МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №6
по дисциплине «Машинное обучение»
Тема: Кластеризация (DBSCAN, OPTICS)

Студент гр. 8303 _____ Гришин К. И.
Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2021

Цель работы

Ознакомиться с методами кластеризации из библиотеки *Sklearn*.

Ход выполнения работы

Загрузка данных

1. Скачать датасет по ссылке:

   https://www.kaggle.com/arjunbhasin2013/ccdata.

2. Загрузить данные в датафрейм (табл. 1)

	0	1	2	4	5
BALANCE	40.901	3202.467	2495.149	817.714	1809.829
BALANCE_FREQUENCY	0.818	0.909	1.000	1.000	1.000
PURCHASES	95.40	0.00	773.17	16.00	1333.28
ONEOFF_PURCHASES	0.00	0.00	773.17	16.00	0.00
INSTALLMENTS_PURCHASES	95.40	0.00	0.00	0.00	1333.28
CASH_ADVANCE	0.000	6442.945	0.000	0.000	0.000
PURCHASES_FREQUENCY	0.167	0.000	1.000	0.083	0.667
ONEOFF_PURCHASES_FREQUENCY	0.000	0.000	1.000	0.083	0.000
PURCHASES_INSTALLMENTS_FREQUENCY	0.083	0.000	0.000	0.000	0.583
CASH_ADVANCE_FREQUENCY	0.00	0.25	0.00	0.00	0.00
CASH_ADVANCE_TRX	0	4	0	0	0
PURCHASES_TRX	2	0	12	1	8
CREDIT_LIMIT	1000.0	7000.0	7500.0	1200.0	1800.0
PAYMENTS	201.802	4103.033	622.067	678.335	1400.058
MINIMUM_PAYMENTS	139.510	1072.340	627.285	244.791	2407.246
PRC_FULL_PAYMENT	0.000	0.222	0.000	0.000	0.000
TENURE	12	12	12	12	12

Таблица 1. Первые пять наблюдений. Наблюдения представлены столбцами.

DBSCAN

1. Проведена кластеризация методом k-средних (рис. 1)

Рисунок 1. Кластеризация методом K-Means. Данные приведены к размерности 2.

2. Стандартизировать данные
3. Проведена кластеризация методом DBSCAN

```
Cluster labels: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, -1}
Total clusters: 36
Non-clustered data: 0.7512737378415933
```

Без предварительной настройки более 75% данных остались не кластеризованными.

4. Графики зависимости количества кластеров и процента некластеризованных наблюдений от максимальной рассматриваемой дистанции (рис. 2)

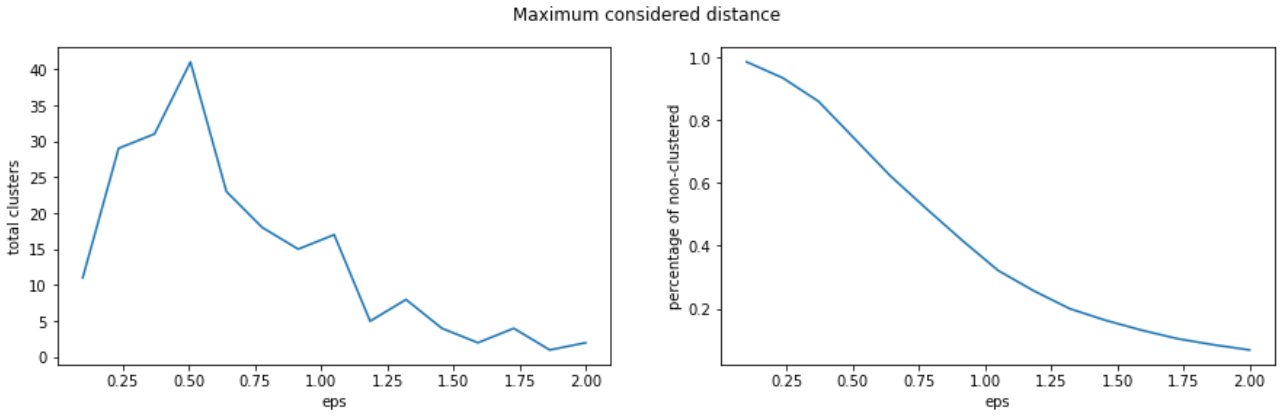

Рисунок 2. Количество кластеров и процент некластеризованных данных от максимальной рассматриваемой дистанции.

5. Графики зависимости количества кластеров и процента некластеризованных наблюдений от минимального количества точек, образующих кластер (рис. 3)

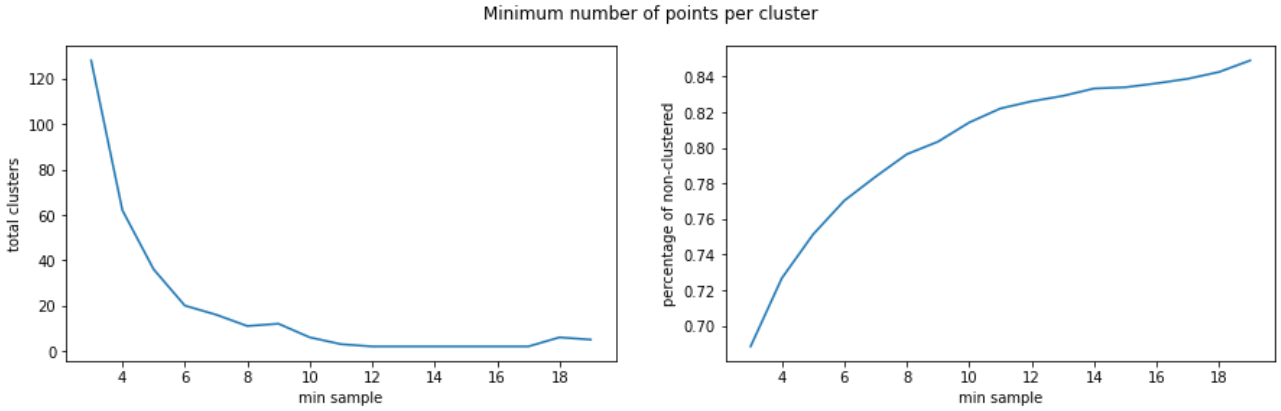

Рисунок 3. Количество кластеров и процент некластеризованных данных от минимального количества точек, образующих кластер.

6. Определены значения параметров *eps* и *min_samples* для которых количество кластеров от 5 до 7, а процент некластеризованных наблюдений не превышает 12%.

   В промежутке eps=[0.1, 3.0] и min_samples=[2, 30] найдены значения для которых количество кластеров находится в промежутке от 5 до 7, а количество некластеризованных данных не превышает 12% (табл. 2)

	min_samples	eps	non_clustered	clusters
0	3	2.0	0.062876	6
1	3	2.6	0.030917	5
2	3	2.7	0.027096	5
3	3	2.9	0.022233	5
4	3	3.0	0.019569	5
5	4	1.7	0.102478	5

Таблица 2. Результаты поиска наиболее подходящих параметров.

Наиболее подходящими параметрами выбраны *eps=3.0*, *min_samples=3*. При таких вводных, наименьший процент некластеризованных данных.

7. Визуализация данных с пониженной размерностью.

   Проведена кластеризация данных, после чего размерность понижена до 2 с помощью метода главных компонент (рис. 4).

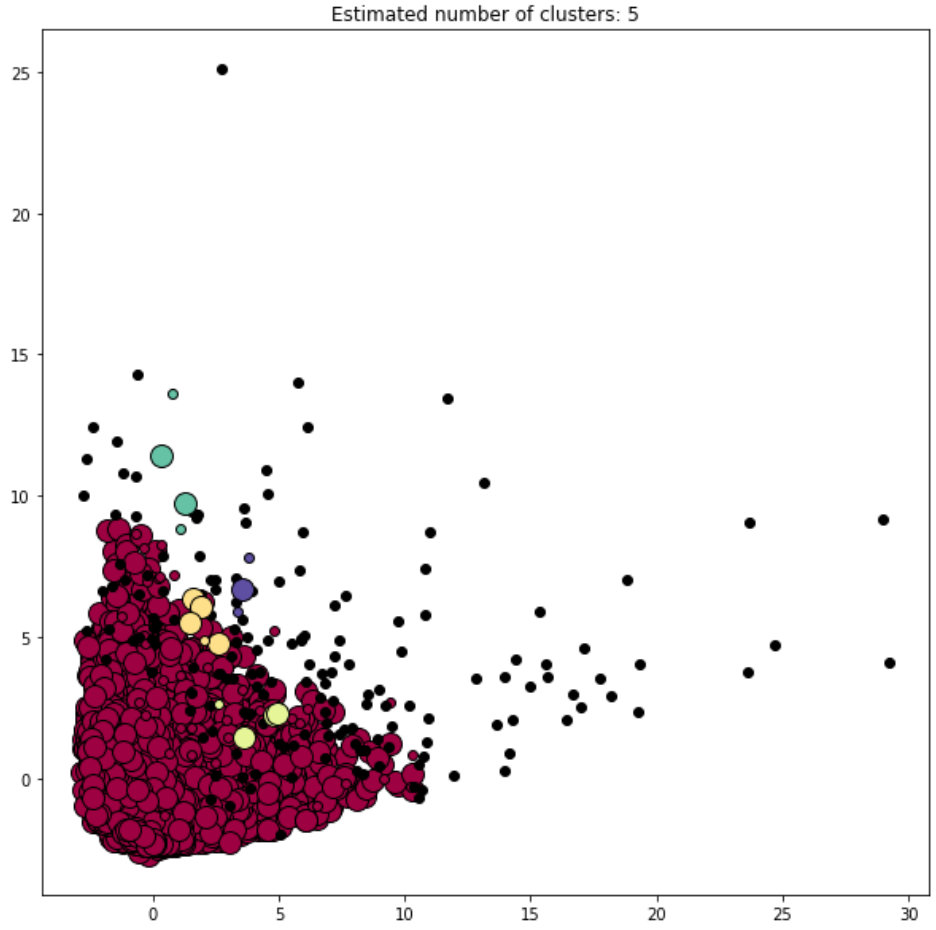

Рисунок 4. Результат кластеризации методом DBSCAN. *eps=3.0, min_samples=3.*

Параметры, которые принимает DBSCAN представлены в таблице 3.

Параметр	**Описание**
eps: float = 0.5	Максимальное расстояние между наблюдениями, чтобы они считались соседними (радиус окрестности наблюдения
min_samples: int = 5	Количество наблюдений вокруг точки, чтобы считать ее базовой
metric: string[1] or callable = "euclidean"	Метрика вычисления расстояния
metric_params: dict = None	Набор параметров для метрики, заданной функцией
algorithm: string[2] = "auto"	Алгоритм поиска ближайших соседей
leaf_size: int = 30	Размер листьев дерева алгоритмов *BallTree* и *KDTree*
p: float = None	Параметр для метрики Минковского
n_jobs: int = None	Количество параллельных рутин, в которых вычисляются ближайшие соседи. -1 означает использование всех процессоров

Таблица 3. Параметры метода DBSCAN

1: метрики scikit-learn[‘cityblock’, ‘cosine’, ‘euclidean’, ‘l1’, ‘l2’, ‘manhattan’]; метрики scipy.spatial.distance[‘braycurtis’, ‘canberra’, ‘chebyshev’, ‘correlation’, ‘dice’, ‘hamming’, ‘jaccard’, ‘kulsinski’, ‘mahalanobis’, ‘minkowski’, ‘rogerstanimoto’, ‘russellrao’, ‘seuclidean’, ‘sokalmichener’, ‘sokalsneath’, ‘sqeuclidean’, ‘yule’]

2: алгоритмы [‘auto’,’ball_tree’,’kd_tree’,’brute’]

OPTICS

1. Параметры метода *OPTICS* (табл. 4)

Параметр	Описание
max_eps: float = ∞	Максимальное расстояние между наблюдениями, чтобы они считались соседними (радиус окрестности наблюдения)
min_samples: int > 1 or float (0,1) = 5	Количество наблюдений вокруг точки, чтобы считать ее базовой
metric: string[1] or callable = "minkowski"	Метрика вычисления расстояния
p: int = 2	Параметр для метрики Минковского
metric_params: dict = None	Набор параметров для метрики, заданной функцией
cluster_method: ("xi", "dbscan") = "xi"	Метод извлечения кластеров
eps: float = None	Максимальное расстояние между наблюдениями, чтобы они считались соседними (радиус окрестности наблюдения). Используется при `cluster_method='dbscan'`
xi: float(0,1) = 0.5	Определяет минимальную крутизну на графике достижимости, который составляет границу кластера. Используется при `cluster_method='xi'`
predecessor_correction: bool = True	Коррекция кластеров в соответствии с предшественниками, рассчитанными *OPTICS*. Используется при `cluster_method='xi'`
min_cluster_size: int > 0 of float(0, 1) = None	Минимальное количество выборок в кластере *OPTICS*, выраженное в виде абсолютного числа доли от количества выборок. Используется при `cluster_method='xi'`
algorithm: string[2] = "auto"	Алгоритм поиска ближайших соседей
leaf_size: int =30	Размер листьев дерева алгоритмов *BallTree* и *KDTree*
memory: string or object[3] = None	Используется для кеширования дерева вычислений. Являясь строкой, определяет путь кеширования
n_jobs: int = None	Количество параллельных рутин, в которых вычисляются ближайшие соседи. -1 означает использование всех процессоров

Таблица 4. Параметры метода OPTICS

1: метрики scikit-learn[‘cityblock’, ‘cosine’, ‘euclidean’, ‘l1’, ‘l2’, ‘manhattan’]; метрики scipy.spatial.distance[‘braycurtis’, ‘canberra’, ‘chebyshev’, ‘correlation’, ‘dice’, ‘hamming’, ‘jaccard’, ‘kulsinski’, ‘mahalanobis’, ‘minkowski’, ‘rogerstanimoto’, ‘russellrao’, ‘seuclidean’, ‘sokalmichener’, ‘sokalsneath’, ‘sqeuclidean’, ‘yule’]

2: алгоритмы [‘auto’,’ball_tree’,’kd_tree’,’brute’]

3: Объект с интерфейсом joblib.Memory

2. Параметры *max_eps* и *min_samples*, при которых результаты близки к результатам пункта 6 *DBSCAN*.

   Вручную найдены параметры max_eps=2.0, min_samples=3, которые удовлетворяют условию. С метрикой по умолчанию (“minkowski”, *p=2*) происходит кластеризация на 6 кластеров 6.3% некластеризованных данных.

3. Визуализация полученных результатов (рис. 5), а также график достижимости (рис. 6).

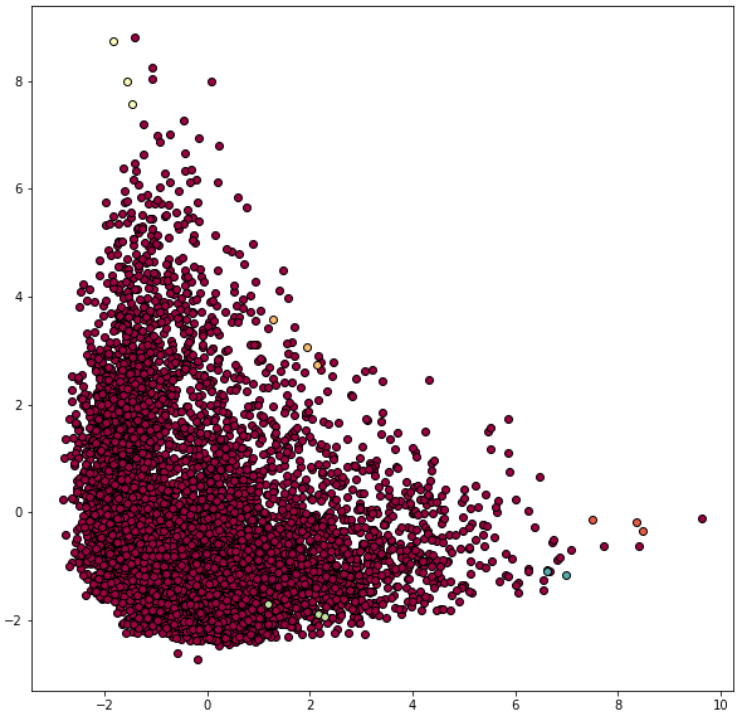

Рисунок 5. Данные, кластеризованные методом *OPTICS* с метрикой Минковского. *max_eps=2.0, min_samples=3.*

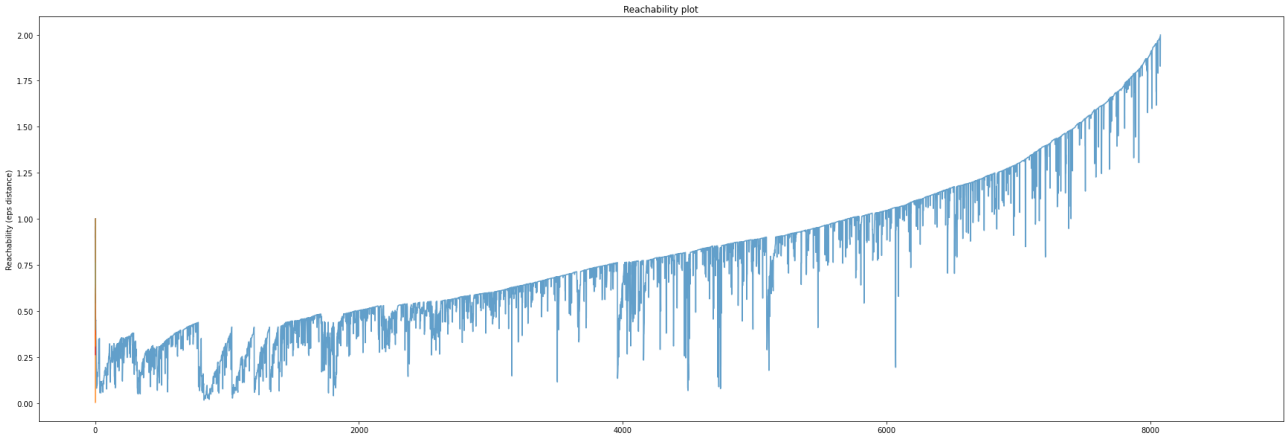

Рисунок 6. График достижимости с метрикой Минковского.

4. Исследование работы OPTICS при различных метриках.

   Метрика *«euclidean»* рисунки 7 и 8.

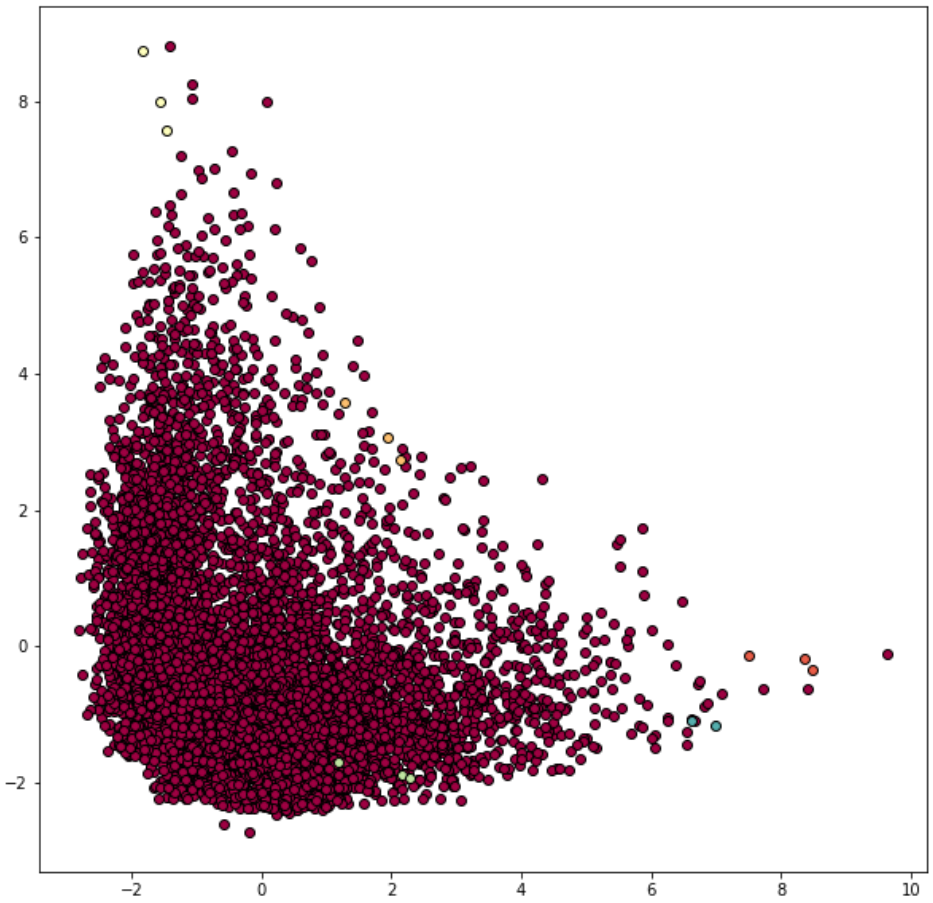

Рисунок 7. Данные, кластеризованные методом *OPTICS* с метрикой *euclidean*. *max_eps=2.0, min_samples=3.*

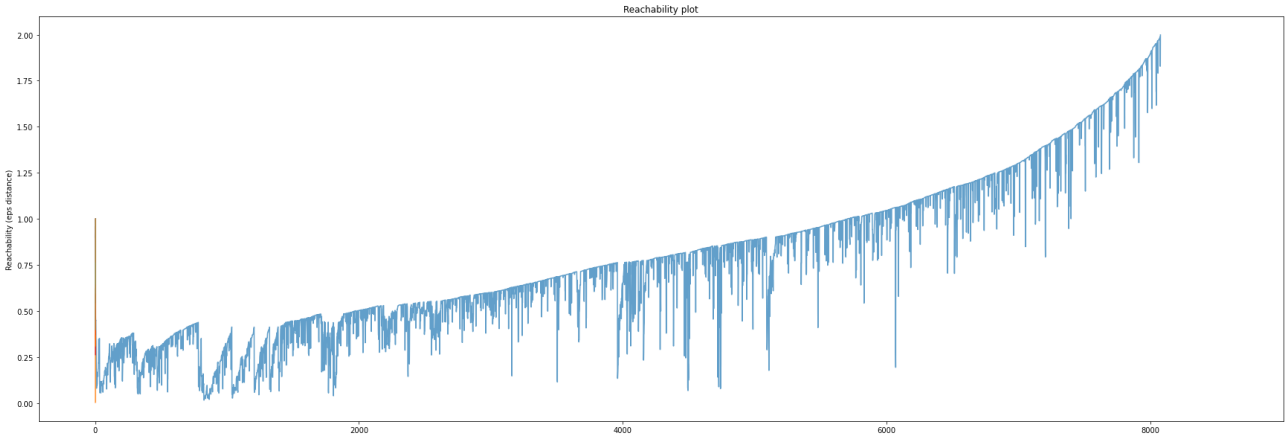

Рисунок 8. График достижимости с метрикой *euclidean*.

Метрика «canberra» рисунки 9 и 10.

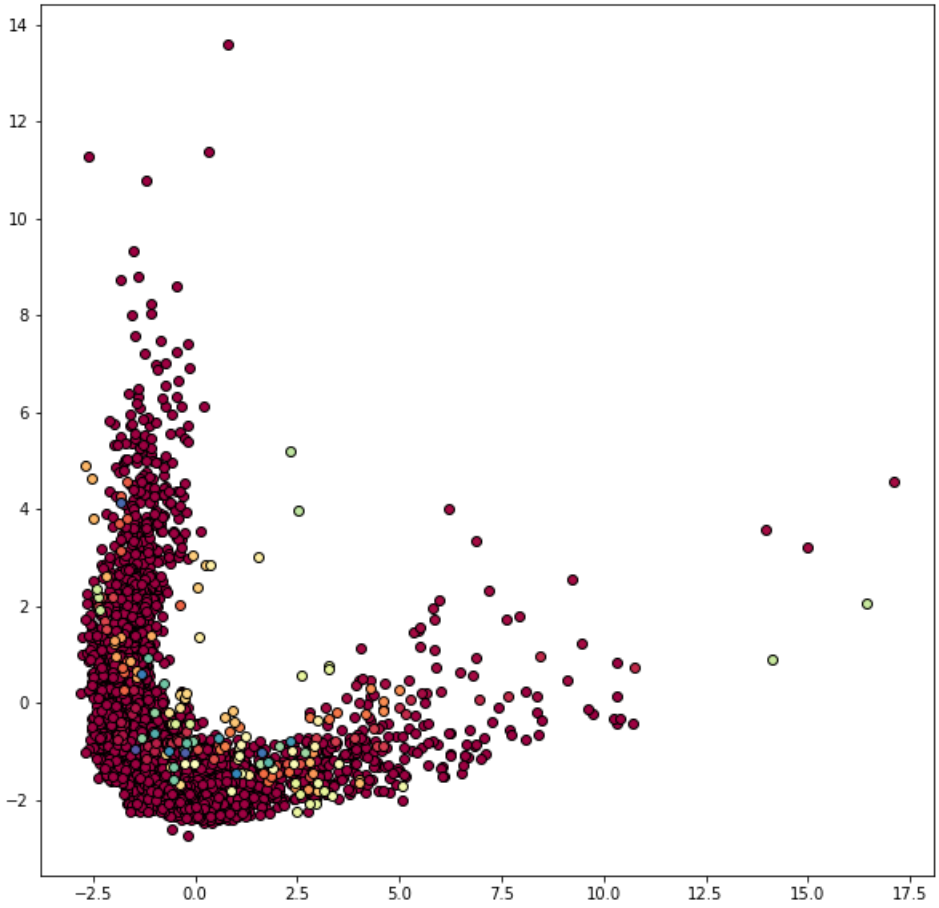

Рисунок 9. Данные, кластеризованные методом *OPTICS* с метрикой *canberra*. *max_eps=2.0, min_samples=3.*

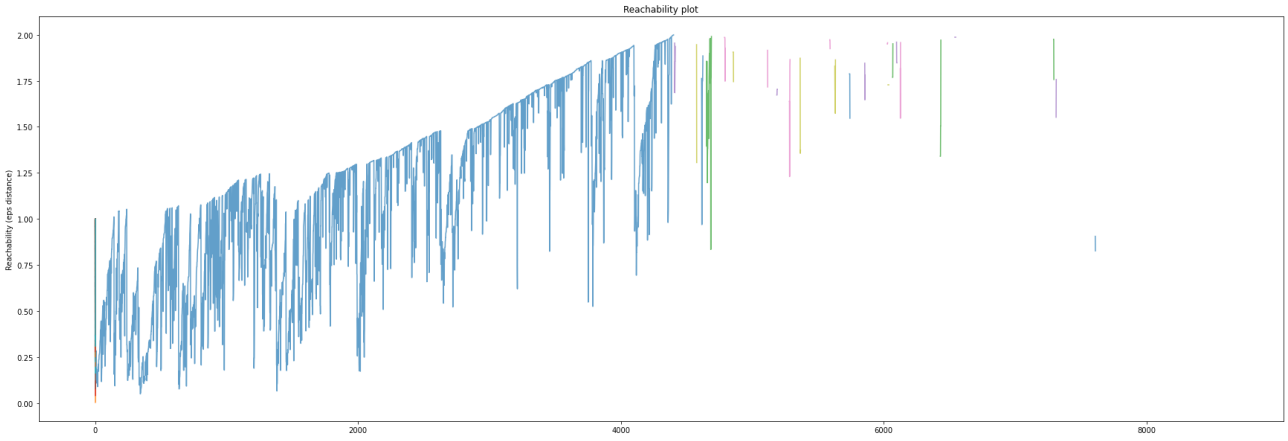

Рисунок 10. График достижимости с метрикой *canberra*.

Метрика «chebyshev» рисунки 11 и 12.

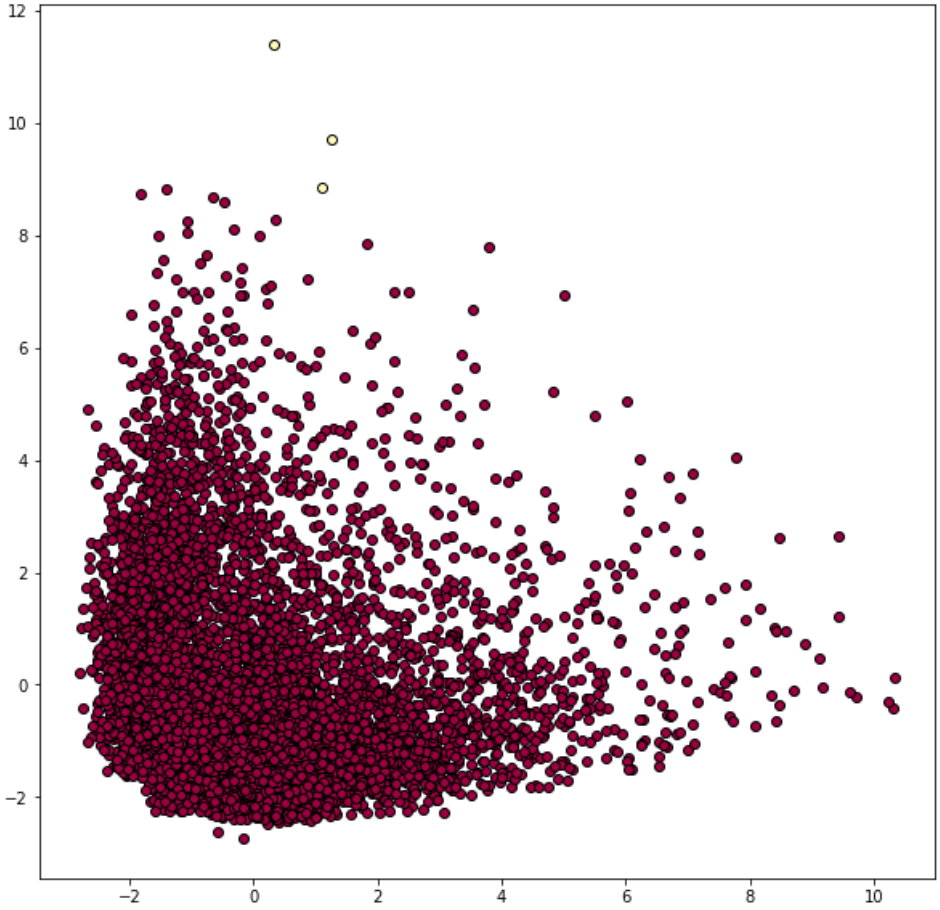

Рисунок 11. Данные, кластеризованные методом *OPTICS* с метрикой *chebyshev*. *max_eps=2.0, min_samples=3*

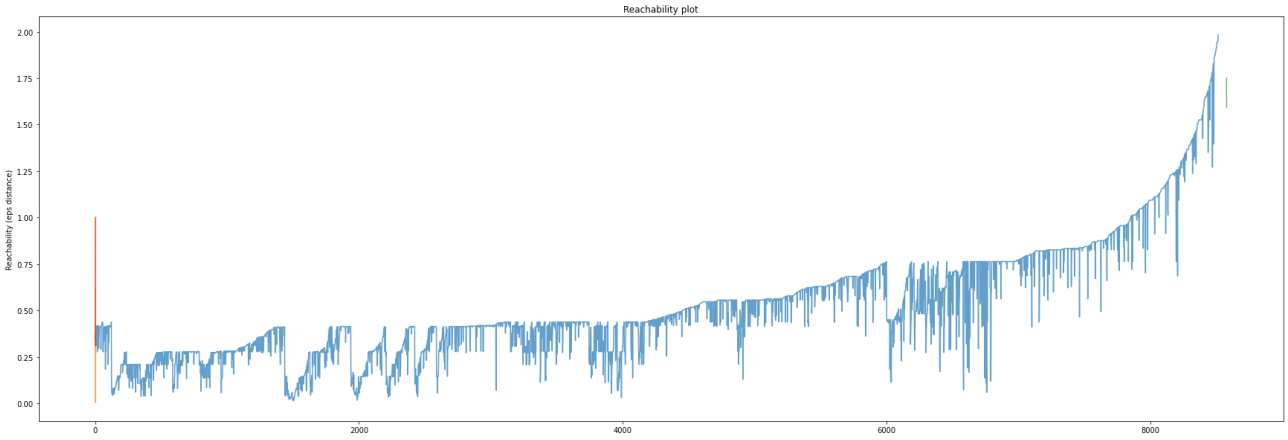

Рисунок 12. График достижимости с метрикой *chebyshev*.

Метрика «manhattan» рисунки 13 и 14.

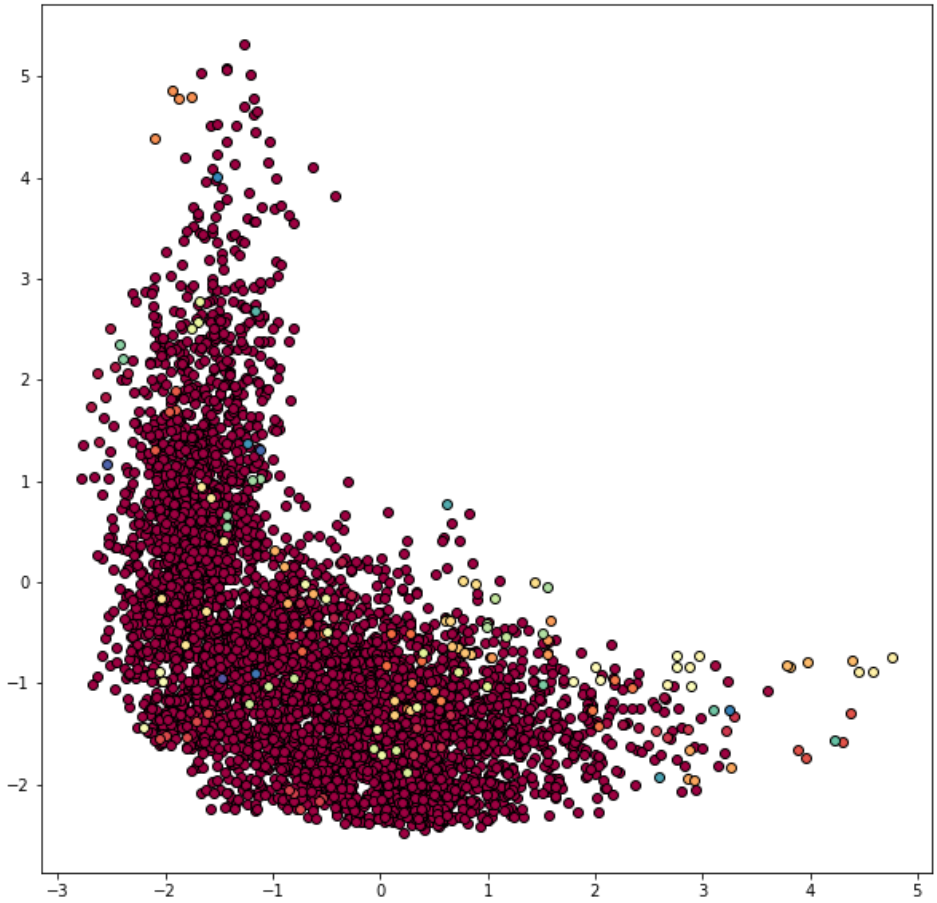

Рисунок 13. Данные, кластеризованные методом *OPTICS* с метрикой *manhattan*. *max_eps=2.0, min_samples=3*.

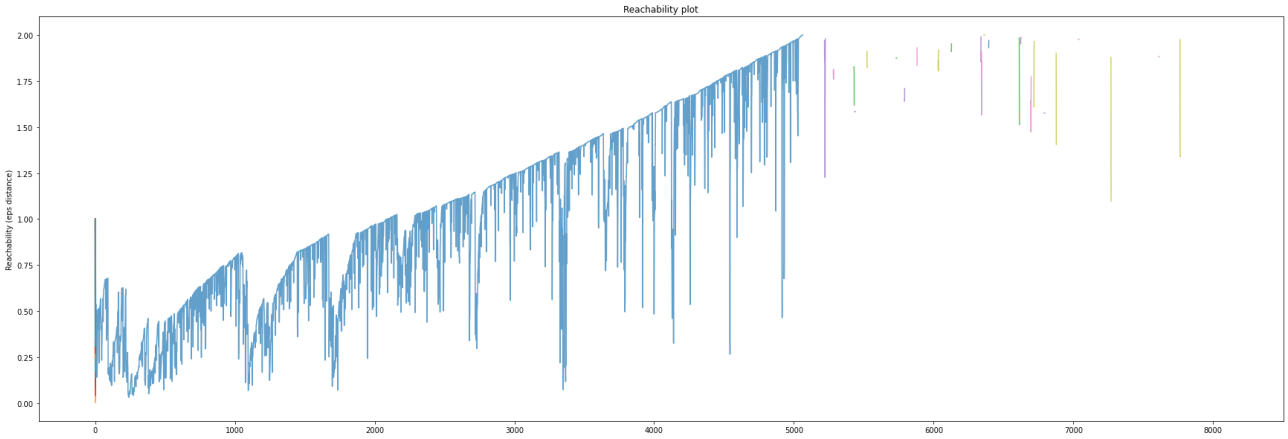

Рисунок 14. График достижимости с метрикой *manhattan*.

Метрика «sqeuclidean» рисунки 15 и 16.

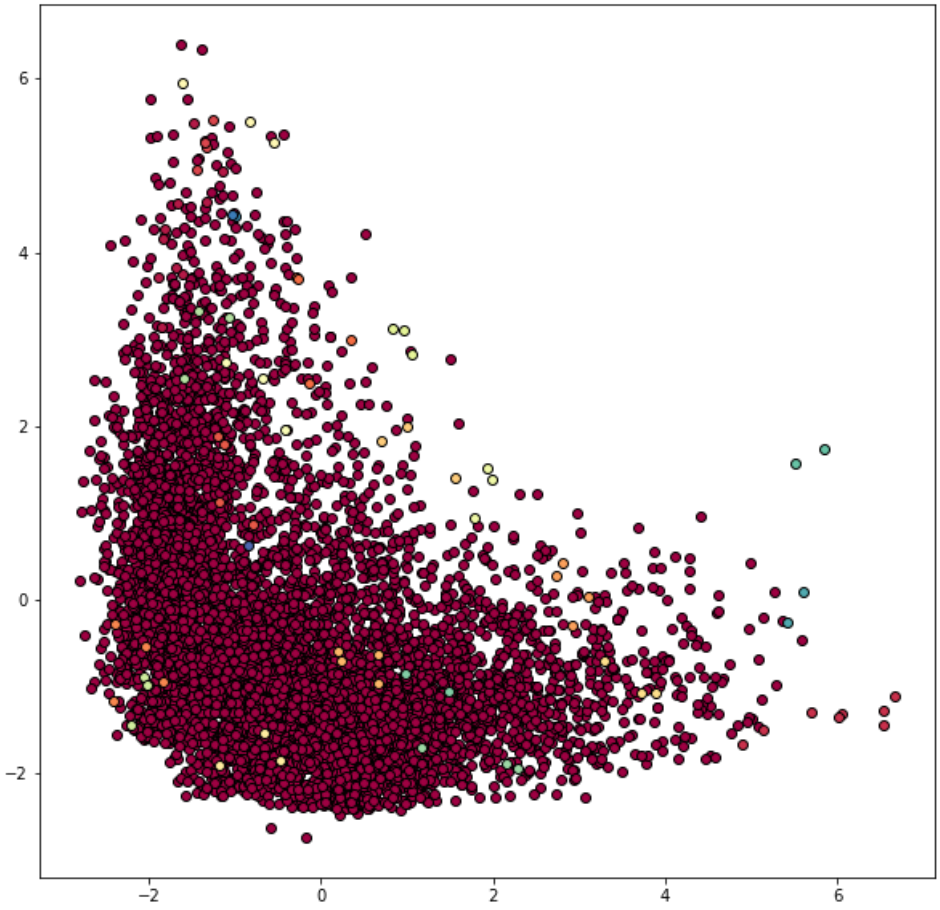

Рисунок 15. Данные, кластеризованные методом *OPTICS* с метрикой *sqeuclidean*. *max_eps=2.0, min_samples=3.*

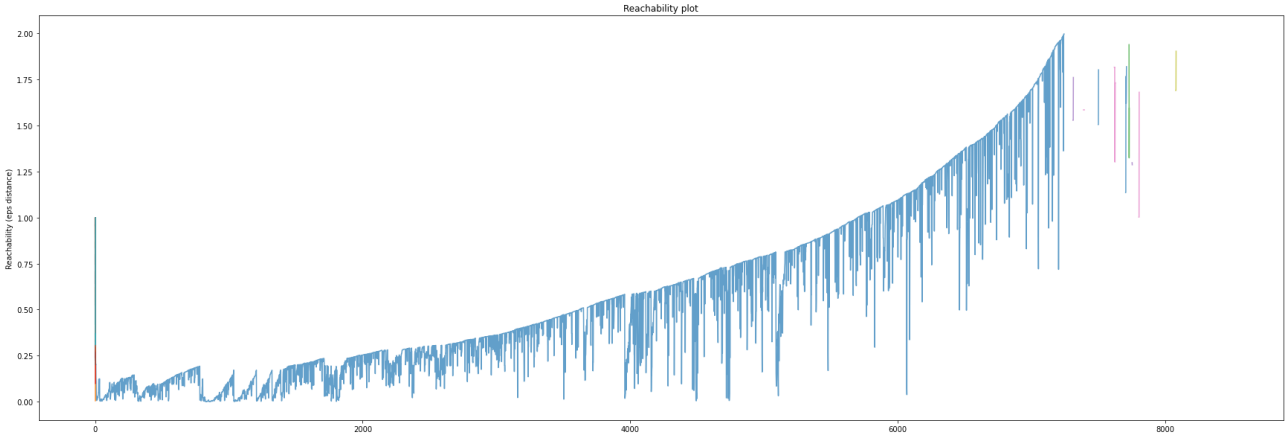

Рисунок 16. График достижимости с метрикой *sqeuclidean*.

Метрика *«minkowski»* при *p=3* рисунки 17, 18.

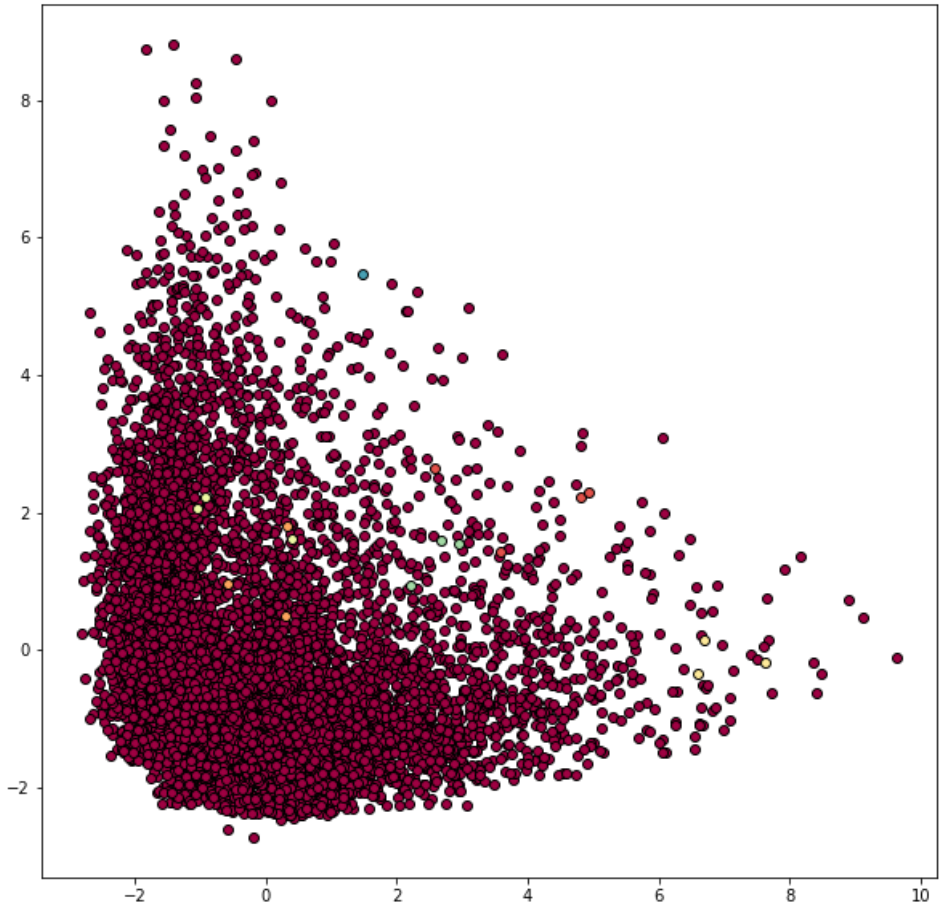

Рисунок 17. Данные, кластеризованные методом OPTICS с метрикой *minkowski p=3*. *max_eps=2.0, min_samples=3.*

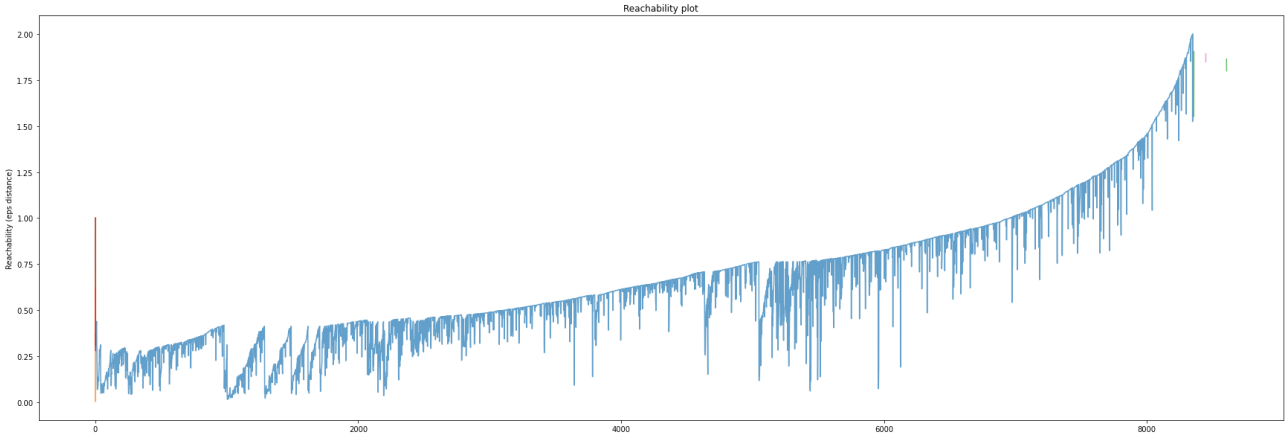

Рисунок 18. График достижимости с метрикой *minkowski p=3*.

Результаты исследования различных метрик представлены в таблице 5.

Метрика	Количество кластеров	Процент выпавших наблюдений
minkowski	7	3.11
euclidean	6	6.31
canberra	60	46.41
chebyshev	2	1.33
manhattan	55	39.50
sqeuclidean	25	15.17

Таблица 5. Результат исследования различных метрик *OPTICS* при *max_eps=2.0*, *min_samples=3*.

Вывод

В ходе лабораторной работы были изучены методы кластеризации *DBSCAN* и *OPTICS*.

DBSCAN опирается на два параметра, которые определяют его работу: максимальный радиус наблюдений, и количество точек на радиус, составляющих базовое состояние. Алгоритм среди всех точек ищет ту, в радиусе которой находится достаточно соседей, чтоб считать ее базовой, затем рассматриваются радиусы соседних точек и так пока не появится новых соседей. После чего ищутся следующие точки, которые удовлетворяют начальным условиям.

OPTICS куда менее опирается на заданные параметры создает граф достижимости. Выбрав из точек подходящие ядра, начинают рассматриваться соседи. Если отобразить график расстояния от кластера до следующего соседа, то можно заметить, что данные, которые разбиты на несколько облаков представятся на таком графике в виде «долин» и «пиков». Пики на таком графике определяют разграничение кластеров. Если горизонтальной линией обрезать значение максимального расстояния, то можно получить искомые кластеры. Однако более удачным методом является определение скачков значений достижимых расстояний.

Для данного набора данных оба метода не являются удачным решением, поскольку данные представляют собой единое облако.