МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
Національний технічний університет
«Харківський політехнічний інститут»
Кафедра ГМКГ

Лабораторна работа №3

З дисципліни «Інтелектуальний аналіз даних»

Виконав:

Студент групи ІКМ-220 г.

Ульянов Кирило Юрійович

Перевірив:

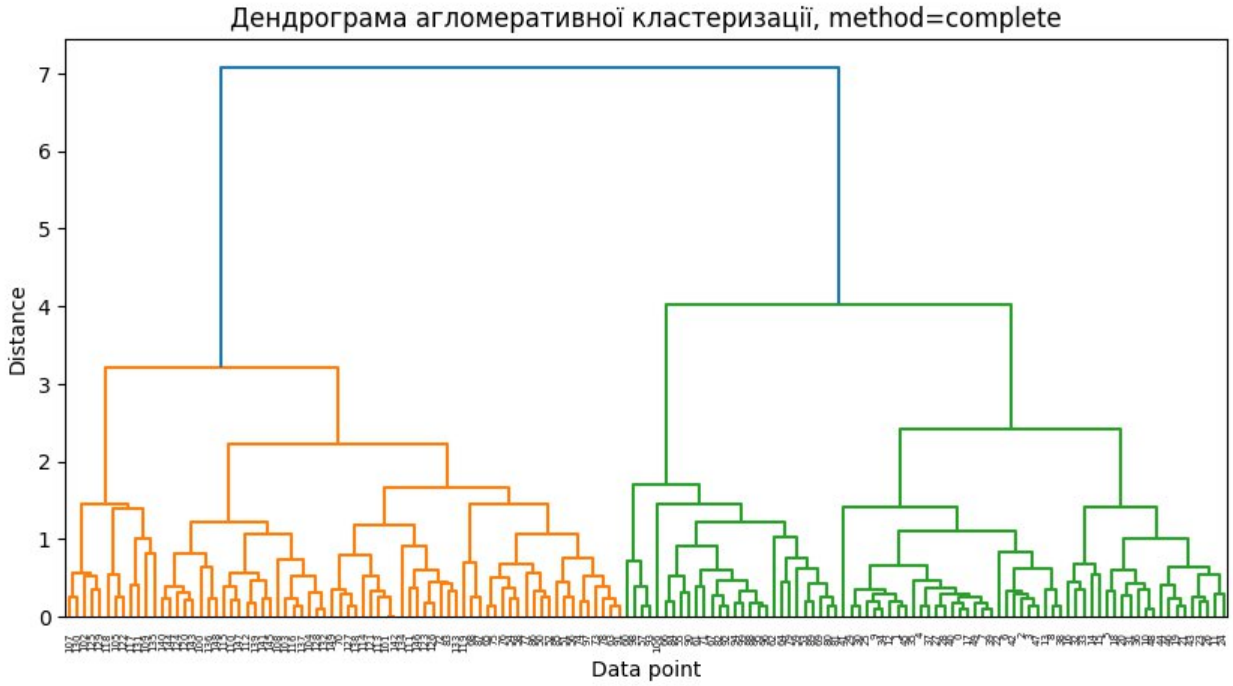
Доц. Дашкевич А.О.

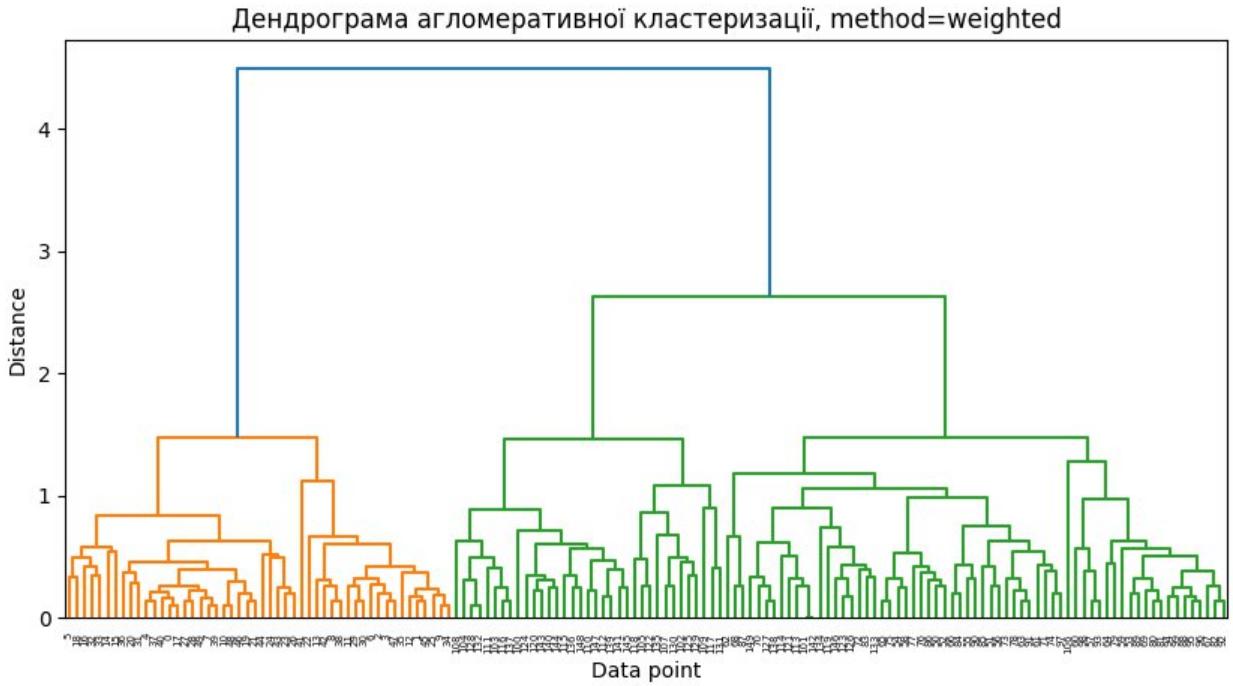
Харків 2023

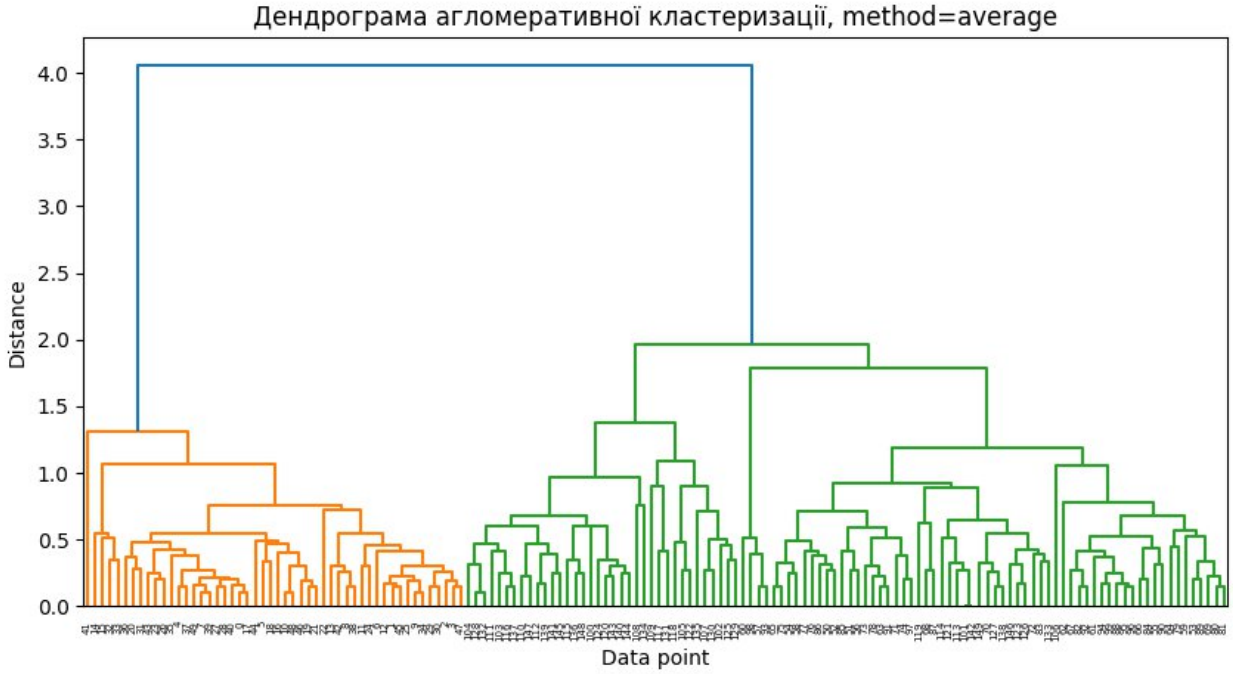
Мета роботи: вивчення базових алгоритмів кластеризації агломеративного та центроїдного типу.

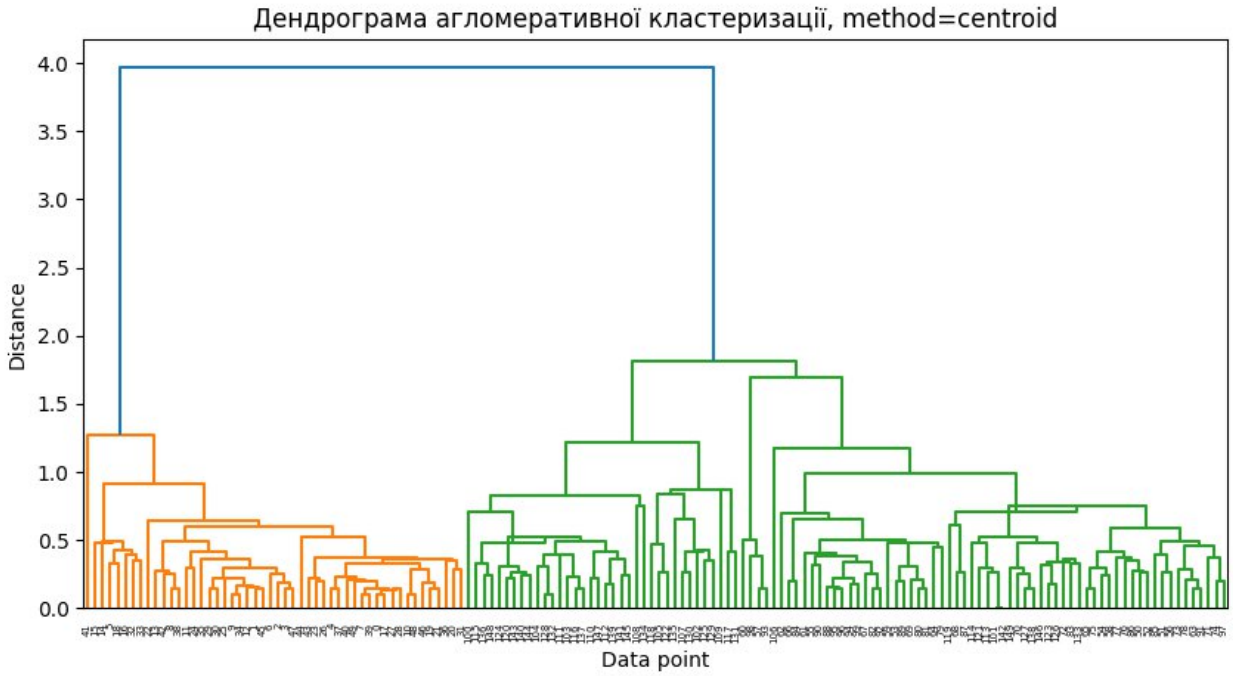
Завдання на роботу: завантаження набору даних, формування вхідної вибірки даних, кластеризація із застосуванням алгоритмів агломеративної та центроїдної (алгоритми k-means та Mean Shift) кластеризації, візуалізація дендрограми за результатами агломеративної кластеризації.

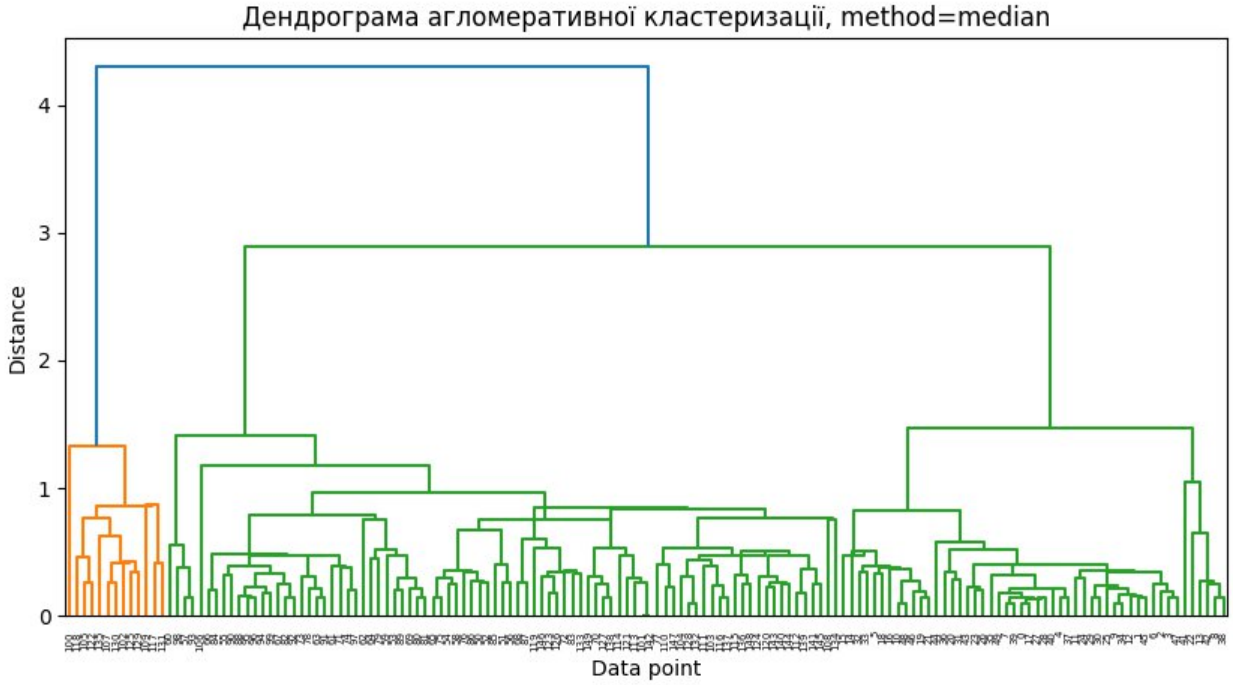
Побудовані дендрограми для набору «Іриси Фішера»




Дендрограма агломеративної кластеризації, method=weighted
Distance
Data point


Дендрограма агломеративної кластеризації, method=average
Distance
Data point





Отримані точності:

Точність на ненормалізованих даних: **0.77629**

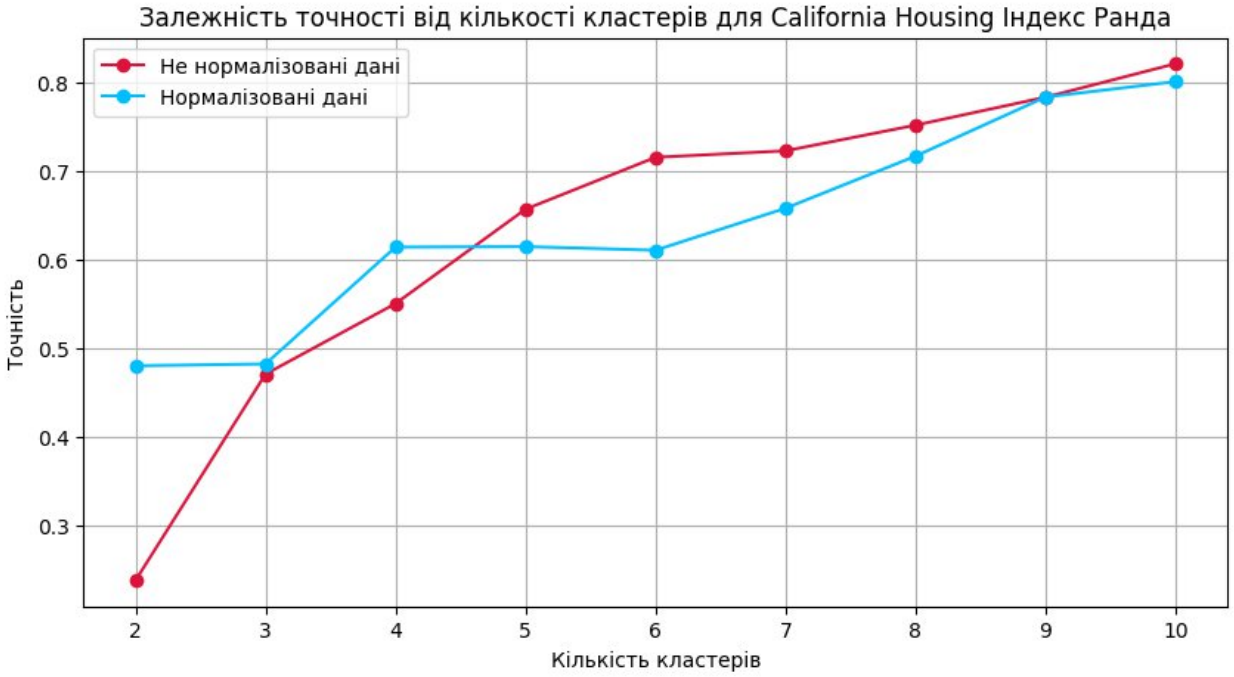
Точність на стандартизованих даних: **0.76295**

Точність на нормалізованих даних: **0.77629**

Залежність точності від кількості кластерів в методі KMeans.

Метрика – Індекс Ранда

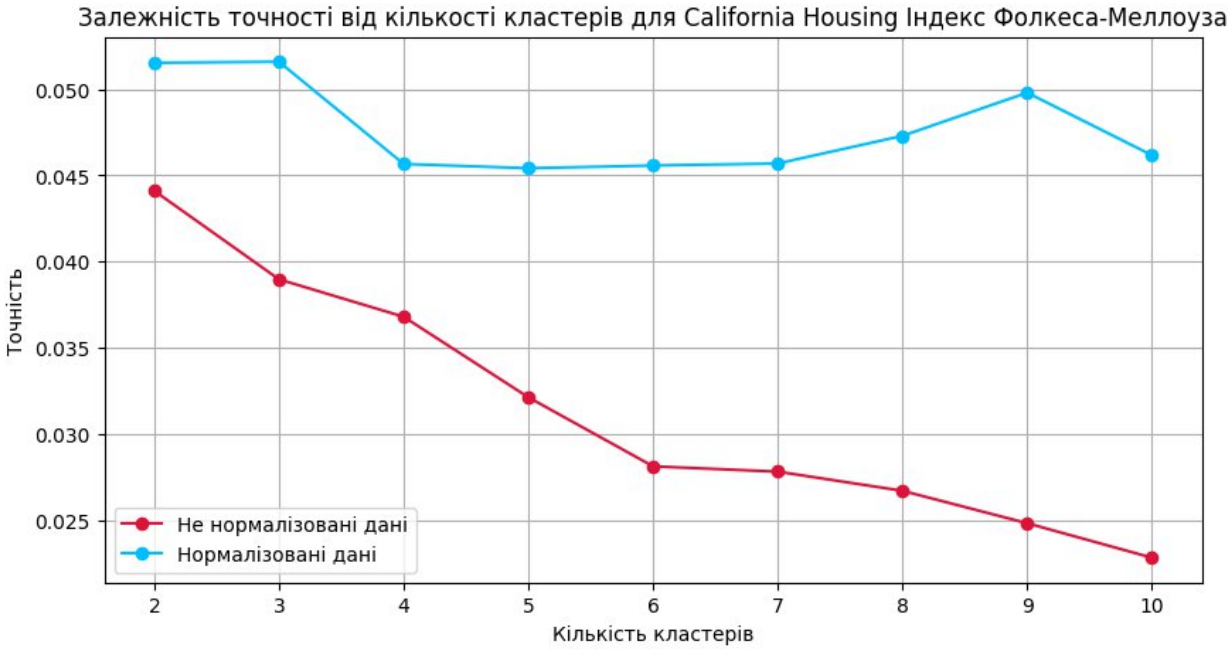
Кількість кластерів	Точність на ненормалізованих даних	Точність на нормалізованих даних
2	0.23788	0.48034
3	0.47083	0.48239
4	0.55051	0.61431
5	0.65709	0.61488
6	0.71569	0.61079
7	0.72302	0.65820
8	0.75200	0.71707
9	0.78359	0.78374
10	0.82141	0.80117



Залежність точності від кількості кластерів в методі KMeans.

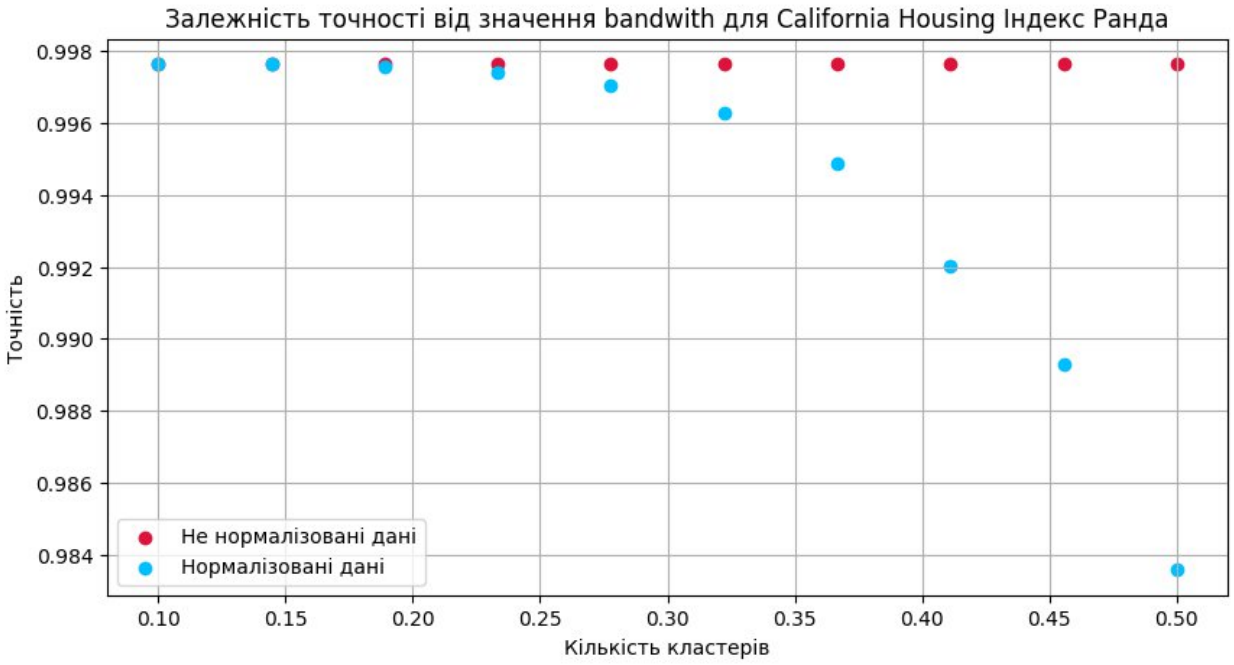
Метрика – Індекс Фолкеса-Меллоуза

Кількість кластерів	Точність на ненормалізованих даних	Точність на нормалізованих даних
2	0.04410	0.05153
3	0.03897	0.05159
4	0.03679	0.04566
5	0.03212	0.04541
6	0.02812	0.04557
7	0.02781	0.04569
8	0.02670	0.04729
9	0.02481	0.04980
10	0.02282	0.04617



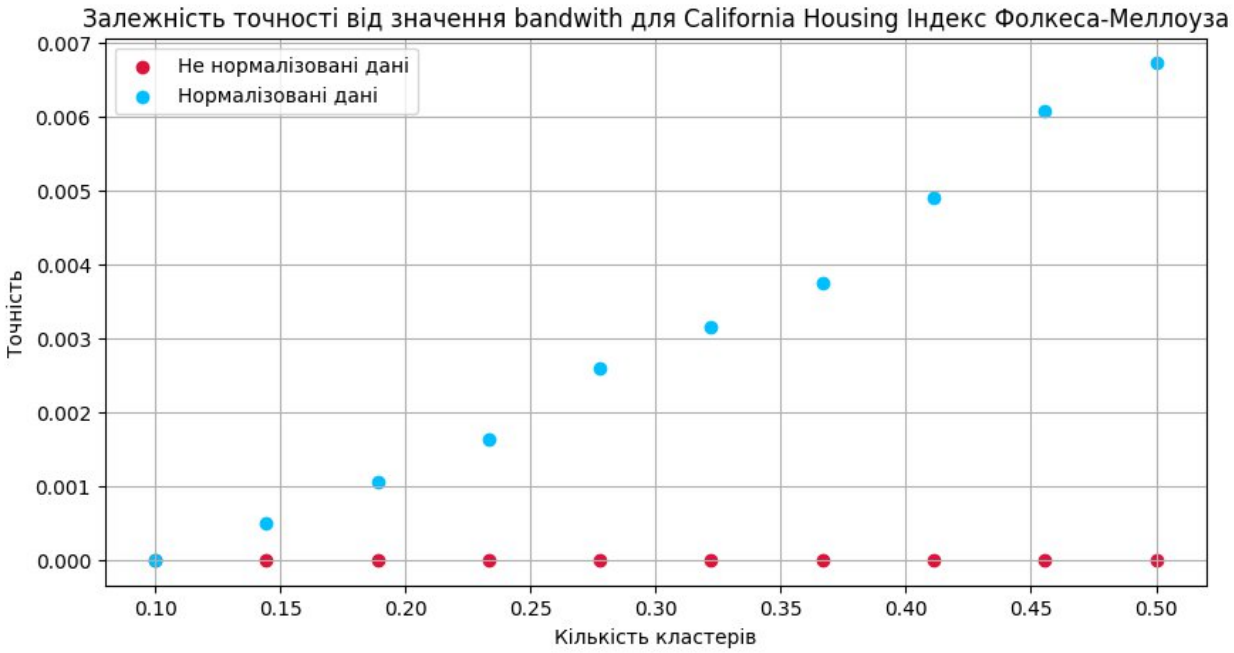
Залежність точності від значення bandwith для алгоритму Mean Shift. Метрика - Індекс Ранда

Значення bandwidth	Точність на ненормалізованих даних	Точність на нормалізованих даних
0.10000	0.99763	0.99763
0.14444	0.99763	0.99762
0.18889	0.99763	0.99757
0.23333	0.99763	0.99741
0.27778	0.99763	0.99705
0.32222	0.99763	0.99628
0.36667	0.99763	0.99486
0.41111	0.99763	0.99202
0.45556	0.99763	0.98929
0.50000	0.99763	0.98359



Залежність точності від значення bandwith для алгоритму Mean Shift. Метрика - Індекс Фолкеса-Меллоуза

Значення bandwidth	Точність на ненормалізованих даних	Точність на нормалізованих даних
0.10000	0.00000	0.00000
0.14444	0.00000	0.00051
0.18889	0.00000	0.00107
0.23333	0.00000	0.00164
0.27778	0.00000	0.00260
0.32222	0.00000	0.00316
0.36667	0.00000	0.00375
0.41111	0.00000	0.00490
0.45556	0.00000	0.00608
0.50000	0.00000	0.00672



Код програми:

```
from sklearn.cluster import AgglomerativeClustering, KMeans, MeanShift

import numpy as np

from scipy.cluster.hierarchy import dendrogram, linkage

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler, Normalizer

from sklearn.datasets import load_iris, fetch_california_housing

from sklearn.metrics import accuracy_score, mean_squared_error, rand_score, fowlkes_mallows_score




iris = load_iris()

X = iris.data


method_list = ['single', 'complete', 'average', 'weighted', 'centroid', 'median', 'ward']


def plot_metrics(cluster_range, metric_list_unnormalized, metric_list_normalized, title=None, scatter=False):

    """

    Визуализирует зависимость метрик от числа кластеров для необработанных и нормализованных данных.




    Параметры:

    - `cluster_range (range)`: Диапазон числа кластеров.

    - `metric_list_unnormalized (list)`: Список метрик для необработанных данных.

    - `metric_list_normalized (list)`: Список метрик для нормализованных данных.

    - `title (str)`: Заголовок графика.
```

```
    - `scatter (boolean)`: Вивести точковий графік.



    Возвращает:

    - `None`

    """

    if scatter:

        plt.figure(figsize=(10, 5))

        plt.scatter(cluster_range, metric_list_unnormalized,
marker='o', label='Не нормалізовані дані', color='crimson')

        plt.scatter(cluster_range, metric_list_normalized,
marker='o', label='Нормалізовані дані', color='deepskyblue')

        plt.xlabel('Кількість кластерів')

        plt.ylabel('Точність')

        plt.title(title)

        plt.legend()

        plt.grid(True)

        plt.show()

    else:

        plt.figure(figsize=(10, 5))

        plt.plot(cluster_range, metric_list_unnormalized,
marker='o', label='Не нормалізовані дані', color='crimson')

        plt.plot(cluster_range, metric_list_normalized,
marker='o', label='Нормалізовані дані', color='deepskyblue')

        plt.xlabel('Кількість кластерів')

        plt.ylabel('Точність')

        plt.title(title)

        plt.legend()

        plt.grid(True)

        plt.show()


def fit_and_predict(X):
```

```
    """
    Кластеризует данные методом агломерации, используя
AgglomerativeClustering.



    Параметры:
    - `X (numpy.ndarray)`: Матрица признаков для кластеризации.



    Возвращает:
    - `float`: Точность предсказаний, измеренная с
использованием индекса Рэнда.
    """
    aglom_cluster = AgglomerativeClustering(n_clusters=2)
    predictions = aglom_cluster.fit_predict(X)
    accuracy = rand_score(iris.target, predictions)
    print(aglom_cluster.labels_)
    return accuracy


for method in method_list:
    A = linkage(X, method=method)
    plt.figure(figsize=(10, 5))
    dendrogram(A)
    plt.title(f'Дендрограма агломеративної кластеризації,
method={method}')
    plt.xlabel('Data point')
    plt.ylabel('Distance')
    plt.show()


accuracy = fit_and_predict(X)
print(f'features are not normalized: {accuracy:.5f}')
```

```
X_scaled = StandardScaler().fit_transform(X)
accuracy_scaled = fit_and_predict(X_scaled)
print(f'features are standardized: {accuracy_scaled:.5f}')

X_normalized = Normalizer().fit_transform(X)
accuracy_normalized = fit_and_predict(X_normalized)
print(f'features are normalized: {accuracy_normalized:.5f}')

california = fetch_california_housing()
X = california.data[:7000]
X_scaled = StandardScaler().fit_transform(X)

cluster_range = range(2, 11)
kmeans_accuracy_list_unnormalized_rand = []
kmeans_accuracy_list_normalized_rand = []
kmeans_accuracy_list_unnormalized_fowlkes = []
kmeans_accuracy_list_normalized_fowlkes = []

for cluster in cluster_range:

    kmeans = KMeans(n_clusters=cluster)
    predictions = kmeans.fit_predict(X)
    accuracy_rand = rand_score(california.target[:7000],
predictions)
    kmeans_accuracy_list_unnormalized_rand.append(accuracy_rand)
    accuracy_fowlkes =
fowlkes_mallows_score(california.target[:7000], predictions)
    kmeans_accuracy_list_unnormalized_fowlkes.append(accuracy_fo
wlkes)

    kmeans = KMeans(n_clusters=cluster)
    predictions = kmeans.fit_predict(X_scaled)
```

```
    accuracy_rand = rand_score(california.target[:7000],
predictions)

    kmeans_accuracy_list_normalized_rand.append(accuracy_rand)

    accuracy_fowlkes =
fowlkes_mallows_score(california.target[:7000], predictions)

    kmeans_accuracy_list_normalized_fowlkes.append(accuracy_fowl
kes)




plot_metrics(cluster_range,
kmeans_accuracy_list_unnormalized_rand,
kmeans_accuracy_list_normalized_rand, 'Залежність точності від
кількості кластерів для California Housing Індекс Ранда')

plot_metrics(cluster_range,
kmeans_accuracy_list_unnormalized_fowlkes,
kmeans_accuracy_list_normalized_fowlkes, 'Залежність точності
від кількості кластерів для California Housing Індекс Фолкеса-
Меллоуза')


bandwidth_range = np.linspace(0.1, 0.5, num=10)


meanshift_accuracy_unnormalized_rand = []

meanshift_accuracy_normalized_rand = []

meanshift_accuracy_unnormalized_fowlkes = []

meanshift_accuracy_normalized_fowlkes = []




for bandwidth in bandwidth_range:

    mean_shift = MeanShift(bandwidth=bandwidth)

    predictions = mean_shift.fit_predict(X)

    accuracy = rand_score(california.target[:7000], predictions)

    meanshift_accuracy_unnormalized_rand.append(accuracy)

    meanshift_accuracy_unnormalized_fowlkes.append(fowlkes_mallo
ws_score(california.target[:7000], predictions))
```

```
    mean_shift = MeanShift(bandwidth=bandwidth)

    predictions = mean_shift.fit_predict(X_scaled)

    accuracy = rand_score(california.target[:7000], predictions)

    meanshift_accuracy_normalized_rand.append(accuracy)

    meanshift_accuracy_normalized_fowlkes.append(fowlkes_mallows
_score(california.target[:7000], predictions))


plot_metrics(bandwidth_range,
meanshift_accuracy_unnormalized_rand,
meanshift_accuracy_normalized_rand,'Залежність точності від
значення bandwith для California Housing Індекс Ранда', True)

plot_metrics(bandwidth_range,
meanshift_accuracy_unnormalized_fowlkes,
meanshift_accuracy_normalized_fowlkes, 'Залежність точності від
значення bandwith для California Housing Індекс Фолкеса-
Меллоуза', True)


for i in cluster_range:

    print(f'{i} | {kmeans_accuracy_list_unnormalized_rand[i-
2]:.5f} | {kmeans_accuracy_list_normalized_rand[i-2]:.5f}')


for i in cluster_range:

    print(f'{i} | {kmeans_accuracy_list_unnormalized_fowlkes[i-
2]:.5f} | {kmeans_accuracy_list_normalized_fowlkes[i-2]:.5f}')


for i, j in zip(bandwidth_range,
range(len(meanshift_accuracy_unnormalized_rand))):

    print(f'{i:.5f} |
{meanshift_accuracy_unnormalized_rand[j]:.5f} |
{meanshift_accuracy_normalized_rand[j]:.5f}')


for i, j in zip(bandwidth_range,
range(len(meanshift_accuracy_unnormalized_rand))):

    print(f'{i:.5f} |
{meanshift_accuracy_unnormalized_fowlkes[j]:.5f} |
{meanshift_accuracy_normalized_fowlkes[j]:.5f}')
```