

Исследовательская работа
По теме «Кластеризация данных»

Выполнила

Студентка КММ0-01-23

Плахотина Ю. С.

Студент группы КММО-01-23

Грибань М. С.

2023–2024 Москва

Оглавление

Введение.....	3
1 Генерация распределений	4
1.1 Пример № 1	4
1.2 Пример № 2	7
1.3 Пример № 3	8
1.4 Пример № 4.....	9
2 Кластеризация изображений	10
2.1 Пример № 1 - Кластеризация банана (Reshape)	10
2.2 Пример № 2 – Кластеризация банана (YUV).....	12
3 Генерация изображений	15
3.1 Пример № 1 – make_dna.....	15
3.2 Пример № 2 – make_circle + make_moons (Noise = None)	18
3.3 Пример № 2 – make_circle + make_moons (Noise = 0.1)	19
3.4 Пример № 3 – make_circle + make_moons (Noise=0.2)	20
4 Анализ результатов.....	22
Заключение	23

Введение

В данной исследовательской работе приводятся примеры и результаты исследования на следующих типах данных:

- 1) Генерация данных на основе распределений
- 2) Генерация данных с использованием make-функций
- 3) Загрузка изображений

В ходе исследования каждый набор данных кластеризуется алгоритмами, заложенными в программе и затем, на основе анализа, делается вывод о качестве кластеризации.

В ClustSystem были реализованы такие методы кластеризации, как BIRCH, CURE и ROCK с помощью библиотек pyclustering и scikit-learn.

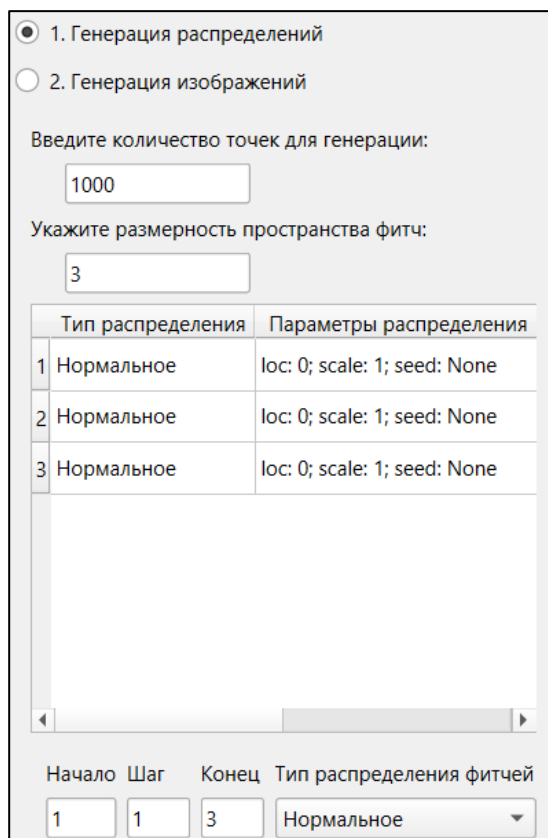
Целью работы является оценка работоспособности и качества кластеризации при генерации и кластеризации данных.

Задача – провести кластеризацию данных различной размерности, поработать с неоднородными данными и рассмотреть влияние задаваемых параметров на конечный результат.

1 Генерация распределений

1.1 Пример № 1

В качестве первого примера приводится генерация дискретных данных на основе распределений.



	Тип распределения	Параметры распределения
1	Нормальное	loc: 0; scale: 1; seed: None
2	Нормальное	loc: 0; scale: 1; seed: None
3	Нормальное	loc: 0; scale: 1; seed: None

Рисунок 1 – входные данные генерации распределения

Сгенерируем 1000 точек в пространстве размерности 3.

Возьмем стандартное нормальное распределение с $\mu = 0$ и $\sigma = 1$.

Сравним результаты работы алгоритмов BIRCH-P и BIRCH-S.

Можно заметить, что при одинаковом количестве кластеров количество элементов в них существенно отличается. Похожая ситуация наблюдается при кластеризации с помощью CURE и ROCK. Дело в том, что данные алгоритмы реализованы с помощью устаревшего pyclustering, а BIRCH-S используется из scikit-learn.

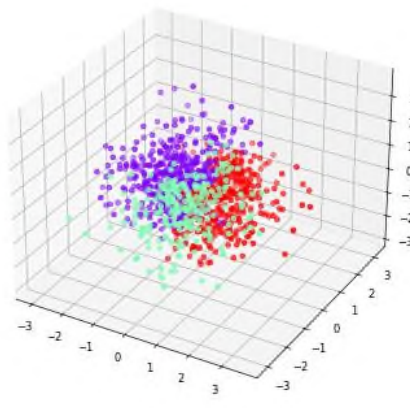
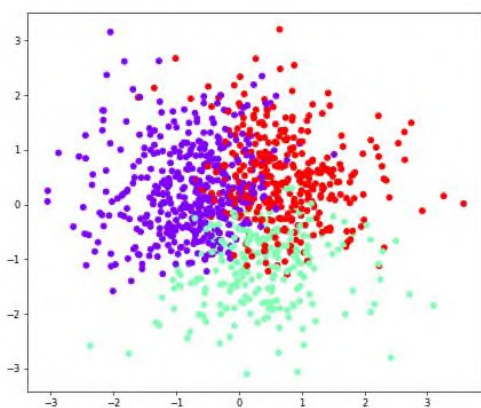


Рисунок 2 – результат кластеризации BIRCH-S алгоритмом

Время работы алгоритма	0.015625
Показатель DunnIndex	0.01788602205943323
Показатель DunnIndexMean	0.421281049512545

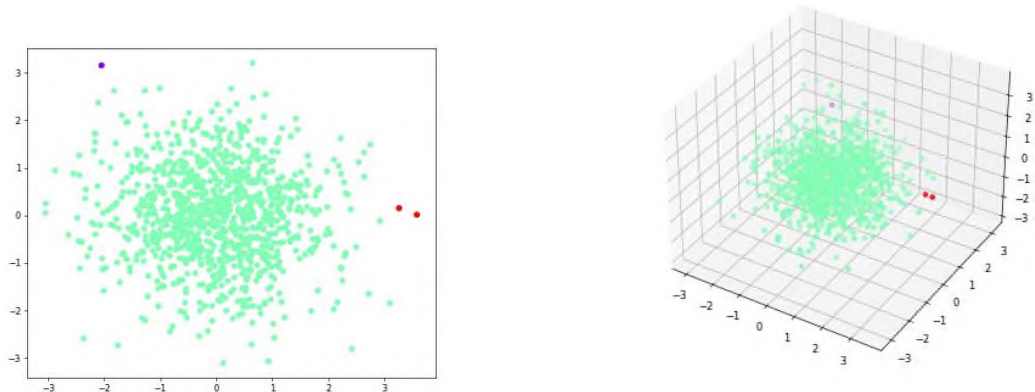


Рисунок 3 – результат кластеризации BIRCH-P алгоритмом

Время работы алгоритма	0.453125
Показатель DunnIndex	0.1474016514473182
Показатель DunnIndexMean	0.5068028560462193

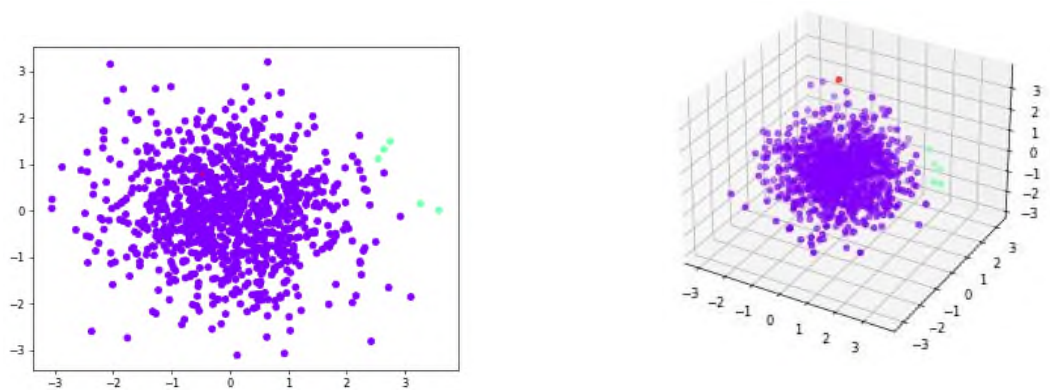


Рисунок 4 – результат кластеризации ROCK алгоритмом

Время работы алгоритма	120.578125
Показатель DunnIndex	0.06046279402787573
Показатель DunnIndexMean	0.47066552694140146

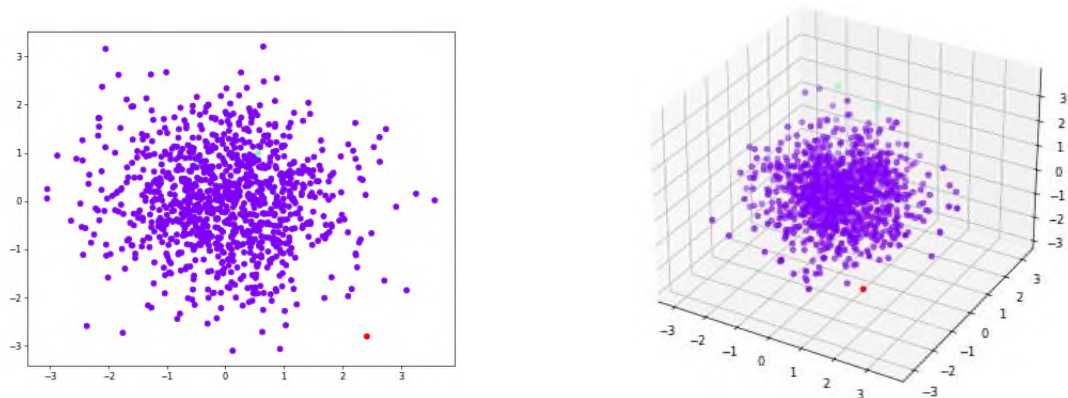


Рисунок 5 – результат кластеризации CURE алгоритмом

Время работы алгоритма	4.359375
Показатель DunnIndex	0.10668970743209803
Показатель DunnIndexMean	0.5269471447786228

Предварительный анализ:

Данные алгоритмы могут работать и с большим количеством точек, однако в рамках данного исследования мы столкнулись с нехваткой вычислительных мощностей, поэтому рассматриваем более малоразмерные примеры.

На данном примере заметим, что BIRCH-S лучше всего показывает себя по времени выполнения. Проблему с несбалансированным количеством элементов в кластерах можно решить, задавая для каждого алгоритма новые параметры.

Алгоритм ROCK имеет высокое время выполнения. Например, на рисунке 4 показано время работы алгоритма ROCK в районе 120.578125 для 1000 точек в трехмерном пространстве. Это свидетельствует о том, что алгоритм является менее эффективным по времени выполнения в сравнении с другими алгоритмами, такими как BIRCH и CURE.

1.2 Пример № 2

При том же распределении и количестве точек, однако с большей размерностью, равной 100 (На изображении приводятся значения только по первым трем размерностям), алгоритмы дали следующий результат:

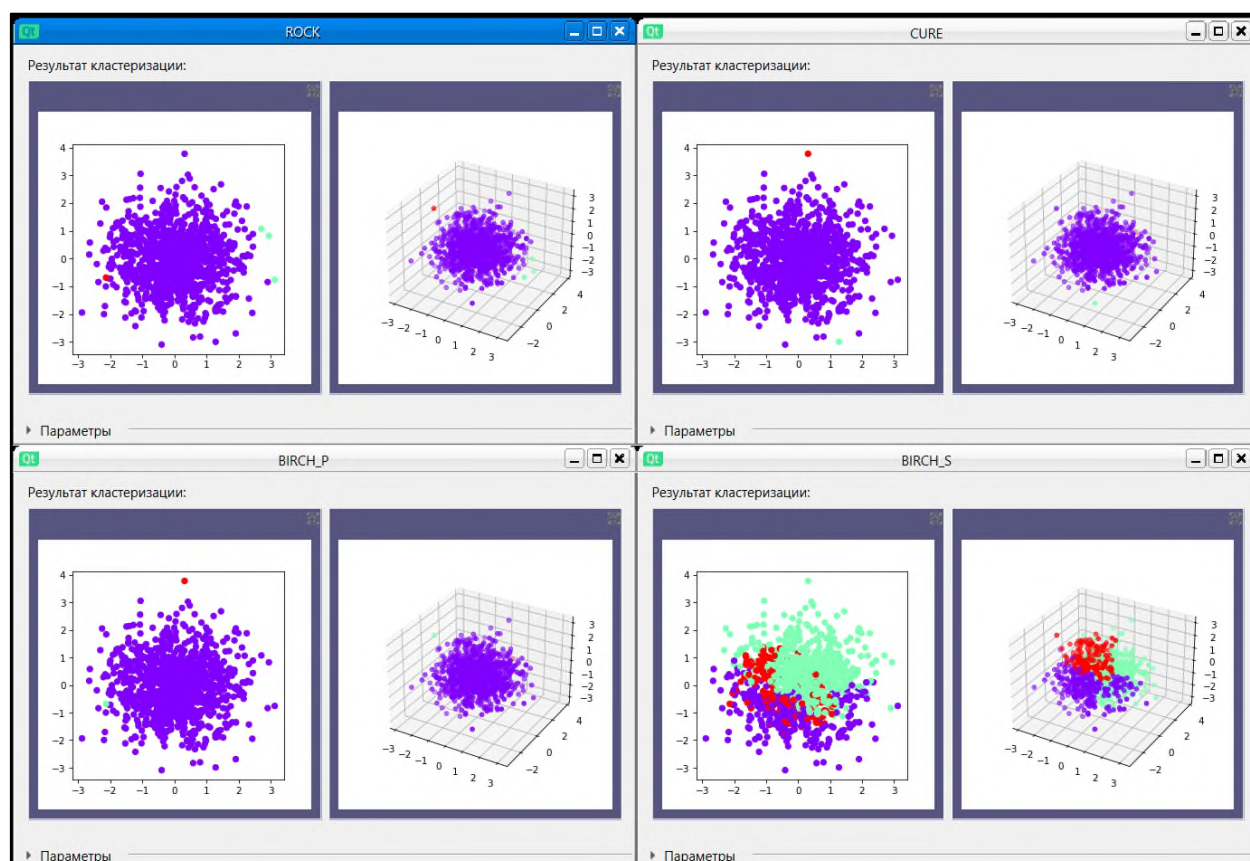


Рисунок 6 – результат кластеризации рассматриваемых алгоритмов

Время работы алгоритма	125.3125
Показатель DunnIndex	0.06569058560357967
Показатель DunnIndexMean	0.4526306510635862

Таблица 1 – результат работы ROCK

Время работы алгоритма	4.75
Показатель DunnIndex	0.1363706126526187
Показатель DunnIndexMean	0.5536874883369137

Таблица 2 – результат работы CURE

Время работы алгоритма	0.46875
Показатель DunnIndex	0.15051303891710913
Показатель DunnIndexMean	0.4969119860692368

Таблица 3 – результат работы BIRCH-P

Время работы алгоритма	0.015625
Показатель DunnIndex	0.012461836985087545
Показатель DunnIndexMean	0.3466501420510158

Таблица 4 – результат работы BIRCH-S

Предварительный анализ:

Сильных отличий не наблюдается,

1.3 Пример № 3

Попробуем сравнить CURE и BIRCH-P на ещё большей размерности. Увеличим размерность пространства до 1000. ROCK уберем из рассмотрения, поскольку он уже показывает худший результат.

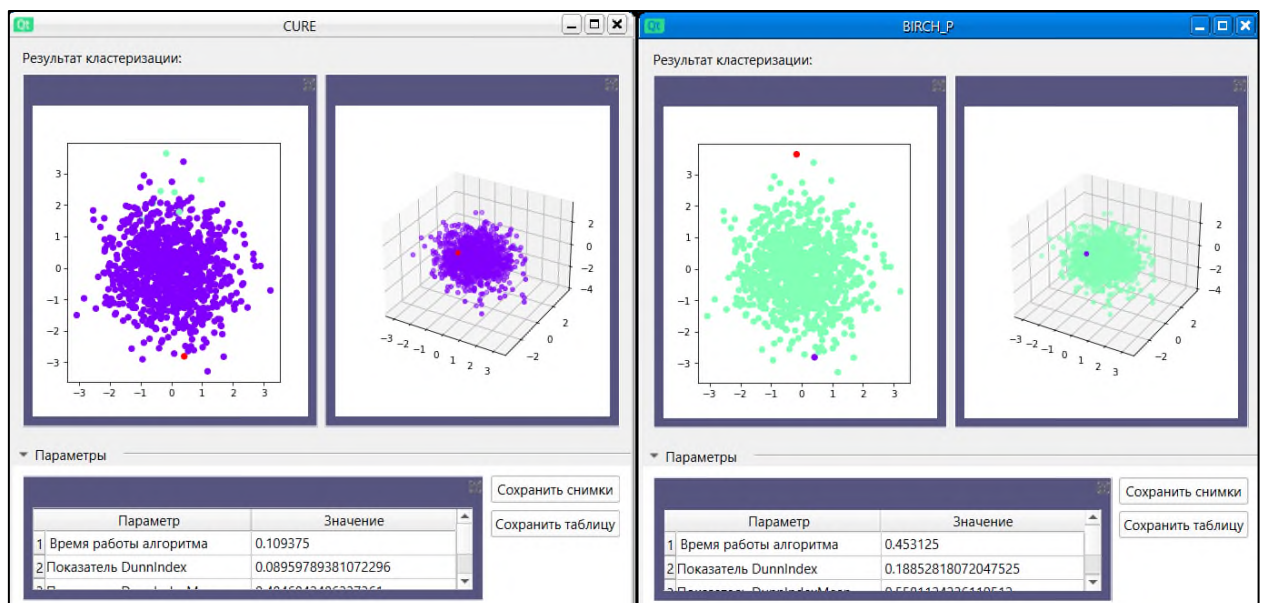


Рисунок 7 – результат кластеризации CURE и BIRCH-P

Предварительный анализ:

CURE, который ранее работал медленнее, чем BIRCH-P, на большей размерности показал лучший результат.

При увеличении размерности до 10 000, CURE справился за 0.0625 секунд, а BIRCH-P за 0.4375.

1.4 Пример № 4

Однако если взять в рассмотрение BIRCH-S, то данный алгоритм показывает лучший результат, что говорит скорее о не совершенности реализации BIRCH-P, нежели о недостатке самого алгоритма.

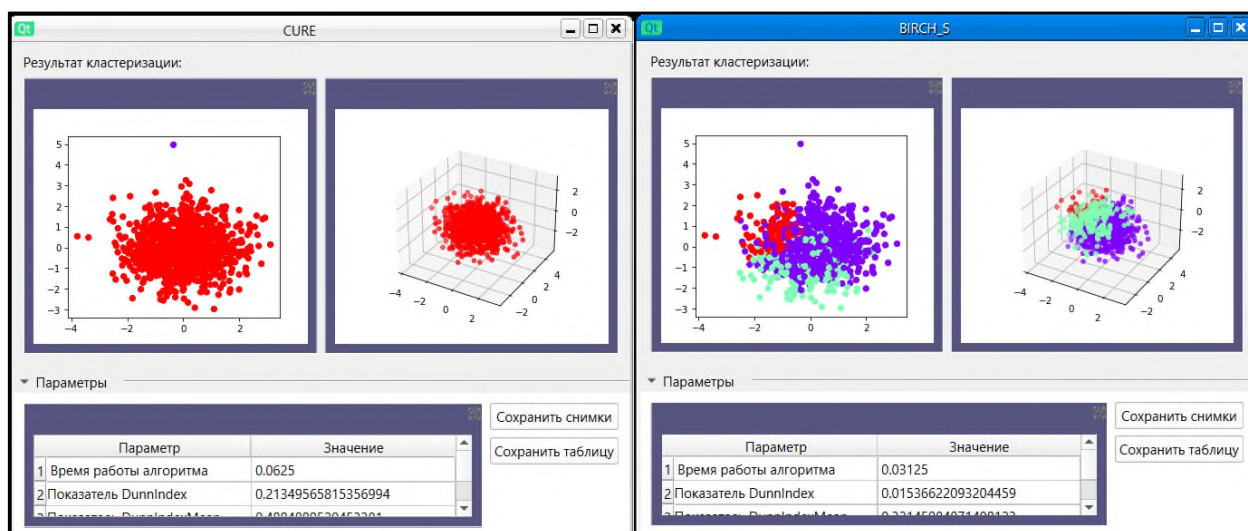


Рисунок 8 – результат кластеризации CURE и BIRCH-S

Предварительный анализ:

На представленном Рисунке 8 наблюдаются результаты кластеризации с применением алгоритмов CURE и BIRCH-S. Важно отметить, что визуально BIRCH-S демонстрирует более четкую структуру кластеров по сравнению с CURE и с меньшим временем работы алгоритма.

2 Кластеризация изображений

В данном разделе рассматривается кластеризация png-изображений.

2.1 Пример № 1 - Кластеризация банана (Reshape)

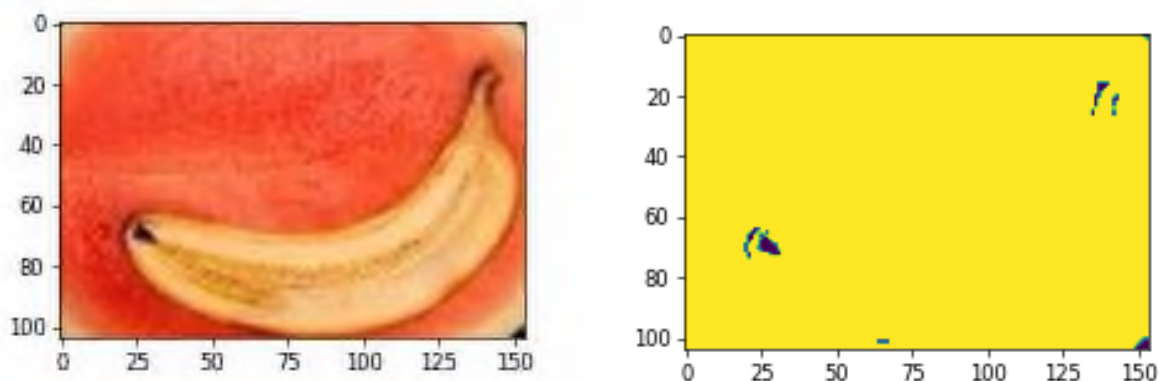


Рисунок 9 – результат кластеризации изображения CURE алгоритмом

Время работы алгоритма	14.671875
------------------------	-----------

Поскольку подсчет расстояния между кластерами при обработке изображений рассматриваемыми алгоритмами требует большого количества времени, показатели DunnIndex и DunnIndexMean в данных примерах опустим.

Также не будет рассмотрен алгоритм ROCK, поскольку в данных условиях он не производителен и в общем случае виснет система при его использовании при кластеризации изображений.



Рисунок 10 – результат кластеризации изображения BIRCH-P алгоритмом

Время работы алгоритма	30.828125
------------------------	-----------

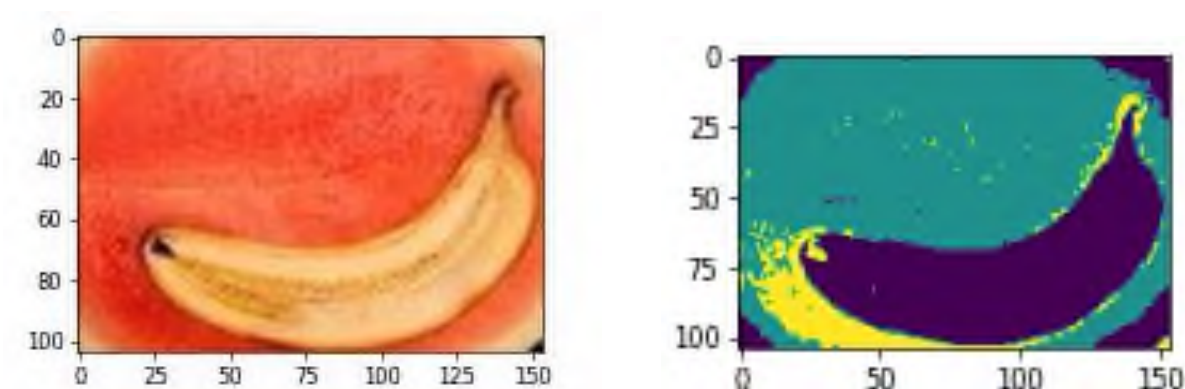


Рисунок 11 – результат кластеризации изображения *BIRCH-S* алгоритмом

Время работы алгоритма	5.8125
------------------------	--------

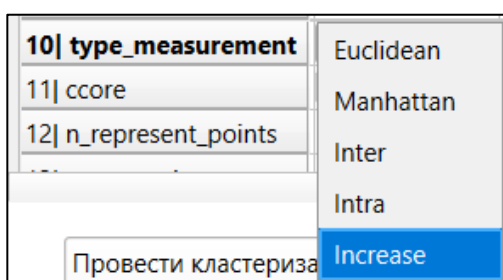


Рисунок 12 – смена метрики

Попробуем улучшить результат работы алгоритма *BIRCH-P*. Для этого, вместо Евклидовой, выберем другую метрику в задаваемых параметрах кластеризации.

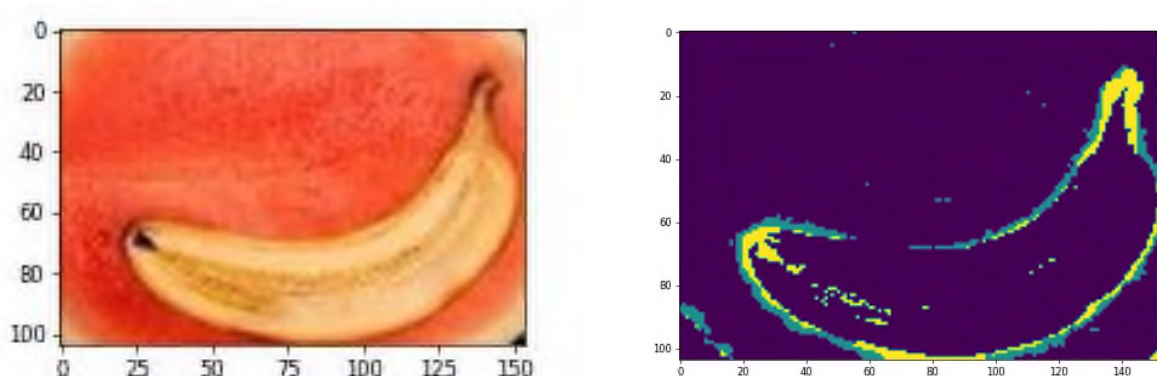


Рисунок 13 – результат кластеризации изображения *BIRCH-P* алгоритмом

Время работы алгоритма	184.828125
------------------------	------------

Предварительный анализ:

Время выполнения после смены метрики увеличилось в 6 раз.

На Рисунке 10: Визуальный анализ результатов кластеризации с BIRCH-P позволяет оценить структуру кластеров. Однако, по сравнению с CURE, время выполнения значительно выше, что может указывать на потенциальные проблемы с производительностью. На Рисунке 11: BIRCH-S продемонстрировал лучшую производительность по сравнению с BIRCH-P.

На Рисунке 12: Для улучшения работы алгоритма BIRCH-P была проведена смена метрики на изображении, использование другой метрики привело к значительному увеличению времени выполнения (184.828125 секунд).

2.2 Пример № 2 – Кластеризация банана (YUV)

Попробуем изменить тип конвертации изображения на YUV.

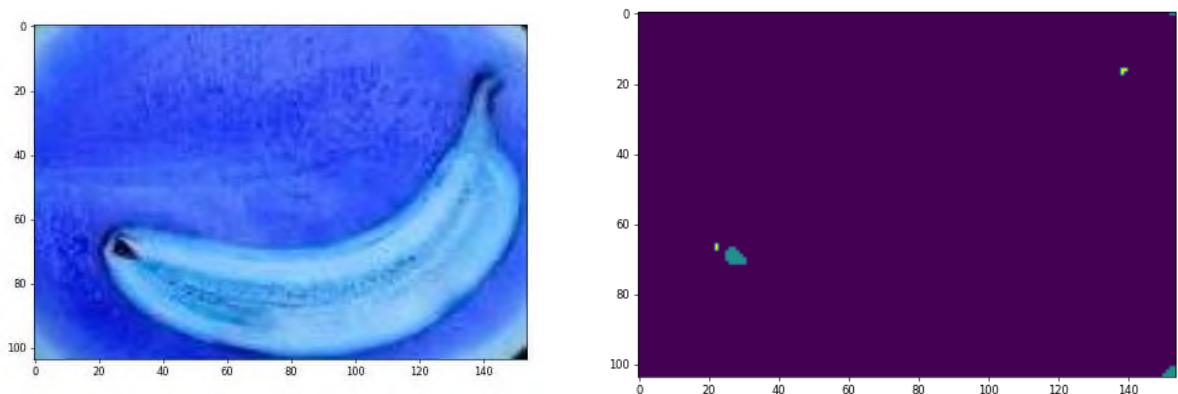


Рисунок 14 – результат кластеризации изображения BIRCH-P алгоритмом

Время работы алгоритма	20.78125
------------------------	----------

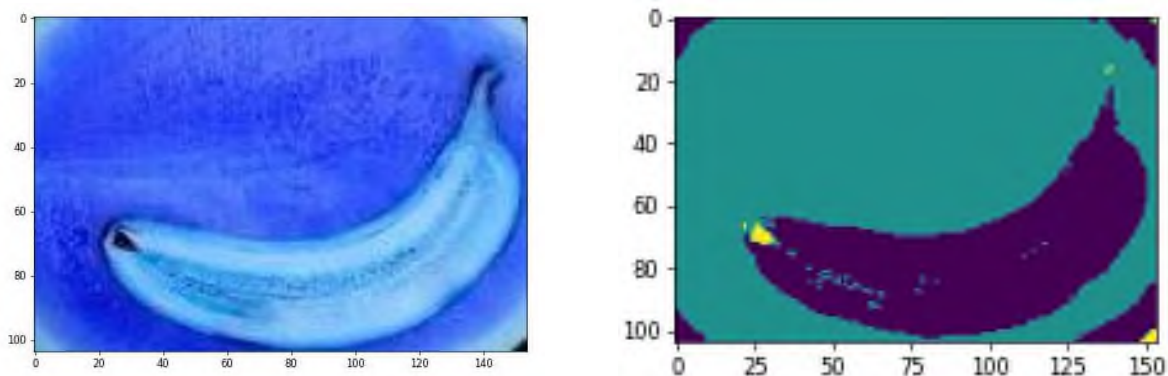


Рисунок 15 – результат кластеризации изображения CURE алгоритмом

Время работы алгоритма	13.171875
------------------------	-----------

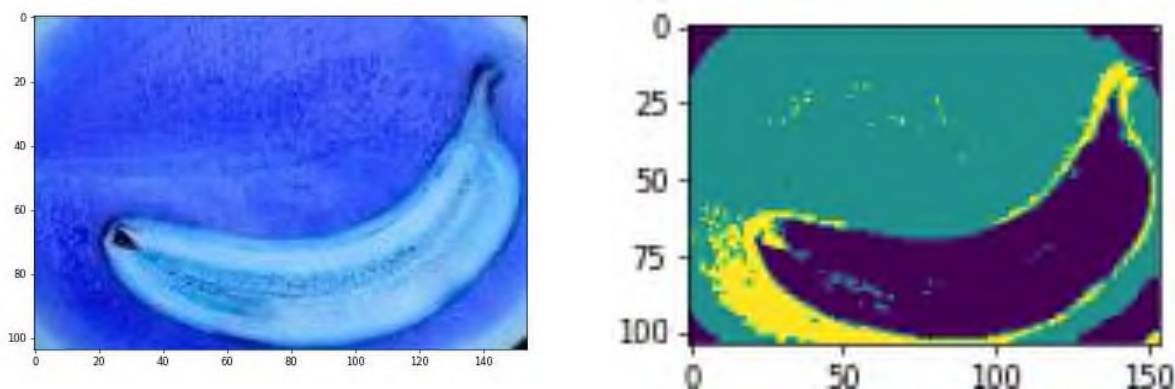


Рисунок 16 – результат кластеризации изображения BIRCH-S алгоритмом

Время работы алгоритма	2.671875
------------------------	----------

Предварительный анализ:

BIRCH, реализованный на pyclustering, все так же плохо справляется с поставленной задачей. А вот CURE показывает уже более приемлемый результат.

В рассмотренных примерах ROCK показывает менее эффективные результаты по времени выполнения по сравнению с другими алгоритмами, особенно при обработке изображений. Это может стать ограничивающим фактором при работе с данными больших размеров.

CURE эффективно обрабатывает данные сложной формы, выделяя кластеры, представляющие банан.

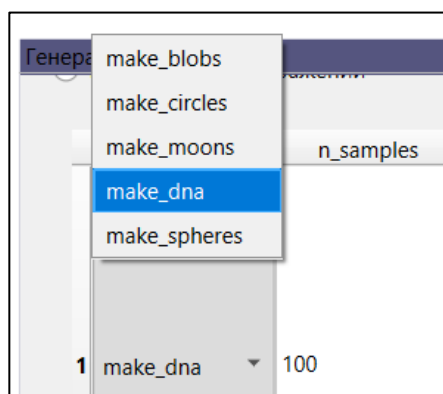
BIRCH-P и BIRCH-S продемонстрировали способность к кластеризации данной формы банана с преимуществом у BIRCH-S.

Важно отметить, что BIRCH-S показывает более сбалансированные кластеры в сравнении с BIRCH-P.

3 Генерация изображений

В данном разделе приводятся примеры генерации и кластеризации данных на основе make-функций.

3.1 Пример № 1 – make_dna



Теперь сгенерируем данные в виде спирали ДНК при помощи make_dna с количеством точек генерации равном 100.

Рисунок 17 – варианты генерации возможных изображений

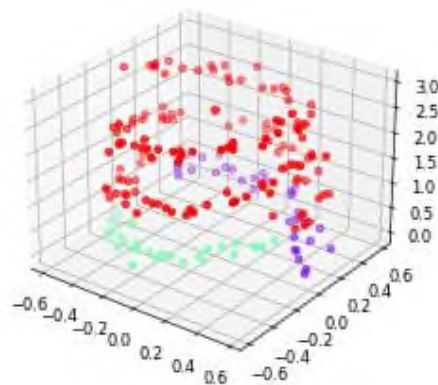
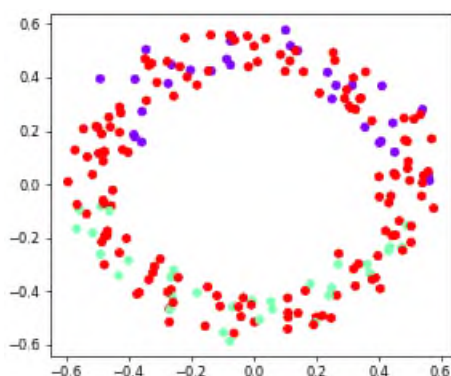


Рисунок 18 – результат кластеризации CURE алгоритмом

Время работы алгоритма	0.0
Показатель DunnIndex	0.06017677934159759
Показатель DunnIndexMean	0.4063039186866146

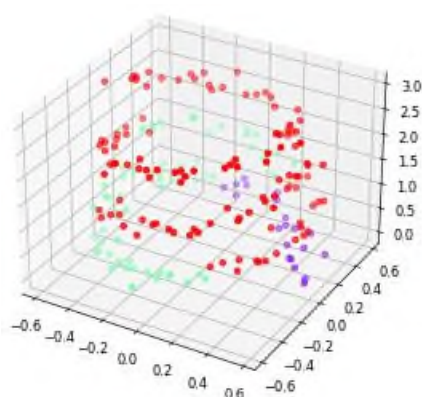
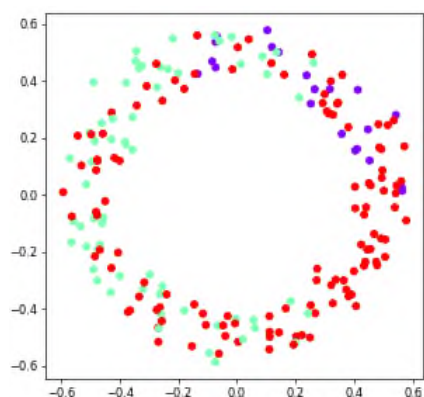


Рисунок 19 – результат кластеризации BIRCH-R алгоритмом

Время работы алгоритма	0.015625
Показатель DunnIndex	0.03312069932852782
Показатель DunnIndexMean	0.4548434130839042

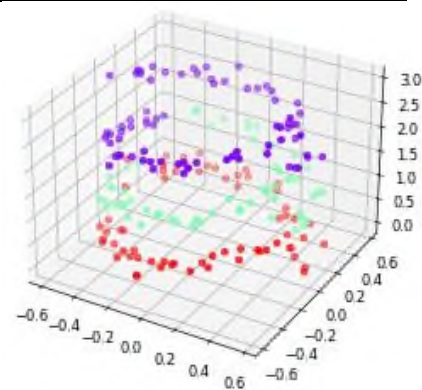
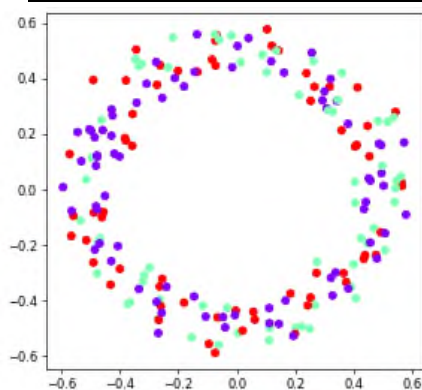


Рисунок 20 – результат кластеризации BIRCH-S алгоритмом

Время работы алгоритма	0.0
Показатель DunnIndex	0.04650108916846331
Показатель DunnIndexMean	0.7317184756117385

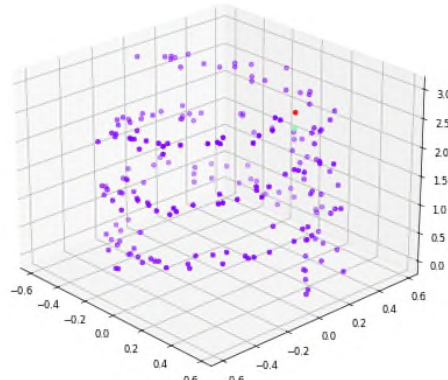
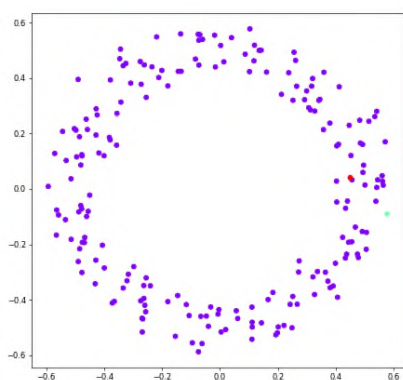


Рисунок 21 – результат кластеризации ROCK алгоритмом

Время работы алгоритма	0.078125
Показатель DunnIndex	0.031742586889493135
Показатель DunnIndexMean	0.05797538276240025

Результат алгоритма ROCK можно улучшить, если в качестве параметра радиуса связности взять значение $\text{eps} = 0.5$.

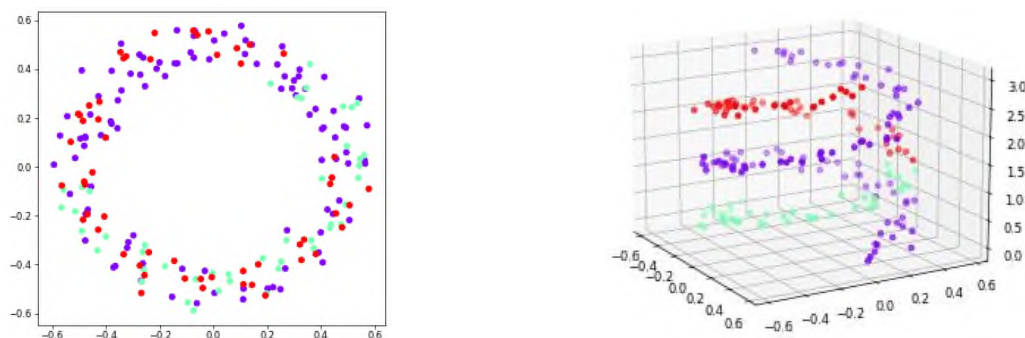


Рисунок 22 – результат кластеризации ROCK алгоритмом при $\text{eps} = 0.5$

Время работы алгоритма	0.09375
Показатель DunnIndex	0.04483810491624281
Показатель DunnIndexMean	0.41606339335052295

Предварительный анализ:

Все алгоритмы справились с поставленной задачей, однако разбиение на кластеры у каждого алгоритма свое. Рисунок 18: Алгоритм CURE успешно кластеризует данные, продемонстрировав хорошие показатели качества кластеризации.

Рисунок 20: BIRCH-S показывает высокие показатели качества кластеризации, подчеркивая свою эффективность в данном примере. Рисунок 21: Алгоритм ROCK показывает удовлетворительные результаты, но его можно улучшить, изменяя параметры, например радиус связности.

Рисунок 22: После оптимизации параметров, ROCK показывает улучшенные результаты. Рисунок 23: В данном примере все алгоритмы эффективно справляются с кластеризацией, но наблюдаются различия в разбиении на кластеры.

Все алгоритмы успешно решают поставленную задачу генерации и кластеризации данных, но различия в разбиении на кластеры свидетельствуют о том, что каждый алгоритм имеет свои особенности и подходит для разных типов данных.

3.2 Пример № 2 – make_circle + make_moons (Noise = None)

Рассмотрим теперь случай, когда сгенерированы одновременно кругов и «луны». Сместим круги по оси z на 0.1.

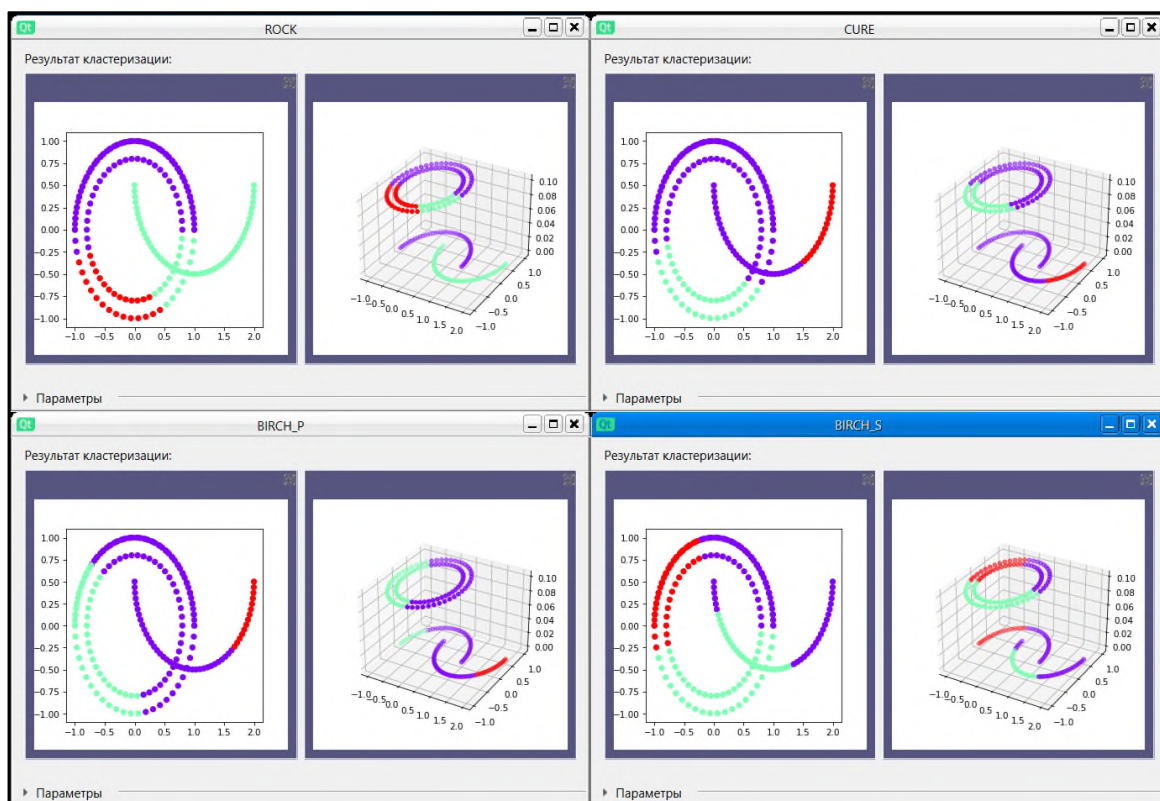


Рисунок 23 – результат кластеризации рассматриваемых алгоритмов

Предварительный анализ:

Все алгоритмы справились с поставленной задачей, однако разбиение на кластеры у каждого алгоритма свое. На основе теории видно, что Birch плохо кластеризует вогнутые множества в отличие от Rock. Если анализировать Cure, то он немного лучше Birch, но хуже Rock.

3.3 Пример № 2 – make_circle + make_moons (Noise = 0.1)

Попробуем добавить шум при генерации изображений. Создадим данные в виде окружности с 1000 точками. Параметр $Noise = 0.1$.

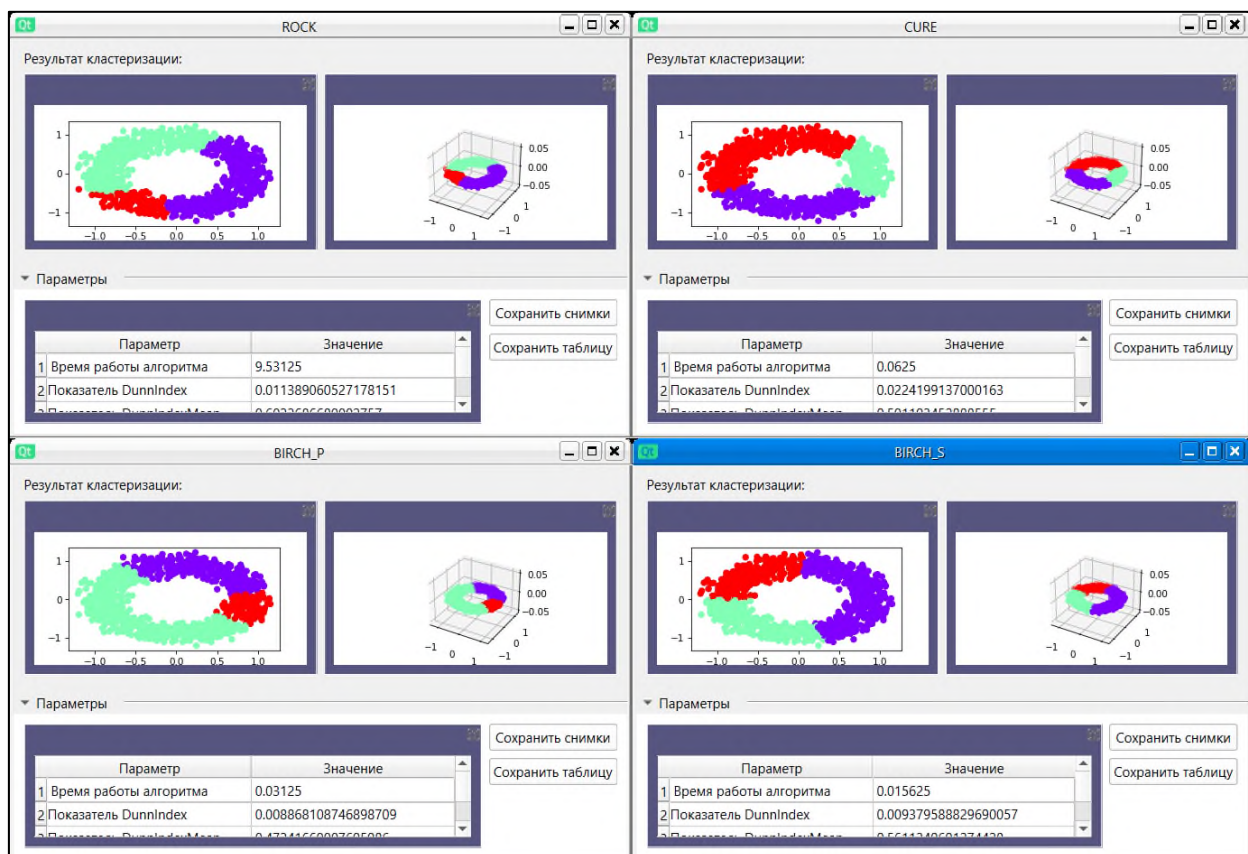


Рисунок 24 – результат кластеризации рассматриваемых алгоритмов

Предварительный анализ:

ROCK, как и в примерах, показанных ранее, все ещё выдает худший результат по времени. У BIRCH-P, так же, как и у ROCK, наблюдается неравномерное количество точек в кластерах.

BIRCH может сталкиваться с проблемами при обработке шума и несбалансированными данными, что отображается в неравномерном распределении элементов в кластерах.

При работе с данными высокой размерности алгоритмы могут давать различные результаты.

3.4 Пример № 3 – make_circle + make_moons (Noise=0.2)

Сгенерируем новые данные: те же окружности, только с параметром $Noise = 0.2$ и количеством точек равном 100. Добавим к ним «луны» с 1000 точек.

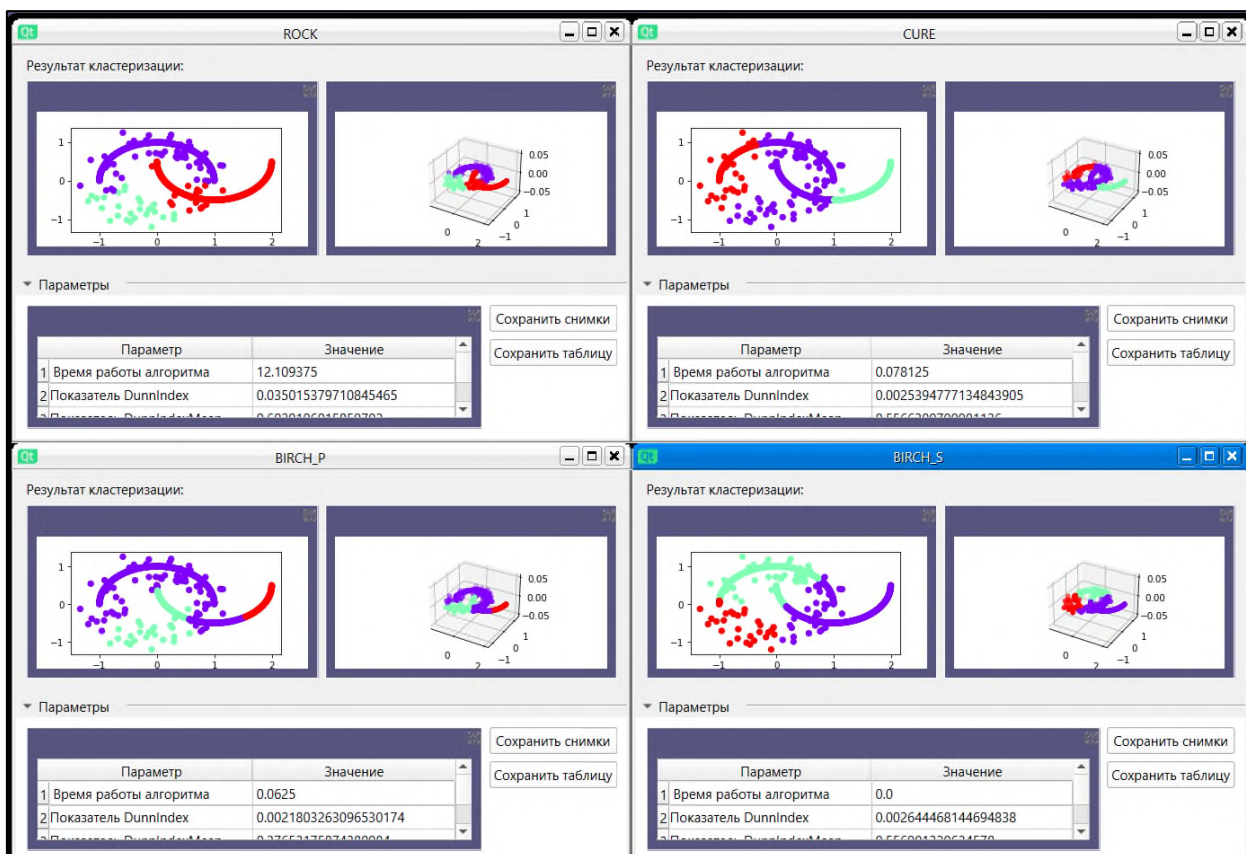


Рисунок 25 – результат кластеризации рассматриваемых алгоритмов

Предварительный анализ:

В рассмотренных примерах с добавлением шума к данным (окружности с параметром $Noise = 0.1$ и $Noise = 0.2$), CURE проявил способность эффективно обрабатывать выбросы и шум.

В контексте времени выполнения CURE также показал себя весьма эффективным. В частности, при увеличении размерности до 1000, CURE справился за 0.0625 секунд, в то время как BIRCH-P занял 0.4375 секунд.

ROCK, как иерархический алгоритм, обладает высокой вычислительной сложностью и может быть медленным при работе с большими объемами данных.

CURE, также являющийся иерархическим алгоритмом, проявил способность эффективно обрабатывать выбросы и шум, что делает его эффективным в таких сценариях.

4 Анализ результатов

После проведения дополнительных исследований ПО были получены следующие выводы о работе рассматриваемых алгоритмов:

ROCK – это иерархический алгоритм кластеризации, который имеет дело не с метрикой, а с такими понятиями, как связи и категориальные типы данных, к которым, например, можно отнести транзакции покупок в каком-либо магазине. Он спроектирован специально для работы с категориальными данными, что делает его эффективным в таких случаях. Иерархическая структура может быть полезной для анализа.

Однако данный алгоритм имеет высокую вычислительную сложность и может быть более медленным по сравнению с оставшимися алгоритмами при работе с большими объемами данных.

Также было показано, что ROCK требует тщательной настройки параметров.

CURE проявил способность эффективно обрабатывать выбросы и шум в данных, а также обнаруживать кластеры сложной формы.

Тем не менее, требует внимательного выбора числа представителей и может потребовать больше вычислительных ресурсов.

Алгоритм CURE также относится к иерархическим алгоритмам кластеризации, однако, в отличие от BIRCH, является алгоритмом кластеризации «сверху вниз», т.е. он разделяет одно общее множество на отдельно заданное количество кластеров.

По отношению к BIRCH является более затратным по производительности. Поскольку в данном алгоритме кластер заменяется выпуклой оболочкой его точек, а посчитать её для многомерного

пространства – это задача экспоненциальной сложности, CURE следует применять для небольших размерностей.

BIRCH показал хорошую масштабируемость и эффективность при работе с данными низкой размерности, но может быть менее эффективным при обнаружении сложных форм кластеров. Чувствителен к таким параметрам, как радиус и порог, что требует тщательной настройки.

Это иерархический алгоритм кластеризации «снизу вверх» основанный на построении CF-tree. Он показывает хороший результат при кластеризации выпуклых множеств (эллипсоиды). В модели алгоритма заложено, что кластеры формируются сферами, поэтому с шарообразными кластерами BIRCH справляется лучше всего.

По сравнению с CURE и ROCK является более производительным. Также стоит отметить, что реализация из `pyclustering` библиотеки является менее удачной по качеству кластеризации, чем реализация из библиотеки `scikit-learn`, что подтверждается результатами, показанными ранее в данном исследовании. Однако не удалось найти ссылок на источники, по которым был реализован BIRCH из `sklearn-learn`, в отличие от `pyclustering`,

Заключение

В данном ПО реализован выбор параметров генерации и кластеризации, а также присутствуют данные для анализа результатов, что удовлетворяет заявленным требованиям и может в дальнейшем использоваться для кластеризации.