Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №9
по дисциплине «Основы систем ИИ»

Тема Кластеризация

Студент Батуев А.Г.

Группа ИУ7-36Б

Преподаватели Строганов Ю.В.

Москва, 2025

Содержание

ВВЕДЕНИЕ

Цель: променить алгоритмы кластеризации (k-средних, c-средних, иерархическая) к векторам документов.

Задачи:

— применить алгоритмы кластеризации с различным количество кластеров
— посчитать среднее внутрикластерное и межкластерное расстояние
— провести анализ для выбора оптимального количества кластеров
— визуализировать кластеры

1 Аналитическая часть

Кластеризация является одной из ключевых задач анализа данных и машинного обучения. Её цель — разделение множества объектов на группы (кластеры) таким образом, чтобы объекты внутри одного кластера были более схожи между собой, чем с объектами из других кластеров.

1.1 Методы кластеризации

1.1.1 Метод *k-means*

k-means — один из наиболее популярных алгоритмов кластеризации. Его цель заключается в минимизации суммы квадратов отклонений объектов от центров их кластеров. Формально:

$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$ [14], где k — количество кластеров, C_i — множество объектов в кластере i, μ_i — центр кластера i (среднее значение объектов кластера).

Процесс выполнения:

1) инициализация k центров кластеров случайным образом.
2) назначение каждого объекта ближайшему центру.
3) пересчёт центров кластеров
4) повторение шагов 2-3 до сходимости.

1.1.2 Метод *c-means* (размытая кластеризация)

Метод *c-means* является обобщением *k-means* и основывается на размытом распределении принадлежности объектов к кластерам. Объекты могут принадлежать нескольким кластерам с различными степенями принадлежности. Функция оптимизации:

$J = \sum_{i=1}^{k} \sum_{j=1}^{n} u_{ij}^m \|x_j - \mu_i\|^2$, [15], где u_{ij} — степень принадлежности объекта x_j к кластеру i, $m > 1$ — параметр размытости.

Алгоритм:

1) инициализация матрицы принадлежностей.
2) обновление центров кластеров: $\mu_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}$. [15]
3) обновление матрицы принадлежностей: $u_{ij} = \frac{1}{\sum_{k=1}^{k} \left(\frac{\|x_j - \mu_i\|}{\|x_j - \mu_k\|} \right)^{\frac{2}{m-1}}}$. [15]
4) повторение шагов 2-3 до сходимости.

1.1.3 Иерархическая кластеризация

Иерархическая кластеризация строит древовидную структуру кластеров (дендрограмму). Алгоритм работает по одному из двух подходов:

— **агломеративный подход**. Начинается с каждого объекта в отдельном кластере, затем кластеры объединяются.

— **дивизионный подход**. Начинается с одного кластера, включающего все объекты, затем кластеры делятся.

Критерий объединения или разделения кластеров может быть основан на различных метриках, таких как минимальное расстояние (метод одиночной связи) или максимальное расстояние (метод полной связи).

1.2 Выбор количества класетров

Кластеризация является важным методом анализа данных, используемым для разделения объектов на группы (кластеры) с учетом их схожести. Однако выбор оптимального количества кластеров k представляет собой сложную задачу. В данной работе рассмотрены два подхода к определению оптимального числа кластеров: метод локтя для k-means и c-means, а также использование дендрограмм в иерархической кластеризации.

1.2.1 Метод локтя

Метод локтя является визуальным подходом к выбору оптимального количества кластеров в алгоритмах k-means и c-means.

Метод основан на анализе изменения значения функции стоимости $W(k)$ при увеличении числа кластеров k. Для k-means и c-means функция стоимости представляет собой сумму квадратов расстояний от объектов до ближайшего центроида (в k-means) или до центроидов с учетом весов принадлежности (в c-means).

1.2.2 Иерархическая кластеризация и дендрограммы

Иерархическая кластеризация позволяет построить дерево кластеров (дендрограмму), где каждый узел представляет собой объединение кластеров. Определение оптимального числа кластеров осуществляется на основе анализа структуры дендрограммы. осуществляется поиск самой длинной вертикальной непрерывной линии в дендрограмме. По ней проводится горизонтальная линия, которая определяет оптимальное количество кластеров, исходя из количества пересечений вертикальных линий.

1.3 Заключение

Выбор метода кластеризации зависит от особенностей данных и целей анализа. Для компактных и чётко разделённых данных хорошо подходит *k-means*. Для данных с размытыми границами кластеров рекомендуется использовать *c-means*. Иерархическая кластеризация подходит для анализа структуры данных и визуализации.

2 Конструкторская часть

В рамках проектирования методов кластеризации основное внимание уделяется разработке алгоритмов, которые обеспечивают эффективность, точность и устойчивость к особенностям исходных данных. Это включает выбор оптимальных параметров, определение структуры данных и обеспечение интерпретируемости результатов.

2.1 Структура реализации алгоритма *k-means*

1. Входные данные:

— $X = \{x_1, x_2, \ldots, x_n\}$ — множество объектов.

— k — число кластеров.

2. Инициализация:

— задание начальных центров кластеров $\mu_1, \mu_2, \ldots, \mu_k$ случайным образом или с использованием стратегии *k-means++*.

3. Цикл оптимизации:

1) назначение каждого объекта x_j кластеру C_i на основе ближайшего центра: $C_i = \{x_j : \|x_j - \mu_i\| \leq \|x_j - \mu_l\|, \forall l \neq i\}$. [14]

2) пересчёт центров кластеров: $\mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j$. [14]

4. Критерий остановки:

— центры кластеров перестают изменяться или достигается максимальное число итераций.

2.2 Структура реализации алгоритма *c-means*

1. Входные данные:

— $X = \{x_1, x_2, \ldots, x_n\}$ — множество объектов.

— k — число кластеров.

— $m > 1$ — параметр размытости.

2. Инициализация:

— задание начальной матрицы принадлежностей $U = \{u_{ij}\}$ случайным образом с условием $\sum_{i=1}^{k} u_{ij} = 1$ для всех j.

3. Цикл оптимизации:

1) пересчёт центров кластеров: $\mu_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}$. [15]

2) обновление матрицы принадлежностей: $u_{ij} = \frac{1}{\sum_{l=1}^{k} \left(\frac{\|x_j - \mu_i\|}{\|x_j - \mu_l\|} \right)^{\frac{2}{m-1}}}$. [15]

4. Критерий остановки:

— изменения в матрице U становятся меньше заданного порога или достигается максимальное число итераций.

2.3 Структура реализации иерархической кластеризации

1. Входные данные:

— $X = \{x_1, x_2, \ldots, x_n\}$ — множество объектов.

— метрика расстояния (например, евклидово расстояние).

2. Инициализация:

— каждому объекту x_j соответствует отдельный кластер.

3. Цикл объединения (агломеративный подход):

1) вычисление расстояния между всеми парами кластеров на основе выбранной метрики.

2) объединение двух ближайших кластеров.

3) обновление расстояний между новыми и оставшимися кластерами.

4. Результат:

— дендрограмма, представляющая иерархическую структуру кластеров.

3 Технологическая часть

3.1 Средства написания программы

Для реализации программного обеспечения были использованы следующие средства:

— среда разработки PyCharm 2023 community edition [10]

— язык разработки Python 3.10 [11]

Используемые библиотеки:

— os — для доступа к файлам системы и создания папок/файлов [2]

— numpy — для быстрой работы с массивами, сохранения промежуточных данных и связи их с другими библиотеками [3]

— matplotlib.pyplot — для отображения графиков [4]

— seaborn — для построения графиков [5]

— pathlib.Path — доступ к файлам системы [9]

— sklearn — методы кластеризации [6]

— scipy — расчет средних внутрикластреных и межкластерных расстояний [7]

— fcmeans — работы с методом кластеризации cmeans [8]

3.2 Функции

Листинг 3.1 — Загрузка векторов документов

```
def load_vectors_from_directory(base_dir):
    vectors = []
    files = []
    for file_path in base_dir.rglob('vectors/*.txt'):
        files.append(file_path.name)
        vector = np.loadtxt(file_path)
        vectors.append(vector)
    return np.vstack(vectors), files
```

Листинг 3.2 — Кластерный показатель для kmeans

```
def kmeans_clustering_wcss_analyz(vectors):
wcss = []
for i in range(MIN_CLUSTERS, MAX_CLUSTERS):
    kmeans = KMeans(n_clusters=i, init="kmeans++", random_state=1)
    kmeans.fit(vectors)
    wcss.append(kmeans.inertia_)
return [wcss, MIN_CLUSTERS, MAX_CLUSTERS]
```

Листинг 3.3 — Внутрикластерное и межкластерное расстояние для kmeans

```
def kmeans_clustering_distance_analyze(vectors):
```

```
    inner_distances = []
    outer_distances = []

    for i in range(MIN_CLUSTERS, MAX_CLUSTERS):
        kmeans = KMeans(n_clusters=i, init="kmeans++", random_state=1)
        kmeans.fit(vectors)

        intra_distance_list = []
        for j in range(i):
            cluster_points = vectors[kmeans.labels_ == j]
            if len(cluster_points) > 0:
                intra_distance_list.append(
                    np.linalg.norm(cluster_points  kmeans.cluster_centers_
                        [j], axis=1).mean()
                )
        if intra_distance_list:
            inner_distances.append(np.mean(intra_distance_list))
        else:
            inner_distances.append(0)

        pairwise_distances = np.linalg.norm(
            kmeans.cluster_centers_[:, np.newaxis]  kmeans.
                cluster_centers_, axis=2
        )
        if np.any(pairwise_distances > 0):
            inter_distance = np.mean(pairwise_distances[pairwise_distances
                > 0])
        else:
            inter_distance = 0
        outer_distances.append(inter_distance)

    return [inner_distances, outer_distances, MIN_CLUSTERS, MAX_CLUSTERS]
```

Листинг 3.4 — Кластерный показатель для cmeans

```
    def cmeans_clustering_wcss_analyz(vectors):
    wcss = []
    for i in range(MIN_CLUSTERS, MAX_CLUSTERS):
        fcm = FCM(n_clusters=i, m=2.0, max_iter=150, error=1e-6,
            random_state=1)
        fcm.fit(vectors)

        centers = fcm.centers
```

```
        membership = fcm.u
        distance = np.linalg.norm(vectors[:, np.newaxis] centers, axis=2)
        weighted_distance = np.sum(membership ** 2 * distance ** 2)
        wcss.append(weighted_distance)

    return [wcss, MIN_CLUSTERS, MAX_CLUSTERS]
```

Листинг 3.5 — Внутрикластерное и межкластерное расстояние для cmeans

```
    pass
```

Листинг 3.6 — Дендрограмма иерархической кластеризации

```
    def hierarchical_clustering_analyze(vectors, method='ward'):
    linkage_matrix = linkage(vectors, method=method)
    dendrogram(linkage_matrix)
```

Листинг 3.7 — Внутрикластерное и межкластерное расстояние для иерархической класетризации

```
    def analyze_cluster_distances(vectors, method='ward', metric='
       euclidean', max_clusters=10):
    linkage_matrix = linkage(vectors, method=method, metric=metric)

    intra_cluster_distances = []
    inter_cluster_distances = []

    for num_clusters in range(2, max_clusters + 1):
        cluster_labels = fcluster(linkage_matrix, num_clusters, criterion
           ='maxclust')
        intra_dist = []
        for cluster_id in np.unique(cluster_labels):
            cluster_points = vectors[cluster_labels == cluster_id]
            if len(cluster_points) > 1:
                intra_dist.append(pdist(cluster_points, metric=metric).
                   mean())
        intra_cluster_distances.append(np.mean(intra_dist) if intra_dist
           else 0)
        inter_dist = []
        unique_clusters = np.unique(cluster_labels)
        for i, cluster_i in enumerate(unique_clusters):
            for j, cluster_j in enumerate(unique_clusters):
                if i < j:
                    cluster_i_points = vectors[cluster_labels == cluster_i
                       ]
```

```
                cluster_j_points = vectors[cluster_labels == cluster_j
                    ]
                inter_dist.append(cdist(cluster_i_points,
                    cluster_j_points, metric=metric).mean())
        inter_cluster_distances.append(np.mean(inter_dist) if inter_dist
            else 0)
```

Листинг 3.8 — Визуализация кластеров

```
def visualize_clusters(data, labels, title, name):
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(data)
plt.figure(figsize=(10, 8))
sea.scatterplot(x=reduced_data[:, 0], y=reduced_data[:, 1], hue=labels
    , palette="viridis", s=50)
plt.title(title)
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.legend()
plt.grid(True)
output_path = CLUSTERS / f"{title} {name}.png"
output_path.parent.mkdir(exist_ok=True, parents=True)
plt.savefig(output_path)
```

Вывод

В этой части разработаны методы для оценки оптимального количества кластеров с использованием различных методов кластеризации, таких как kmeans, cmeans и иерархическая кластеризация. Написаны функции для отображения самих кластеров, так и для расчета и отображения изменения внутрикластреного и межкластерного расстояния.

4 Исследовательская часть

Цель исследования — разбить данные на кластеры.

4.1 Оборудование

Характеритстики ноутбука:

— процессор intel-core i5-12500H [13]

— ОС Windows 11 [12]

4.2 Результаты исследования

4.2.1 Расстояния

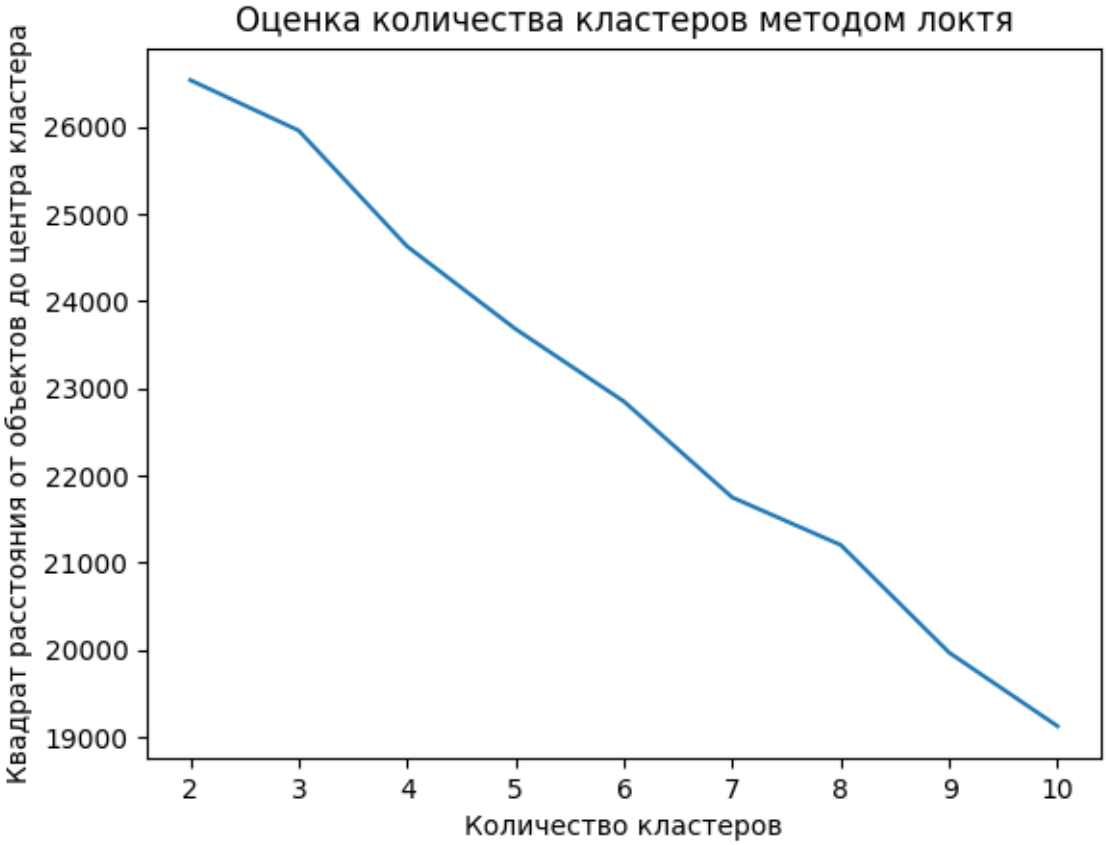

Рисунок 4.1 — Wcss отклонения для kmeans

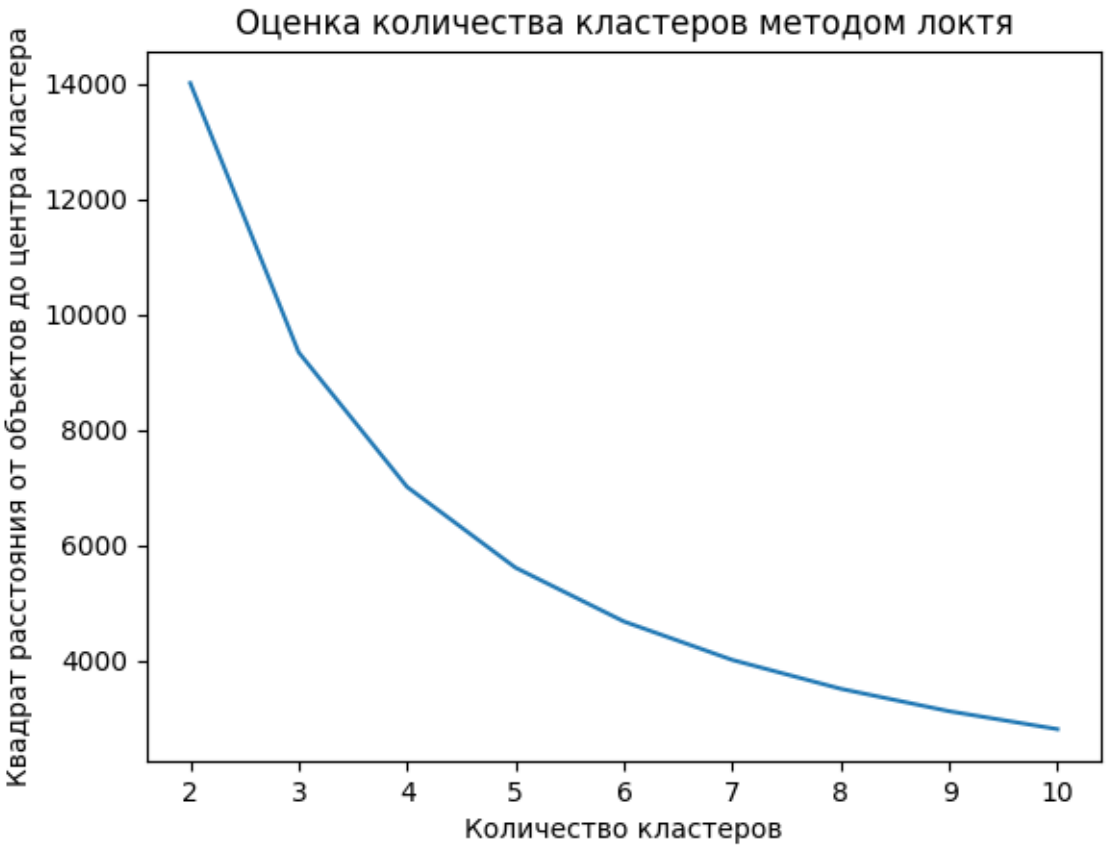

Рисунок 4.2 — Wcss отклонения для cmeans

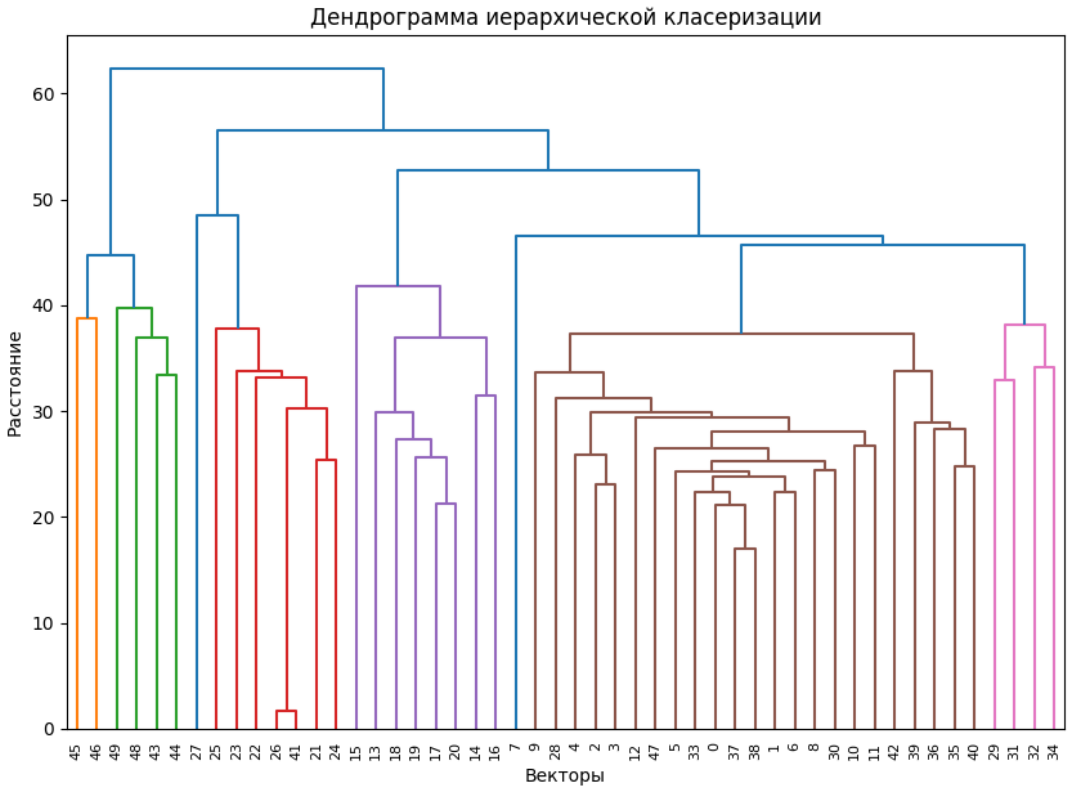

Рисунок 4.3 — Дендрограмма

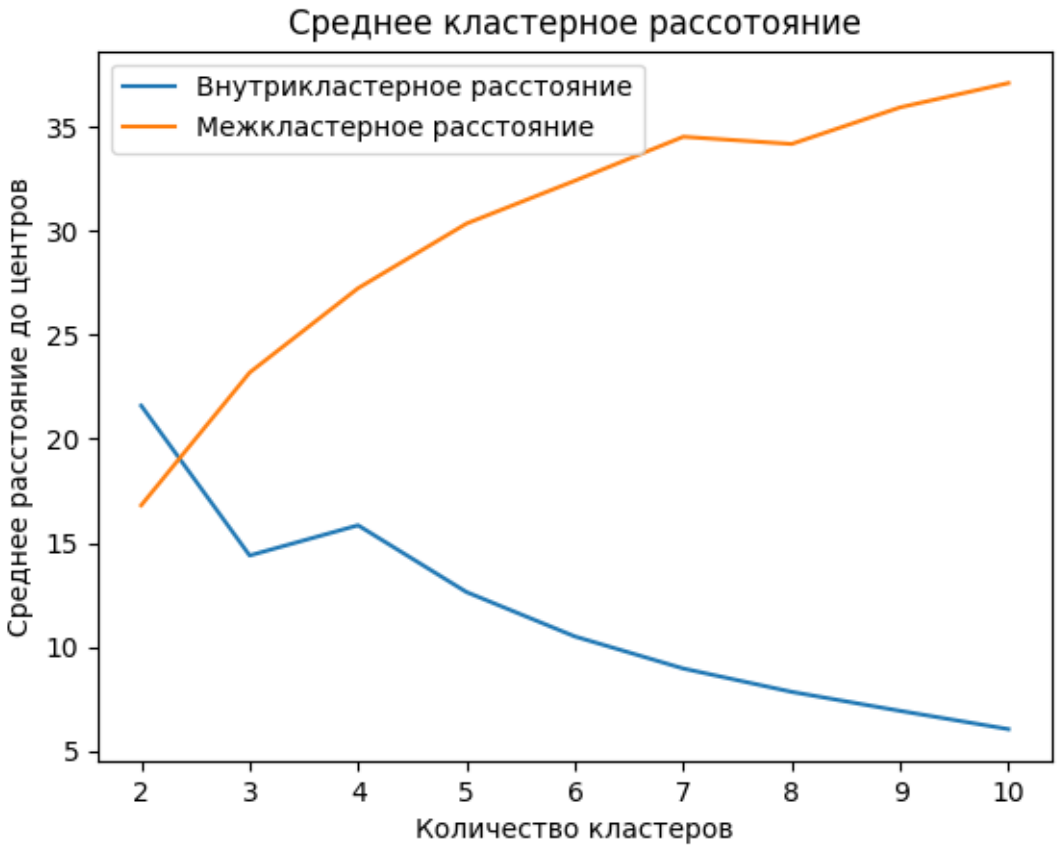

Рисунок 4.4 — Внутрикластерное и межкластерное расстояние для kmeans

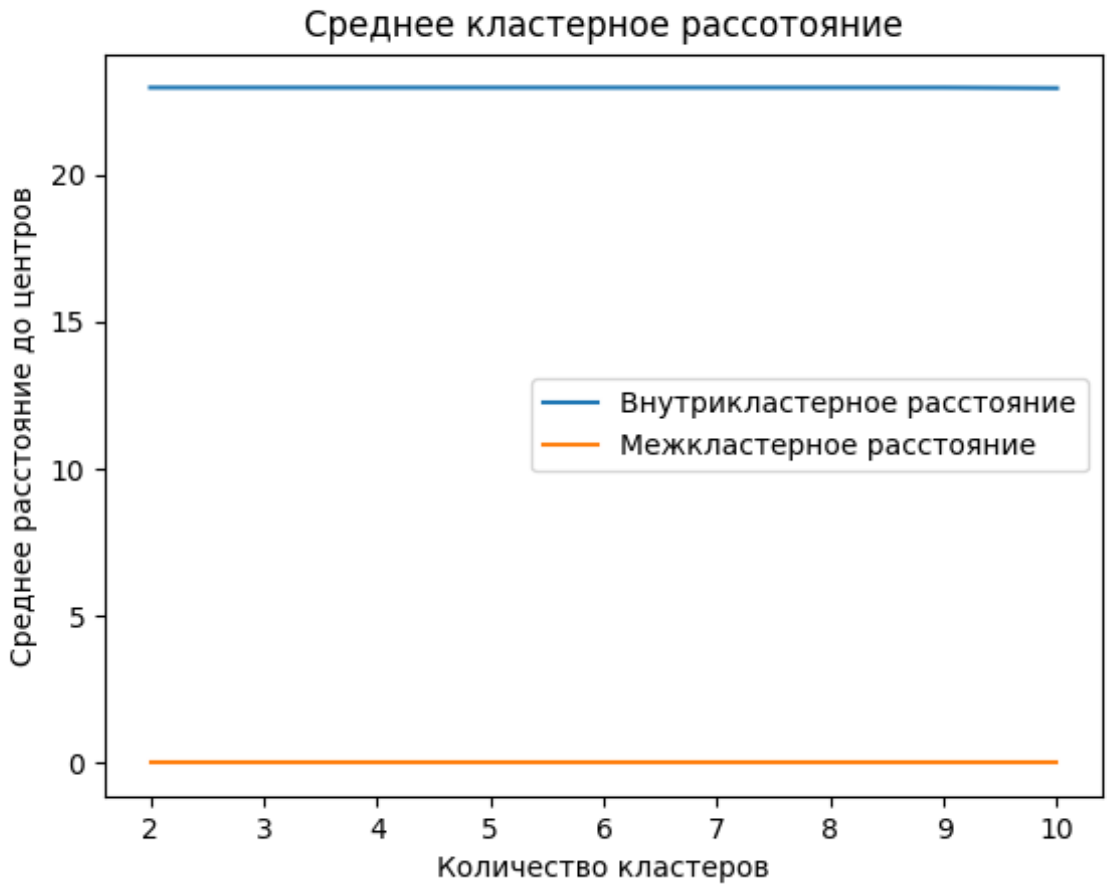

Рисунок 4.5 — Внутрикластерное и межкластерное расстояние для cmeans

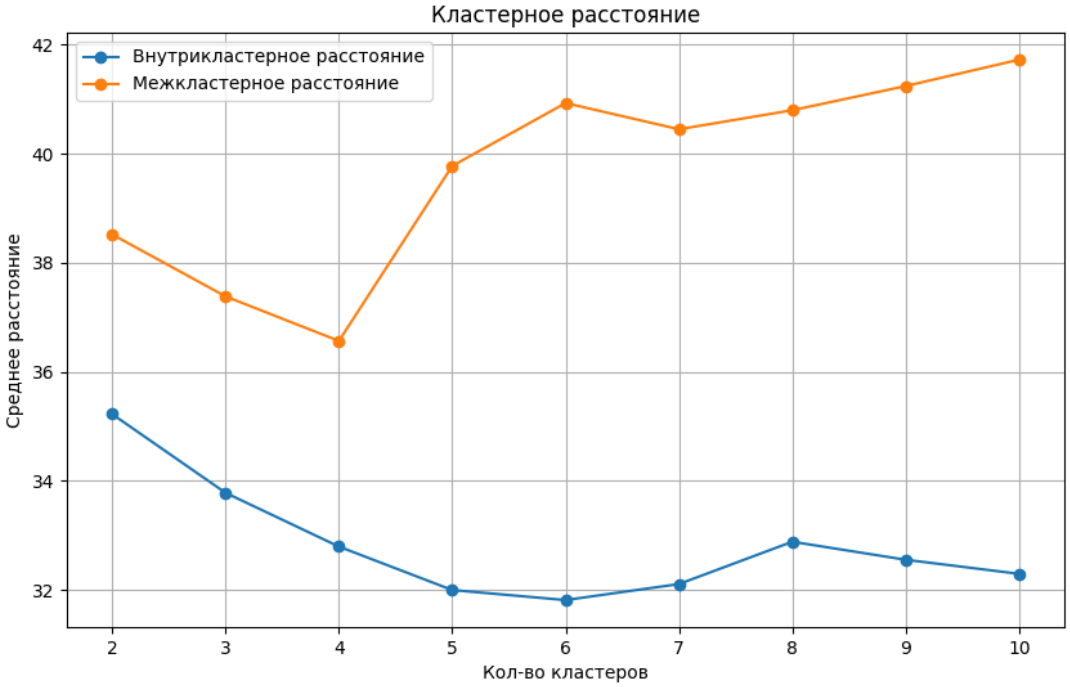

Рисунок 4.6 — Внутрикластерное и межкластерное расстояние для иерархической кластеризации

4.2.2 Кластеры

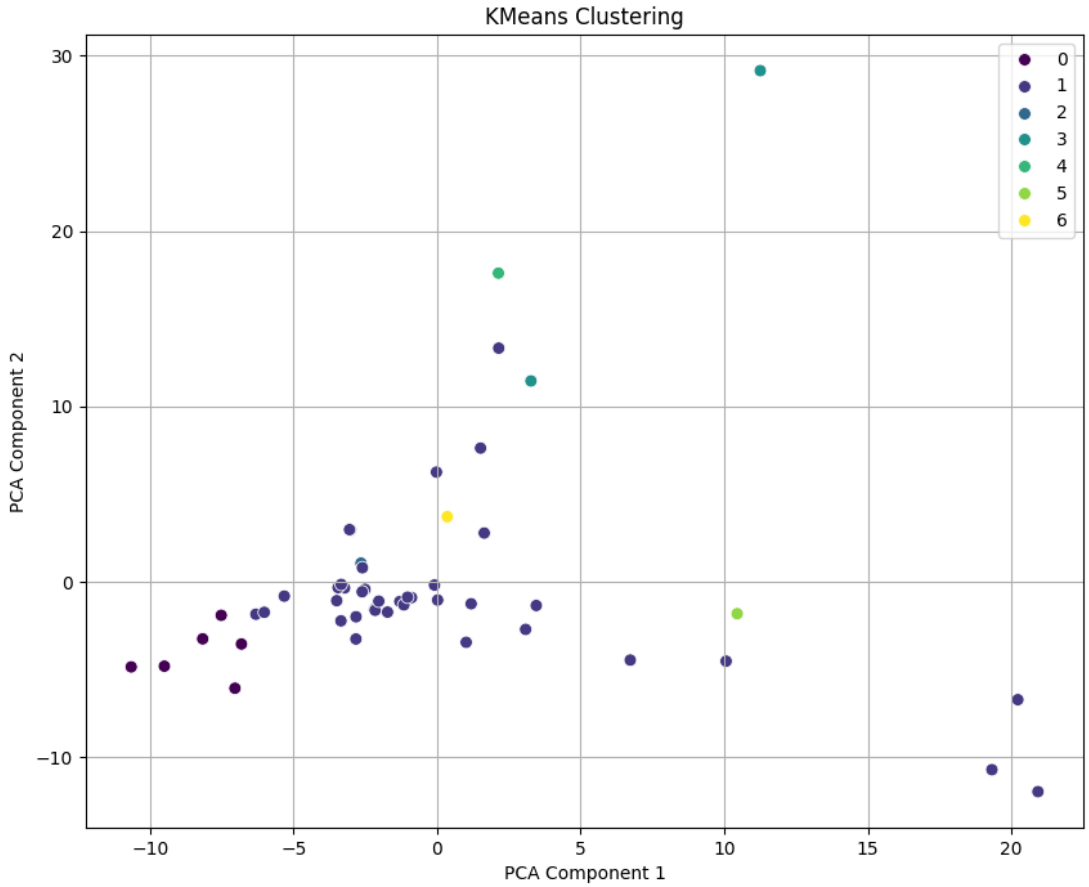

Рисунок 4.7 — Кластеры по методу kmeans (экспертная оценка)

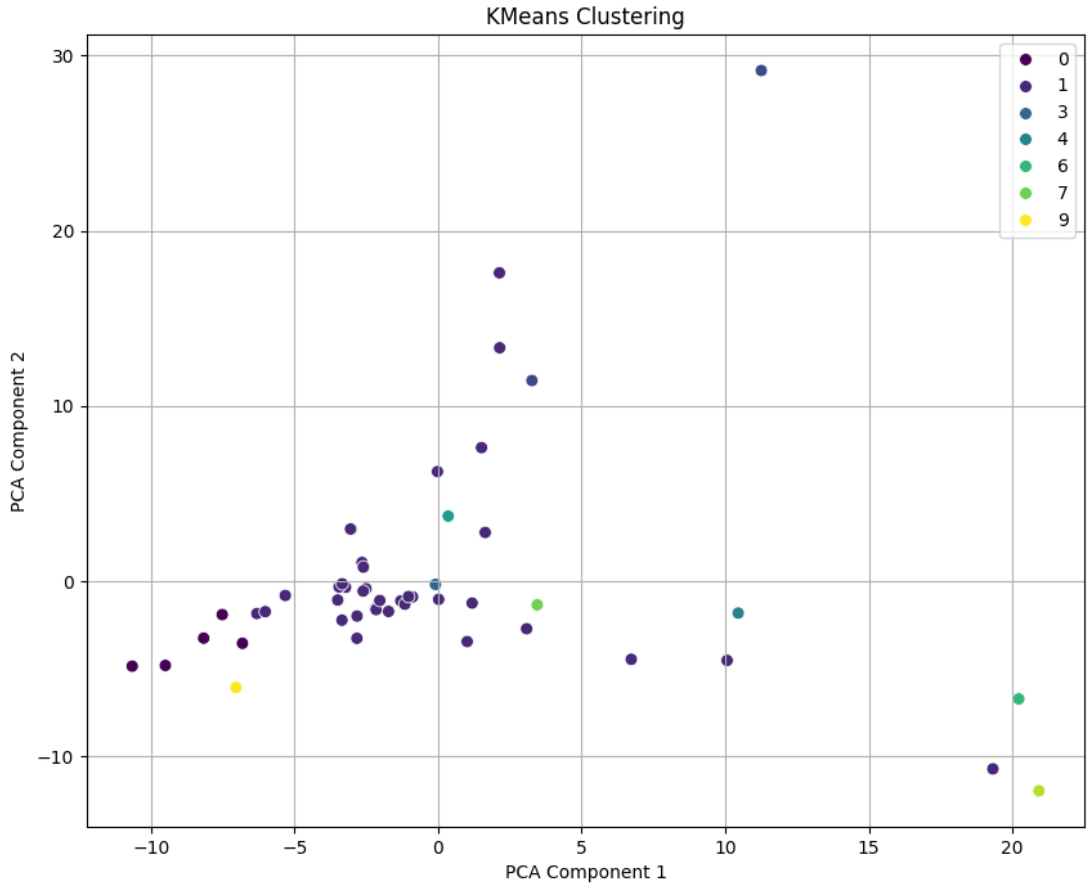

Рисунок 4.8 — Кластеры по методу kmeans (оценка студента)

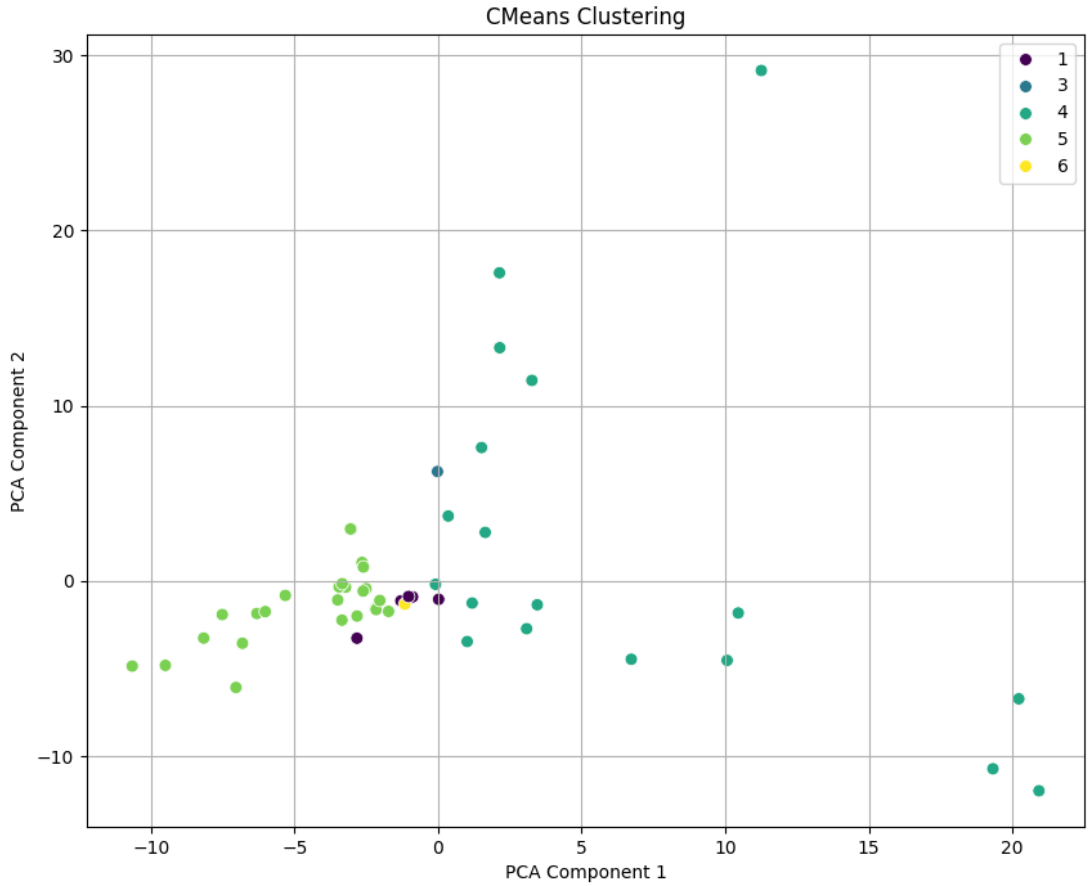

Рисунок 4.9 — Кластеры по методу cmeans (экспертная оценка)

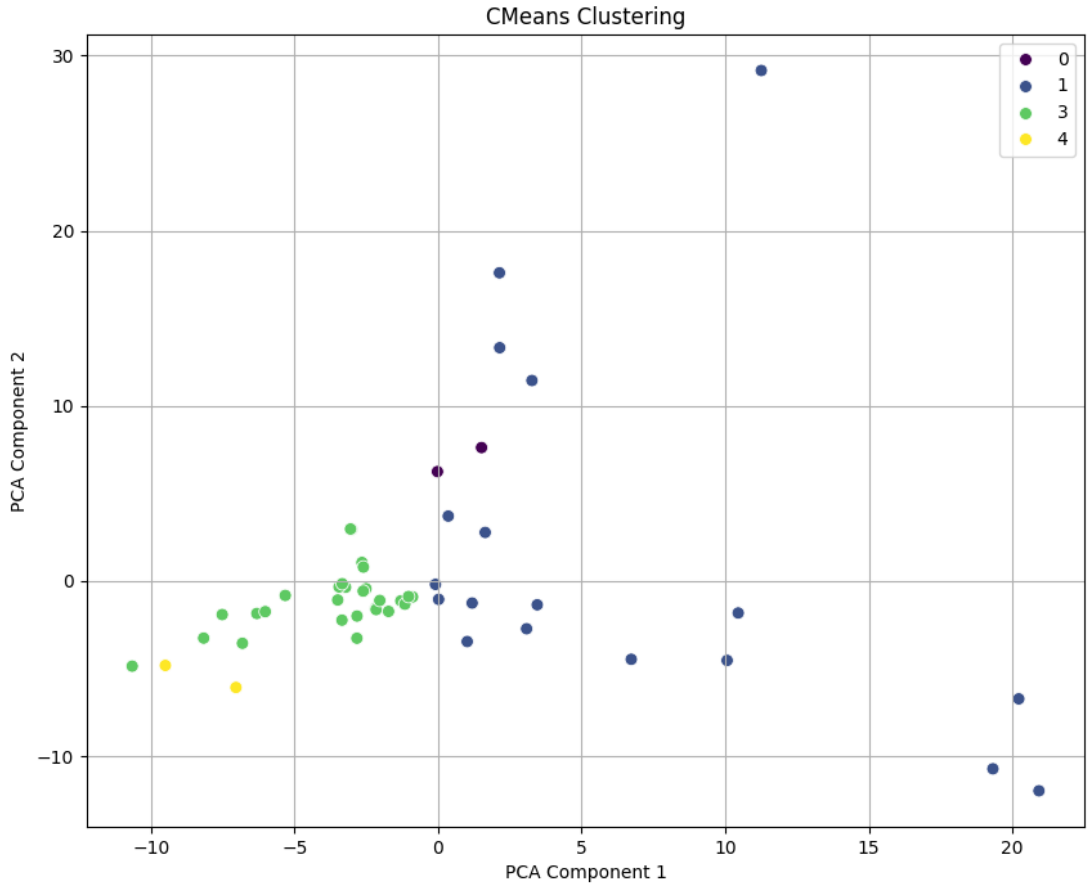

Рисунок 4.10 — Кластеры по методу cmeans (оценка студента)

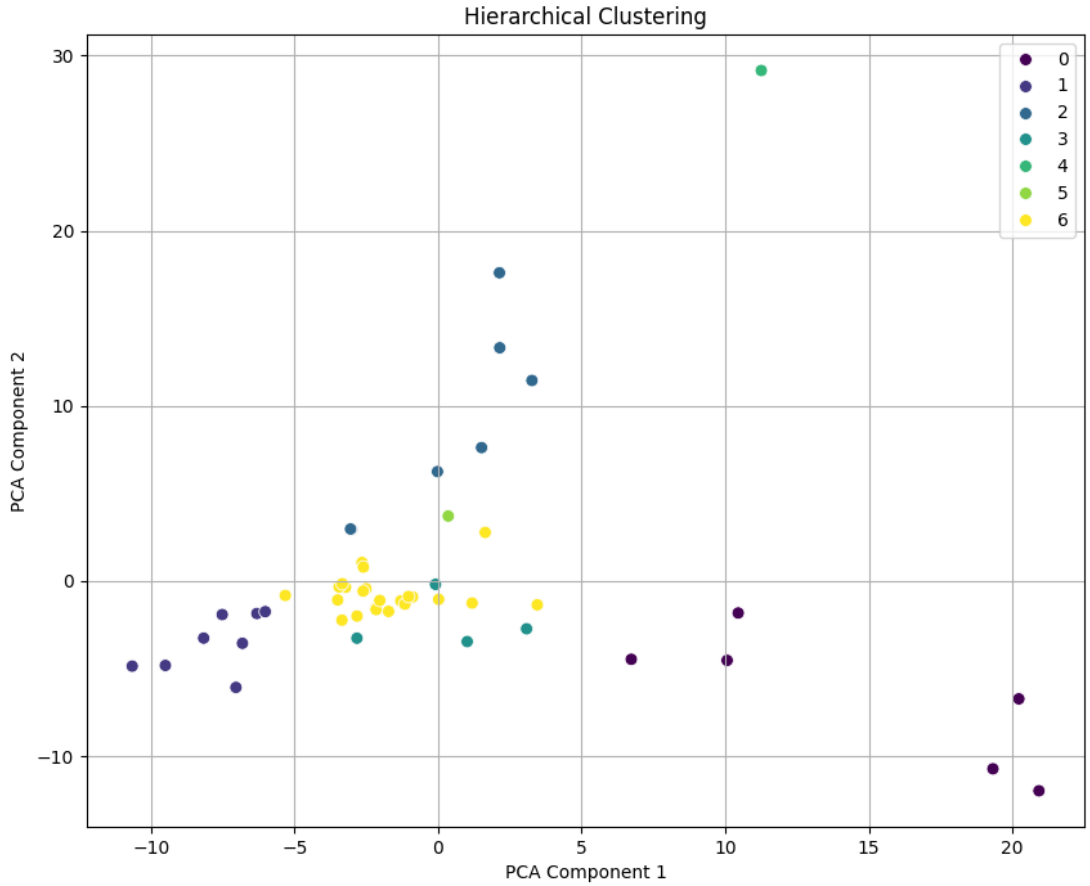

Рисунок 4.11 — Кластеры по методу иерархической класетризации (экспертная оценка)

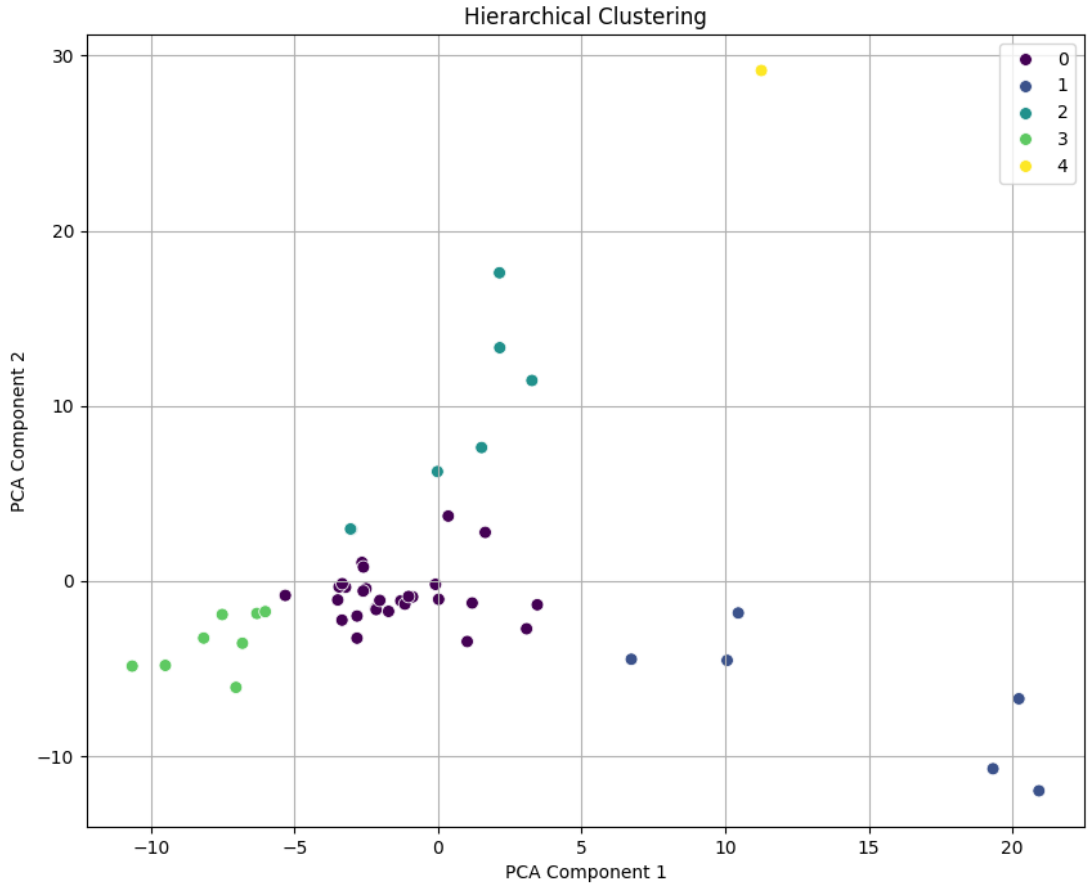

Рисунок 4.12 — Кластеры по методу иерархической класетризации (оценка студента)

4.3 Вывод

В ходе выполнения лабораторной работы были получены графики, которые могут помочь определиться с оптимальным количеством кластеров, а также графики, которые отражают внутрикластерное и межкластерное расстояние. В разделе 4.2.2 представлены результаты кластеризации методами kmeans, cmeans и иерархической кластеризации с количеством кластеров, определённых экспертом и студентом.

ЗАКЛЮЧЕНИЕ

В заключение, проведенный анализ продемонстрировал применение различных методов кластеризации — K-Means, C-Means и иерархической кластеризации — к исследуемому набору данных. Использование метрик внутрикластерного и межкластерного расстояний позволило оценить влияние количества кластеров на компактность и разделение групп данных. Графики, демонстрирующие динамику этих расстояний, а также визуализации самих кластеров, выявили общую тенденцию: увеличение количества кластеров приводит к уменьшению внутрикластерного расстояния, но в определенный момент это уменьшение замедляется, а межкластерное расстояние, достигнув некоторого значения, перестает существенно расти.

Анализ визуализаций кластеров показал, что все три метода кластеризации в целом сформировали похожие кластеры, несмотря на различия в их алгоритмах. Однако, C-Means, как метод нечеткой кластеризации, продемонстрировал большую гибкость в случаях, когда данные имели некоторую степень пересечения. Иерархическая кластеризация предоставила более полное представление о структуре кластеров, позволяя оценить иерархию их объединения. K-Means также показал себя достаточно эффективно, хотя и с некоторой потерей гибкости в сравнении с C-Means.

Оценка оптимального количества кластеров, проведенная как с помощью "метода локтя так и визуального анализа, привела к заключению о том, что разбиение данных на 5 кластеров является наиболее адекватным, что отличается от оценки эксперта в 7 кластеров. Этот выбор позволяет достичь баланса между компактностью кластеров и их разделением.

В целом, полученные результаты подчеркивают эффективность всех примененных методов для кластеризации данных и указывают на то, что при правильной интерпретации и оценке их результатов, они могут быть успешно применены для выявления скрытых закономерностей в данных.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Seaborn: statistical data visualization URL: https://seaborn.pydata.org/ (Дата обращения 05.01.2025)
2. Модуль os в Python, доступ к функциям ОС URL: https://docs-python.ru/standart-library/modul-os-python/ (Дата обращения 05.01.2025)
3. NumPy Documentation URL: https://numpy.org/doc/ (Дата обращения 05.01.2025)
4. Matplotlib 3.9.2 documentation URL: https://matplotlib.org/stable/index.html (Дата обращения 05.01.2025)
5. seaborn: statistical data visualization URL: https://seaborn.pydata.org/ (Дата обращения 05.01.2025)
6. Документация sklearn URL: https://scikit-learn.org/stable/ (Дата обращения 05.01.2025)
7. Документация scipy URL: https://docs.scipy.org/doc/scipy/ (Дата обращения 05.01.2025)
8. URL: https://github.com/refaqtor/fuzzy-c-means-1 (Дата обращения 05.01.2025)
9. Документация Pathlib https://docs.python.org/3/library/pathlib.html (Дата обращения 05.01.2025)
10. Среда разработки Pycharm 2023 Community edition URL: https://www.jetbrains.com/ru-ru/pycharm/download/other.html
11. Язык разработки Python 3.10 URL: https://www.python.org/downloads/release/python-3100/
12. ОС Windows 11 URL: https://www.microsoft.com/ru-ru/software-download/windows11
13. Intel i5-12500H URL: https://www.intel.com/content/www/us/en/products/sku/96141/intel-core-i512500h-processor-18m-cache-up-to-4-50-ghz/specifications.html?wapkw=12500h
14. Алгоритм k-means URL: https://cyberleninka.ru/article/n/analiz-kachestva-elektronnogo-kontenta-razbienie-dannyh-o-potreblenii-kontenta-na-optimalnoe-chislo-klasterov-pri-pomoschi
15. Алгоритм c-means URL: https://cyberleninka.ru/article/n/obrabotka-navigatsionnyh-parametrov-na-osnove-algoritma-nechetkoy-klasterizatsii
16. Алгоритм иерархической кластеризации URL: https://cyberleninka.ru/article/n/optimizatsiya-ierarhicheskoy-klasterizatsii-pri-realizatsii-rekomendatelnyh-sistem