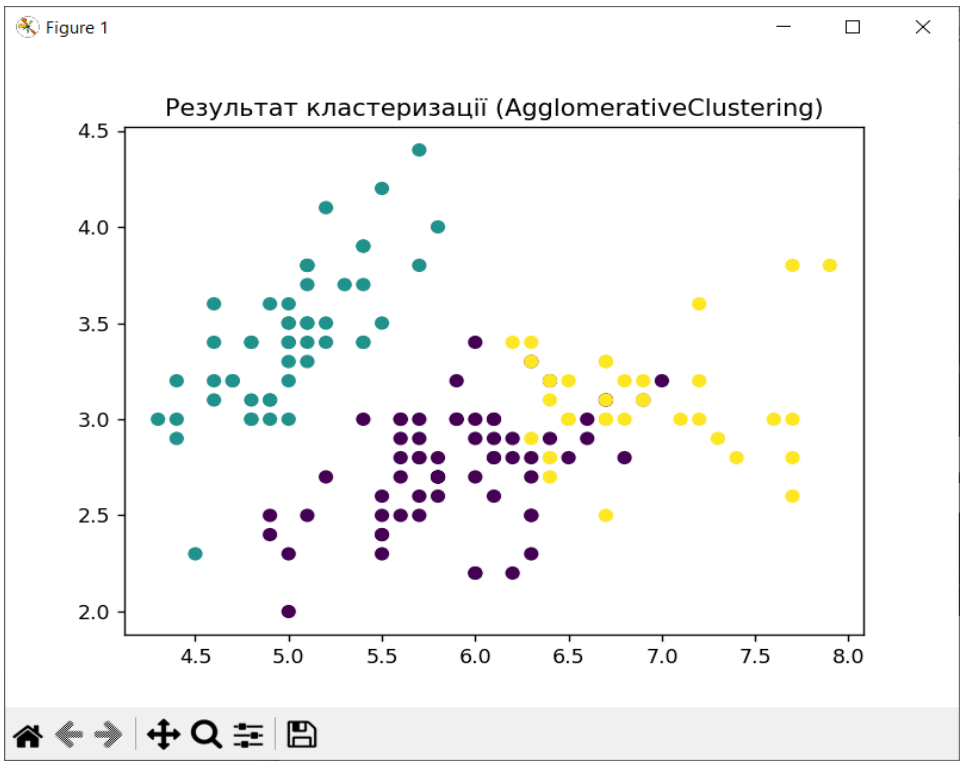
2+3. Побудуємо модель кластеризації, зафіксуємо час роботи моделі:

```
# Завдання (2б): Створення моделі AgglomerativeClustering
model2 = AgglomerativeClustering(n_clusters=3)
# Завдання (3б): Кластеризація даних
start_time = time.time()
y_pred2 = model2.fit_predict(X)
elapsed_time = time.time() - start_time
print(f"Час кластеризації: {elapsed_time:.2f} сек")
```

```
Час кластеризації: 0.01 сек
```

4.Візуалізуємо отриману кластеризацію (за замовчанням linkage = ‘ward’):

```
# Завдання (4б): Візуалізація розбиття на кластери
plt.scatter(X[:, 0], X[:, 1], c=y_pred2, cmap='viridis')
plt.title("Результат кластеризації (AgglomerativeClustering)")
plt.show()
```



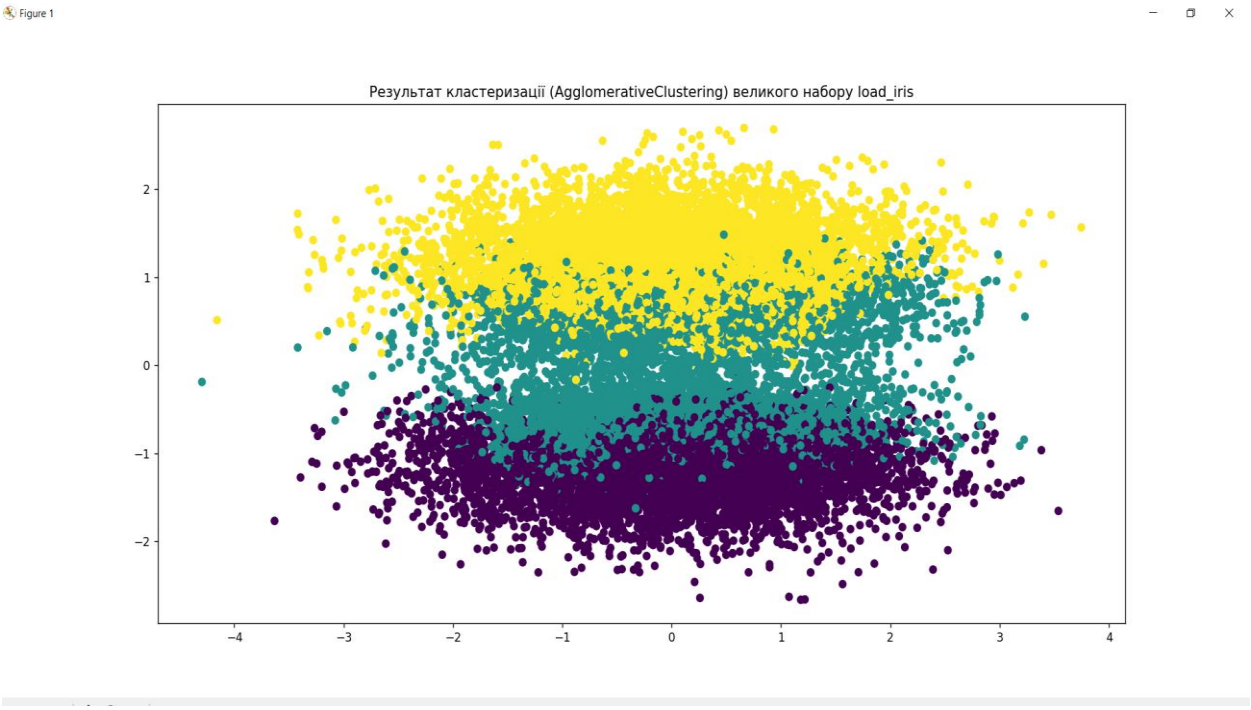
5. Оцінимо швидкодію методу на надвеликих наборах даних, збільшимо кількість об’єктів в наборі до 20000:

(Зауваження: набір load_iris є класичним і має фіксовану кількість об’єктів, тож йому не можна додати samples, але можна симулювати його збільшення з допомогою створення іншого набору, наприклад з make_classification):

```
# Завдання (5б): Оцінка швидкості та якості для великого набору даних
X_large, y_large = datasets.make_classification(n_samples=20000,
n_features=4, n_informative=2, n_redundant=1, n_clusters_per_class=1,
random_state=42)
model_large2 = AgglomerativeClustering(n_clusters=3)

start_time = time.time()
y_pred_large2 = model_large2.fit_predict(X_large)
elapsed_time_large = time.time() - start_time
print(f"Час кластеризації для великого набору даних (load_iris):
{elapsed_time_large:.2f} сек")
plt.scatter(X_large[:, 0], X_large[:, 1], c=y_pred_large2,
cmap='viridis')
plt.title("Результат кластеризації (AgglomerativeClustering) великого
набору load_iris")
plt.show()
```

Отримуємо:

```
Час кластеризації для великого набору даних (load_iris): 15.84 сек
```

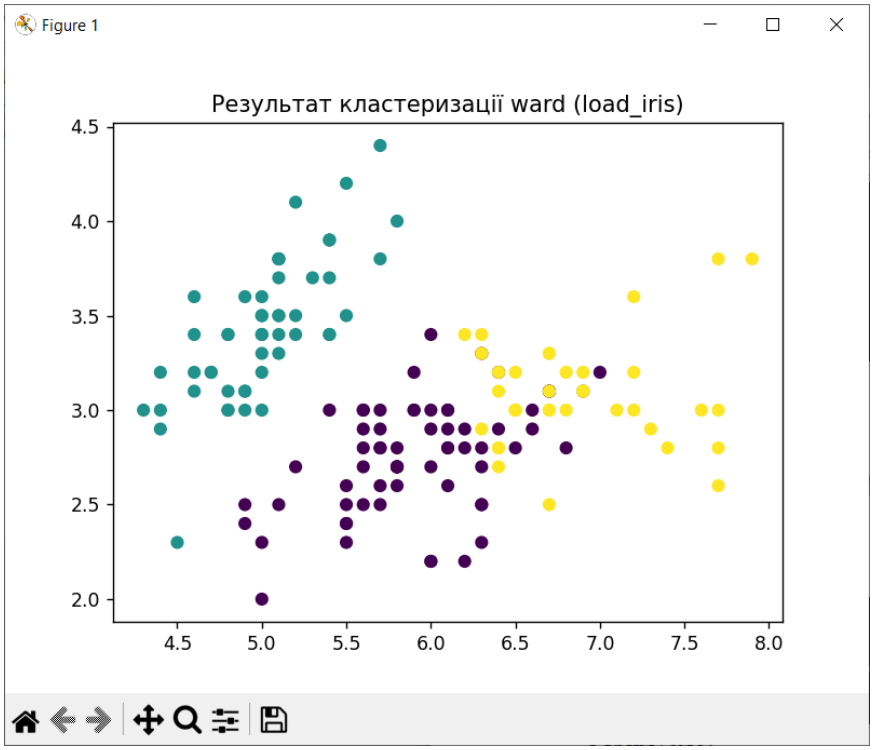


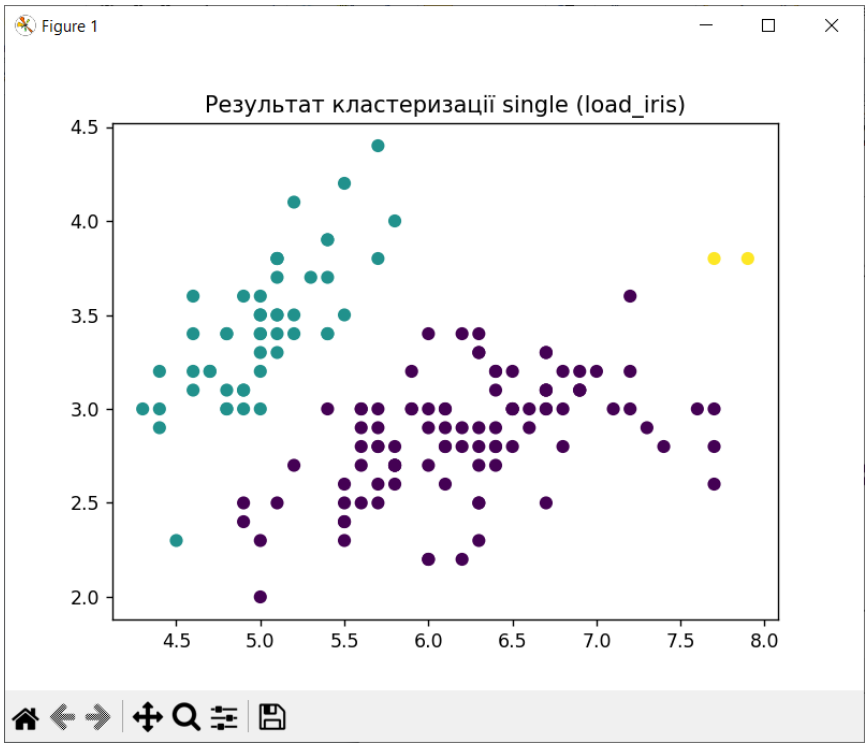
6+7. В циклі побудуємо альтернативні моделі кластеризації (використовуючи різні функції відстані), візуалізуємо їх, розрахуємо задані метрики якості моделей:

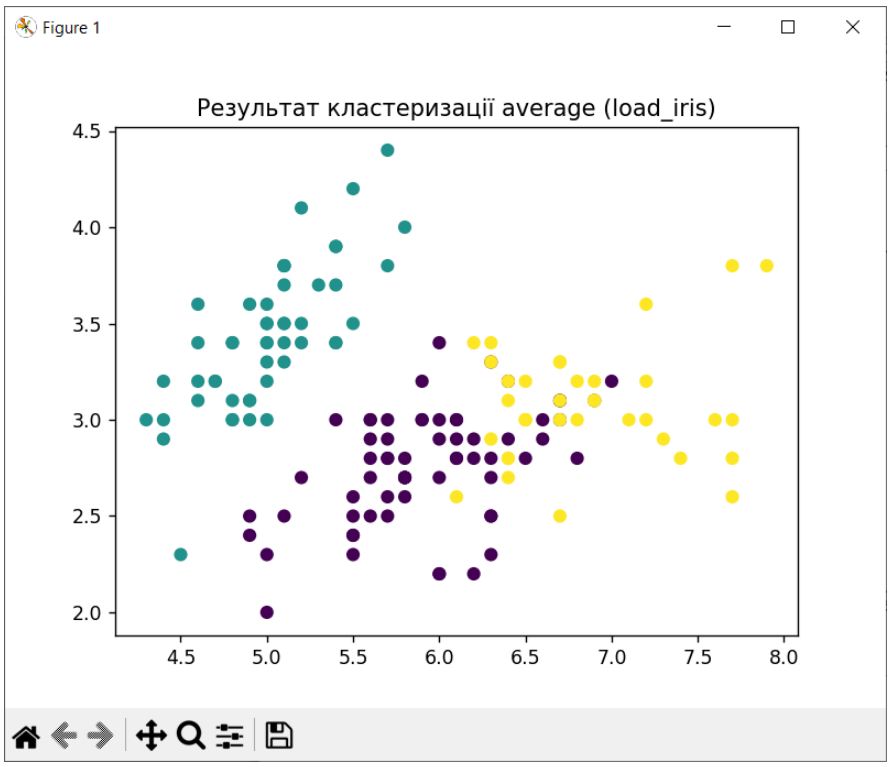
```
# Завдання (6+7 (б)): Побудова альтернативних моделей для load_iris
models_iris_alternative = []

for method in linkage_methods:
    model_iris_alternative = AgglomerativeClustering(n_clusters=3,
linkage=method)
    models_iris_alternative.append(model_iris_alternative)

# Візуалізація альтернативних моделей та метрик + матриця
pairwise_distances для кластерів
for i, model2 in enumerate(models_iris_alternative):
    y_pred_iris_alternative = model2.fit_predict(X)
    clusters_iris = get_clusters(X, y_pred_iris_alternative)
    plt.scatter(X[:, 0], X[:, 1], c=y_pred_iris_alternative,
cmap='viridis')
    plt.title(f"Результат кластеризації {linkage_methods[i]}
(load_iris)")
    plt.show()
    estimated_clusters_iris = estimated_num_clusters(model2, X)
    ari_iris = adjusted_rand_score(y, y_pred_iris_alternative)
    v_measure_iris = v_measure_score(y, y_pred_iris_alternative)
    pairwise_distances_matrix_iris =
cluster_distance_matrix(clusters_iris)
    print(f"\nEstimated number of clusters {linkage_methods[i]}
(load_iris): {estimated_clusters_iris}")
    print(f"Adjusted Rand Index {linkage_methods[i]} (load_iris):
{ari_iris:.2f}")
```

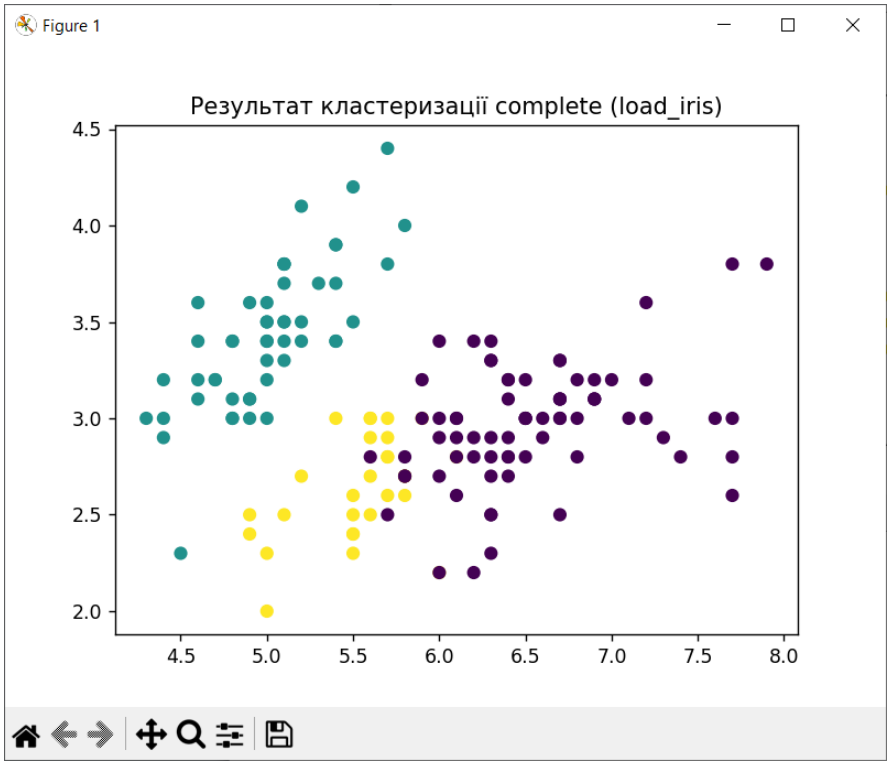
```
    print(f"V-measure {linkage_methods[i]} (load_iris):
{v_measure_iris:.2f}")
    print(f"\nМатриця відстаней кластеризації {linkage_methods[i]}
(load_iris):\n")
    print(pairwise_distances_matrix_iris)
```

Отримуємо:









Метрики та матриці відстаней між кластерами:

```
Estimated number of clusters ward (load_iris): 3
Adjusted Rand Index ward (load_iris): 0.73
V-measure ward (load_iris): 0.77

Матриця відстаней кластеризації ward (load_iris):

[[0.         3.47275396 1.96201896]
 [3.47275396 0.         5.11144486]
 [1.96201896 5.11144486 0.        ]]

Estimated number of clusters single (load_iris): 3
Adjusted Rand Index single (load_iris): 0.56
V-measure single (load_iris): 0.72

Матриця відстаней кластеризації single (load_iris):

[[0.         4.02056879 2.61637748]
 [4.02056879 0.         6.12626354]
 [2.61637748 6.12626354 0.        ]]
```

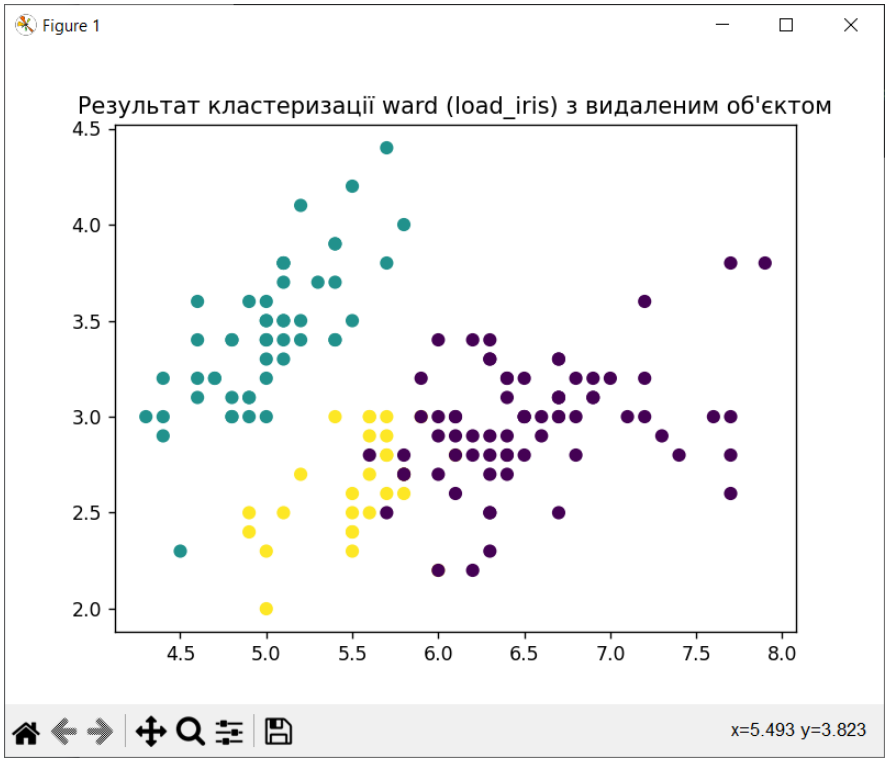
```
Estimated number of clusters average (load_iris): 3
Adjusted Rand Index average (load_iris): 0.76
V-measure average (load_iris): 0.81

Матриця відстаней кластеризації average (load_iris):

[[0.         3.46800637 1.96361409]
 [3.46800637 0.         5.11988503]
 [1.96361409 5.11988503 0.        ]]

Estimated number of clusters complete (load_iris): 3
Adjusted Rand Index complete (load_iris): 0.64
V-measure complete (load_iris): 0.72

Матриця відстаней кластеризації complete (load_iris):

[[0.         4.50202196 1.93411212]
 [4.50202196 0.         2.93295313]
 [1.93411212 2.93295313 0.        ]]
```

8. Виконаємо аналіз результатів кластеризації одним з неформальних методів (видаливши окремий елемент):

```python
# Завдання (8б): Аналіз стабільності розбиття
# Видалення одного об'єкта та повторна кластеризація
idx_to_remove = np.random.randint(0, len(X))
X_removed = np.delete(X, idx_to_remove, axis=0)
y_removed = np.delete(y, idx_to_remove)
y_pred_removed = model2.fit_predict(X_removed)
clusters_r = get_clusters(X_removed, y_pred_removed)
```

```
plt.scatter(X_removed[:, 0], X_removed[:, 1], c=y_pred_removed,
cmap='viridis')
plt.title("Результат кластеризації ward (load_iris) з видаленим
об'єктом")
plt.show()
pairwise_distances_matrix_r = cluster_distance_matrix(clusters_r)
estimated_clusters_r = estimated_num_clusters(model2, X_removed)
ari_r = adjusted_rand_score(y_removed, y_pred_removed)
v_measure_r = v_measure_score(y_removed, y_pred_removed)
print(f"\nEstimated number of clusters після видалення об'єкта
(load_iris): {estimated_clusters_r}")
print(f"Adjusted Rand Index після видалення об'єкта (load_iris):
{ari_r:.2f}")
print(f"V-measure після видалення об'єкта (load_iris):
{v_measure_r:.2f}")
print(f"\nМатриця відстаней кластеризації 'ward' після видалення
об'єкта (load_iris):\n")
print(pairwise_distances_matrix_r)
```

Отримуємо:




```
Estimated number of clusters після видалення об'єкта (load_iris): 3
Adjusted Rand Index після видалення об'єкта (load_iris): 0.64
V-measure після видалення об'єкта (load_iris): 0.72

Матриця відстаней кластеризації 'ward' після видалення об'єкта (load_iris):

[[0.         4.50027839 1.93081964]
 [4.50027839 0.         2.93295313]
 [1.93081964 2.93295313 0.        ]]
```

Для даного набору з видаленим елементом розбиття менш стабільне, порівняно з початковими результатами.

Проаналізувавши побудовані моделі кластеризації AgglomerativeClustering, їх різні методи, одержані метрики та швидкодію для наших наборів, можна зробити висновок, що для набору make_moons найкращим чином кластеризація одержана з single – зв’язком, а для load_iris найбільш наближеним до правильного розбиття був метод з ward зв’язком.

Висновки: У ході виконання комп’ютерного практикуму були опрацьовані механізми кластеризації моделей засобами sklearn, зокрема проведено дослідження алгоритму AgglomerativeClustering та протестовано всі можливі його різновиди знаходження кластерів для заданих наборів, також продемонстровано одержані метрики для моделей.